Journal of Machine Learning Research 8 (2007) 1687-1723 Submitted 1/07; Revised 4/07; Published 8/07

A Nonparametric Statistical Approach to Clustering via Mode Identification

Jia Li JIALI@STAT.PSU.EDU
Department of Statistics
The Pennsylvania State University
University Park, PA 16802, USA

Surajit Ray SRAY@MATH.BU.EDU
Department of Mathematics and Statistics
Boston University
Boston, MA 02215, USA

Bruce G. Lindsay BGL@STAT.PSU.EDU
Department of Statistics
The Pennsylvania State University
University Park, PA 16802, USA

Editor: Charles Elkan

Abstract

A new clustering approach based on mode identification is developed by applying new optimization techniques to a nonparametric density estimator. A cluster is formed by those sample points that ascend to the same local maximum (mode) of the density function. The path from a point to its associated mode is efficiently solved by an EM-style algorithm, namely, the Modal EM (MEM). This method is then extended for hierarchical clustering by recursively locating modes of kernel density estimators with increasing bandwidths. Without model fitting, the mode-based clustering yields a density description for every cluster, a major advantage of mixture-model-based clustering. Moreover, it ensures that every cluster corresponds to a bump of the density. The issue of diagnosing clustering results is also investigated. Specifically, a pairwise separability measure for clusters is defined using the ridgeline between the density bumps of two clusters. The ridgeline is solved for by the Ridgeline EM (REM) algorithm, an extension of MEM. Based upon this new measure, a cluster merging procedure is created to enforce strong separation. Experiments on simulated and real data demonstrate that the mode-based clustering approach tends to combine the strengths of linkage and mixture-model-based clustering. In addition, the approach is robust in high dimensions and when clusters deviate substantially from Gaussian distributions. Both of these cases pose difficulty for parametric mixture modeling. A C package on the new algorithms is developed for public access at http://www.stat.psu.edu/∼jiali/hmac.

Keywords: modal clustering, mode-based clustering, mixture modeling, modal EM, ridgeline EM, nonparametric density

1. Introduction

Clustering is a technology employed in tremendously diverse areas for a multitude of purposes. It simplifies massive data by extracting essential information, based on which many subsequent analysis or processes become feasible or more efficient. For instance, in information systems, clustering

is applied to text documents or images to speed up indexing and retrieval (Li, 2005a). Clustering can be a stand-alone process. For example, microarray gene expression data are often clustered in order to find genes with similar functions. Clustering is also the technical core of several prototype-based supervised learning algorithms (Hastie et al., 2001) and has been extended to non-vector data in this regard (Li and Wang, 2006). Recent surveys (Kettenring, 2006; Jain et al., 1999) discuss the methodologies, practices, and applications of clustering.

1.1 Background

Clustering methods fall roughly into three types. The first type uses only pairwise distances between objects to be clustered. These methods enjoy wide applicability since a tractable mathematical representation for objects is not required. However, they do not scale well with large data sets due to the quadratic computational complexity of calculating all the pairwise distances. Examples include linkage clustering (Gower and Ross, 1969) and spectral graph partitioning (Pothen et al., 1990). The second type targets on optimizing a given merit function. The merit function reflects the general belief about good clustering, that is, objects in the same cluster should be similar to each other while those in different clusters be as distinct as possible. Different algorithms vary in the similarity measure and the criterion for assessing the global quality of clustering. K-means and k-center clustering (Gonzalez, 1985) belong to this type.

The third type relies on statistical modeling (Fraley and Raftery, 2002). In particular, each cluster is characterized by a basic parametric distribution (referred to as a component), for instance, the multivariate Gaussian for continuous data and the Poisson distribution for discrete data. The overall probability density function (pdf) is a mixture of the parametric distributions (McLachlan and Peel, 2000). The clustering procedure involves first fitting a mixture model, usually by the EM algorithm, and then computing the posterior probability of each mixture component given a data point. The component possessing the largest posterior probability is chosen for that point. Points associated with the same component form one cluster. Moreover, the component posterior probabilities evaluated in mixture modeling can be readily used as a soft clustering scheme. In addition to partitioning data, a probability distribution is obtained for each cluster, which can be helpful for gaining insight into the data. Another advantage of mixture modeling is its flexibility in treating data of different characteristics. For particular applications, mixtures of distributions other than Gaussian have been explored for clustering (Banfield and Raftery, 1993; Li and Zha, 2006). Banerjee et al. (2005) have also used the mixture of Mises-Fisher distributions to cluster data on a unit sphere.

The advantages of mixture modeling naturally result from its statistical basis. However, the parametric assumptions about cluster distributions are found restrictive in some applications. Li (2005b) addresses the problem by assuming each cluster itself is a mixture of Gaussians, providing greater flexibility for modeling a single cluster. This method involves selecting the number of components for each cluster and is sensitive to initialization. Although some success has been shown using the Bayesian Information Criterion (BIC), choosing the right number of components for a mixture model is known to be difficult, especially for high dimensional data.

Another limitation for mixture modeling comes from the sometimes serious disparity between a component and a cluster complying to geometric heuristics. If a probability density is estimated from the data, preferably, every cluster corresponds to a unique "bump" in the density resulting from a tight mass of data. We refer to a bump as a "hill" and the local maximum associated with it the

"hilltop", that is, the mode. The rational for clustering by a mixture model is that if the component distributions each possess a single hilltop, by fitting a mixture of them, every component captures one separate hilltop in the data. However, this is often not true. Without careful placement and control of their shapes, the mixture components may not align with the hills of the density, especially when clusters are poorly separated or the assumed parametric component distribution is violated. It is known that two Gaussians located sufficiently close result in a single mode. (On the other hand, a two component multivariate Gaussian mixture can have more than two modes, as shown by Ray and Lindsay, 2005). In this case, equating a component with a cluster is questionable. This profound limitation of mixture modeling has not been adequately investigated. In fact, even to quantify the separation between components is not easy.

Here, we develop a new nonparametric clustering approach, still under a statistical framework. This approach inherits the aforementioned advantages of mixture modeling. Furthermore, data are allowed to reveal a nonparametric distribution for each cluster as part of the clustering procedure. It is also guaranteed that every cluster accounts for a distinct hill of the probability density.

1.2 Clustering via Mode Identification

To avoid restrictions imposed by parametric assumptions, we model data using kernel density functions. By their nature, such densities have a mixture structure. Given a density estimate in the form of a mixture, a new algorithm, aptly called the Modal EM (MEM), enables us to find an increasing path from any point to a local maximum of the density, that is, a hilltop. Our new clustering algorithm groups data points into one cluster if they are associated with the same hilltop. We call this approach modal clustering. A new algorithm, namely the Ridgeline EM (REM), is also developed to find the ridgeline linking two hilltops, which is proven to pass through all the critical points of the mixture density of the two hills.

The MEM and REM algorithms allow us to exploit the geometry of a probability density function in a nontrivial manner. As a result, clustering can be conducted in accurate accordance with our geometric heuristics. Specifically, every cluster is ensured to be associated with a hill, and every sample point in the cluster can be moved to the corresponding hilltop along an ascending path without crossing any "valley" that separates two hills. Moreover, by finding the ridgeline between two hilltops, the way two hills separate from each other can be adequately measured and exhibited, enabling diagnosis of clustering results and application-dependent adjustment of clusters. Modal clustering using MEM also has practical advantages such as the irrelevance of initialization and the ease of implementing required optimization techniques.

Our modal clustering algorithm is not restricted to kernel density estimators. In fact, it can be used to find the modes of any density in the form of a mixture distribution. It is known that when the number of components in a mixture increases, as long as there are sufficiently many components, the overall fitted density of the mixture is not sensitive to that number. On the other hand, the resulting partition of data can change dramatically if we identify each mixture component with a cluster, as normally practiced in mixture-model-based clustering. In modal clustering, there is no such identification, and mixture components only play the role of approximating a density. We thus have much more flexibility at choosing mixture distributions. Specifically, we adopt the fully nonparametric kernel density estimation, using Gaussian kernels for continuous data.

We summarize the main contributions of this paper as follows:

- A new nonparametric statistical clustering algorithm and its hierarchical extension are developed by associating data points to their modes identified by MEM. Approaches to improve computational efficiency and to visualize cluster structures for high dimensional data are presented.

- The REM algorithm is developed to find the ridgeline between the modes of any two clusters. Measures for the pairwise separability between clusters are proposed using ridgelines. A cluster merging algorithm to enhance modal clustering is developed.

- Experiments are conducted using both simulated and real data sets. Comparisons are made with several other clustering algorithms such as linkage clustering, k-means, and Gaussian mixture modeling.

1.3 Related Work

Clustering is an extensively studied research topic with vast existing literature (see Jain et al., 1999; Everitt et al., 2001, for general coverage). Works most related to ours are mode-based clustering methods independently developed in the communities of pattern recognition (Leung et al., 2000) and statistics (Cheng et al., 2004). The MEM and REM algorithms, the cluster diagnosis tool, and cluster merging procedure built upon REM are unique to our work.

In pattern recognition, mode-based clustering is studied under the name of scale-space method, inspired by the smoothing effect of the human visual system. The scale-space clustering method is pioneered by Wilson and Spann (1990), and furthered studied by Roberts (1997) using density estimation and by Chakravarthy and Ghosh (1996) using the radial basis function neural network. Leung et al. (2000) forms a function called "space scale image" for a data set. This function is essentially a Gaussian kernel density estimate (differing from a true density by an ignorable constant). The modes of the density function are solved by numerical methods. To associate a data point with a mode, a gradient dynamic system starting from the point is defined and solved by the Euler difference method. A hierarchical clustering algorithm is proposed by gradually increasing the Gaussian kernel bandwidth. The authors also note the non-nested nature of clustering results obtained from increasing bandwidths.

It can be shown that the iteration equation derived from the Euler difference method is identical to that from MEM. However, MEM applies generally to any mixture of Gaussians as well as mixtures of other parametric distributions. Its ascending property is proved rather than based on approximation. Under the framework of scale-space clustering, general mixtures do not arise as a concern, and naturally, only the case of Gaussian kernel density is discussed (Leung et al., 2000). It is not clear whether the gradient dynamic system can be efficiently solved for general mixture models. Our hierarchical clustering algorithm differs slightly from that of Leung et al. (2000) by enforcing nested clustering. This difference only reflects an algorithmic preference and is not intrinsic to the key ideas of modal clustering.

Cheng et al. (2004) defined a gradient tree to be the set of steepest ascent curves of a kernel density estimate, treating each sample point as the starting position of a curve. The gradient curves are similar to the paths solved by the gradient dynamic system of Leung et al. (2000), but are computed by discrete approximation. Minnotte and Scott (1993) developed the mode tree as a visualization tool for nonparametric density estimate. The emphasis is on graphically illustrating the

relationship between kernel bandwidths and the modes of uni- or bivariate kernel density functions. Minnotte et al. (1998) also extended the mode tree to the mode forest.

Because clustering of independent vectors has been a widely used method for image segmentation, especially in the early days (Jain and Dubes, 1988), we test the efficiency of our algorithm on segmentation, a good example of computationally intensive applications. We have no intention to present our clustering algorithm as a state-of-the-art segmentation method although it may well be applied as a fast method. We note that much advance has been achieved in image segmentation using approaches beyond the framework of clustering independent vectors (Pal and Pal, 1993; Shi and Malik, 2000; Joshi et al., 2006).

The rest of the paper is organized as follows. Notations and the MEM algorithm are introduced in Section 2. In Section 3, a new clustering algorithm and its hierarchical extension are developed by associating data points with the modes of a kernel density estimate. In Section 4, the REM algorithm for finding ridgelines between modes is presented. In addition, several measures of pairwise separability between clusters are defined, which lead to the derivation of a new cluster merging algorithm. This merging method strengthens the framework of modal clustering. In Section 5, we present a method to visualize high dimensional data so that the discrimination between clusters is well preserved. Experimental results on both simulated and real data sets and comparisons with other clustering approaches are provided in Section 6. Finally, we conclude and discuss future work in Section 7.

2. Preliminaries

We introduce in this section the *Modal EM (MEM)* algorithm that solves a local maximum of a mixture density by ascending iterations starting from any initial point. The algorithm is named Modal EM because it comprises two iterative steps similar to the expectation and the maximization steps in EM (Dempster et al., 1977). The objective of the MEM algorithm is different from the EM algorithm. The EM algorithm aims at maximizing the likelihood of data over the parameters of an assumed distribution. The goal of MEM is to find the local maxima, that is, modes, of a given distribution.

Let a mixture density be $f(x) = \sum_{k=1}^{K} \pi_k f_k(x)$, where $x \in \mathcal{R}^d$, π_k is the prior probability of mixture component k, and $f_k(x)$ is the density of component k. Given any initial value $x^{(0)}$, MEM solves a local maximum of the mixture by alternating the following two steps until a stopping criterion is met. Start with $r = 0$.

1. Let
$$p_k = \frac{\pi_k f_k(x^{(r)})}{f(x^{(r)})}, \; k = 1, ..., K.$$
2. Update
$$x^{(r+1)} = \operatorname*{argmax}_x \sum_{k=1}^{K} p_k \log f_k(x).$$

The first step is the "Expectation" step where the posterior probability of each mixture component k, $1 \le k \le K$, at the current point $x^{(r)}$ is computed. The second step is the "Maximization" step. We assume that $\sum_{k=1}^{K} p_k \log f_k(x)$ has a unique maximum, which is true when the $f_k(x)$ are normal densities. In the special case of a mixture of Gaussians with common covariance matrix, that is, $f_k(x) = \phi(x \mid \mu_k, \Sigma)$, where $\phi(\cdot)$ is the pdf of a Gaussian distribution, we simply have

$x^{(r+1)} = \sum_{k=1}^{K} p_k \mu_k$. For other parametric densities $f_k(x)$, the solution to the maximization in the second step can be more complicated and sometimes requires numerical procedures. On the other hand, similarly as in the EM algorithm, it is usually much easier to maximize $\sum_{k=1}^{K} p_k \log f_k(x)$ than the original objective function $\log \sum_{k=1}^{K} \pi_k f_k(x)$.

The proof of the ascending property of the MEM algorithm is provided in Appendix A. We omit a rigorous discussion regarding the convergence of $x^{(r)}$ here. By Theorem 1 of Wu (1983), if $f(x)$ is a mixture of normal densities, all the limit points of $\{x^{(r)}\}$ are stationary points of $f(x)$. It is possible that $\{x^{(r)}\}$ converges to a stationary, but not locally maximal, point, although we have not observed this in our experiments. We refer to (Wu, 1983) for a detailed treatment of the convergence properties of EM style algorithms.

3. Clustering by Mode Identification

We focus on clustering continuous vector data although the framework extends readily to discrete data. Given a data set $\{x_1, x_2, ..., x_n\}, x_i \in \mathcal{R}^d$, a probability density function for the data is estimated nonparametrically using Gaussian kernels. As the kernel density estimate is in the form of a mixture distribution, MEM is applied to find a mode using every sample point x_i, $i = 1, ..., n$, as the initial value for the iteration. Two points x_i and x_j are grouped into one cluster if the same mode is obtained from both. When the variances of Gaussian kernels increase, the density estimate becomes smoother and tends to group more points into one cluster. A hierarchy of clusters can thus be constructed by gradually increasing the variances of Gaussian kernels. Next, we elaborate upon the clustering algorithm, illustrate it with an example, and discuss approaches to speed up computation.

3.1 The Algorithm

Let the set of data to be clustered be $S = \{x_1, x_2, ..., x_n\}$, $x_i \in \mathcal{R}^d$. The Gaussian kernel density estimate is formed:

$$f(x) = \sum_{i=1}^{n} \frac{1}{n} \phi(x \mid x_i, \Sigma),$$

where the Gaussian density function

$$\phi(x \mid x_i, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp(-\frac{1}{2}(x - x_i)^t \Sigma^{-1} (x - x_i)) .$$

We use a spherical covariance matrix $\Sigma = diag(\sigma^2, \sigma^2, ..., \sigma^2)$. The standard deviation σ is also referred to as the *bandwidth* of the Gaussian kernel. We use notation $D(\sigma^2) = diag(\sigma^2, \sigma^2, ..., \sigma^2)$ for brevity.

With a given Gaussian kernel covariance matrix $D(\sigma^2)$, data are clustered as follows:

1. Form kernel density

$$f(x \mid S, \sigma^2) = \sum_{i=1}^{n} \frac{1}{n} \phi(x \mid x_i, D(\sigma^2)), \tag{1}$$

2. Use $f(x|S, \sigma^2)$ as the density function. Use each x_i, $i = 1, 2, ..., n$, as the initial value in the MEM algorithm to find a mode of $f(x|S, \sigma^2)$. Let the mode identified by starting from x_i be $\mathcal{M}_\sigma(x_i)$.

3. Extract distinctive values from the set $\{\mathcal{M}_\sigma(x_i), i = 1,2,...,n\}$ to form a set G. Label the elements in G from 1 to $|G|$. In practice, due to finite precision, two modes are regarded equal if their distance is below a threshold.

4. If $\mathcal{M}_\sigma(x_i)$ equals the kth element in G, x_i is put in the kth cluster.

In the basic version of the algorithm, the density $f(x|S,\sigma^2)$ is a sum of Gaussian kernels centered at every data point. However, the algorithm can be carried out with any density estimate in the form of a mixture. The key step in the clustering algorithm is the identification of a mode starting from any x_i. MEM moves from x_i via an ascending path, or, figuratively, via hill climbing, to a mode. Points that climb to the same mode are located on the same hill and hence grouped into one cluster. We call this the *Mode Association Clustering (MAC)* algorithm.

Although the density of each cluster is not explicitly modeled by MAC, this nonparametric method retains a major advantage of mixture-model-based clustering, that is, a pdf is obtained for each cluster. These density functions facilitate soft clustering as well as cluster assignment of samples outside the data set. Denote the set of points in cluster k, $1 \le k \le |G|$, by C_k. The density estimate for cluster k is

$$g_k(x) = \sum_{x_i : x_i \in C_k} \frac{1}{|C_k|} \phi(x \mid x_i, D(\sigma^2)) \,. \tag{2}$$

Because we do not assume a parametric form for the densities of individual clusters, our method tends to be more robust and characterizes clusters more accurately when the attempted parametric assumptions are violated.

It is known in the literature of mixture modeling that if the density of a cluster is estimated using only points assigned to this cluster, the variance tends to be under estimated, although the effect on clustering may be small (Celeux and Govaert, 1993). The under estimation of variance becomes more severe for poorly separated clusters, which often decay towards zero too quickly on leaving the cluster. We will see a similar phenomenon here with $g_k(x)$ having over fast decaying tails. A correction to this problem in mixture modeling is to use soft instead of hard clustering. Every point is allowed to contribute to every cluster by a weight computed from the posterior probability of the cluster.

Under this spirit, we can make an ad-hoc modification on the density estimation. With $g_k(x)$ in (2) as the initial cluster density, compute the posterior of cluster k given each x_i by $p_{i,k} \propto \frac{|C_k|}{n} g_k(x)$, $k = 1, ..., |G|$, subject to $\sum_{k'=1}^{|G|} p_{i,k'} = 1$. Form the updated density of cluster k by

$$\tilde{g}_k(x) = \frac{\sum_{i=1}^n p_{i,k} \phi(x \mid x_i, D(\sigma^2))}{\sum_{i=1}^n p_{i,k}} \,.$$

With the cluster density modified, it is natural to question if $p_{i,k}$ should be updated again, which in turn will lead to another update of $\tilde{g}_k(x)$. This iterative procedure can be carried out infinitely. Whether it converges is not clear and may be worthy of investigation. On the other hand, if the maximum a posteriori clustering based on the final $\tilde{g}_k(x)$ differs significantly from the result of modal clustering, this procedure may have defeated the very purpose of modal clustering and turned it into merely an initialization scheme. We thus do not recommend many iterative updates on $\tilde{g}_k(x)$. One round of adjustment from $g_k(x)$ may be sufficient. Or one can take $g_k(x)$ cautiously as a smooth signature of a cluster, likely tighter than an accurate density estimate.

When the bandwidth σ increases, the kernel density estimate $f(x|S,\sigma^2)$ in (1) becomes smoother and more points tend to climb to the same mode. This suggests a natural approach for hierarchical clustering. Given a sequence of bandwidths $\sigma_1 < \sigma_2 < \cdots < \sigma_\eta$, hierarchical clustering is performed in a bottom-up manner. We start with every point x_i being a cluster by itself. The set of cluster representatives is thus $G_0 = S = \{x_1, ..., x_n\}$. This extreme case corresponds to the limit when σ approaches 0. At any bandwidth σ_l, the cluster representatives in G_{l-1} obtained from the preceding bandwidth are input into MAC using the density $f(x|S,\sigma_l^2)$. Note that the kernel centers remain at all the original data points although modes are identified only for cluster representatives when $l > 1$. The modes identified at this level form a new set of cluster representatives G_l. This procedure is repeated across all σ_l's. We refer to this hierarchical clustering algorithm as *Hierarchical MAC (HMAC)*. It corresponds to the mappings $x_i \to \mathcal{M}_{\sigma_1}(x_i) \to \mathcal{M}_{\sigma_2}(\mathcal{M}_{\sigma_1}(x_i)) \to \cdots$.

Denote the partition of points obtained at bandwidth σ_l by $\mathcal{P}_l$, a function mapping x_i's to cluster labels. If K clusters labeled 1, 2, ..., K, are formed at bandwidth σ_l, $\mathcal{P}_l(x_i) \in \{1, 2, ..., K\}$. HMAC ensures that $\mathcal{P}_l$'s are nested, that is, if $\mathcal{P}_l(x_i) = \mathcal{P}_l(x_j)$, then $\mathcal{P}_{l+1}(x_i) = \mathcal{P}_{l+1}(x_j)$. Recall that the set of cluster representatives at level l is G_l. HMAC starts with $G_0 = \{x_1, ..., x_n\}$ and solves G_l, $l = 1, 2, ..., \eta$, sequentially by the following procedure:

1. Form kernel density
$$f(x|S,\sigma_l^2) = \sum_{i=1}^{n} \frac{1}{n} \phi(x \mid x_i, D(\sigma_l^2)).$$

2. Cluster G_{l-1} by MAC using density $f(x|S,\sigma_l^2)$. Let the set of distinct modes obtained be G_l.

3. If $\mathcal{P}_{l-1}(x_i) = k$ and the kth element in G_{l-1} is clustered to the k'th mode in G_l, then $\mathcal{P}_l(x_i) = k'$. That is, the cluster of x_i at level l is determined by its cluster representative in G_{l-1}.

It is worthy to note that HMAC differs profoundly from linkage clustering, which also builds a hierarchy of clusters. In linkage clustering, at every level, only the two clusters with the minimum pairwise distance are merged. The hierarchy is constructed by a sequence of such small greedy merges. The lack of overall consideration tends to result in skewed clusters. In HMAC, however, at any level, the merging of clusters is conducted globally and the effect of every original data point on clustering is retained through the density $f(x|S,\sigma_l^2)$.

3.2 An Example

We now illustrate the HMAC algorithm using a real data set. This is the glass identification data provided by the UCI machine learning database repository (Blake et al., 1998). The original data were contributed by Spiehler and German at the Diagnostic Products Corporation. For clarity of demonstration, we take 105 sample points from two types of glass in this data set. Moreover, we only use the first two principal components of the original 9 dimensions.

HMAC is applied to the data using a sequence of kernel bandwidths $\sigma_1 < \sigma_2 < \cdots < \sigma_\eta$, $\eta = 20$, chosen equally spaced from $[0.1\hat{\sigma}, 2\hat{\sigma}] = [0.225, 4.492]$, where $\hat{\sigma}$ is the larger one of the sample standard deviations of the two dimensions. Among the 20 different σ_l's, only 6 of them result in clustering different from σ_{l-1}, reflecting the fact that the bandwidth needs to increase by a sufficient amount to drive the merging of some existing cluster representatives. The number of clusters at the 6 levels is sequentially $21, 11, 5, 3, 2, 1$.

We demonstrate the clustering results at the 2nd and 3rd level in Figure 1. At the 2nd level, 11 clusters are formed, as shown by different symbols in Figure 1(a). The 11 modes identified at level 2 are merged into 5 modes at level 3 when the bandwidth increases from 0.449 to 0.674. Figure 1(b) shows the ascending paths generated by MEM from the 11 modes (crosses) at level 2 to the 5 modes (squares) at level 3. The contour lines of the density function with the corresponding bandwidth of level 3 are plotted in the background for better illustration. The 5 clusters at level 3 are shown in Figure 1(c). These 5 modes are again merged into 3 modes at level 4, as shown in Figure 1(d).

3.3 Measures for Enhancing Speed

Because the nonparametric density estimate in (1) is a sum of kernels centered at every data point, the amount of computation to identify the mode associated with a single point grows linearly with n, the size of the data set. The computational complexity of clustering all the data by MAC is thus quadratic in n. In practice, however, it is often unnecessary to use the basic kernel estimate. A preliminary clustering can be first applied to $\{x_1,...,x_n\}$ to yield m clusters, where m is significantly smaller than n, but still much larger than the desired number of clusters. Suppose the m cluster centroids are $\overline{S} = \{\overline{x}_1,...,\overline{x}_m\}$ and the number of points in cluster $\overline{S}_j$ is n_j, $j = 1,2,...,m$. We use the density estimate

$$f(x \mid \overline{S}, D(\sigma^2)) = \sum_{j=1}^{m} \frac{n_j}{n} \phi(x \mid \overline{x}_j, D(\sigma^2))$$

in MAC to cluster the x_i's. Since MEM applies to general mixture models, the modified density function causes no essential changes to the clustering procedure.

The purpose of the preliminary clustering is more of quantizing the data than clustering. Computation is reduced by not discerning points in the same quantization region when formulating the density estimate. If m is sufficiently large, $\overline{S}$ is adequate to retain the topological structures in the nonparametric density estimate. In this fast version of MAC, we search for a mode for every $\overline{x}_i$. Examples exploiting the fast MAC are given in Section 6.

4. Analysis of Cluster Separability via Ridgelines

A measure for the separability between clusters is useful for gaining deeper understanding of clustering structures in data. With this measure, the difference between clusters is quantified, rather than being simply categorical. This quantification can be useful in certain situations. For instance, in taxonomy study, after grouping instances into species, scientists may need to numerically assess the disparity between species, often taken as an indicator for evolutionary proximity. A separability measure between the clusters of species can effectively reflect such disparity. Such a measure is also useful for diagnosing clustering results and for the mere interest of designing clustering algorithms. Based upon it, we derive a mechanism to merge weakly separated clusters. Although the separability measure is a diagnostic tool and the cluster merging method can adjust the number of clusters, in this paper, we do not pursue the problem of choosing the number of clusters fully automatically. It is well known that determining this number is a deep problem, and domain knowledge often needs to be called upon for a final decision in various applications.

The separability measure we define here exploits the geometry of the density functions of two clusters in a comprehensive manner. We only require the cluster pdf to be a mixture distribution, for example, a Gaussian kernel density estimate. Before formulating the separability measure, we

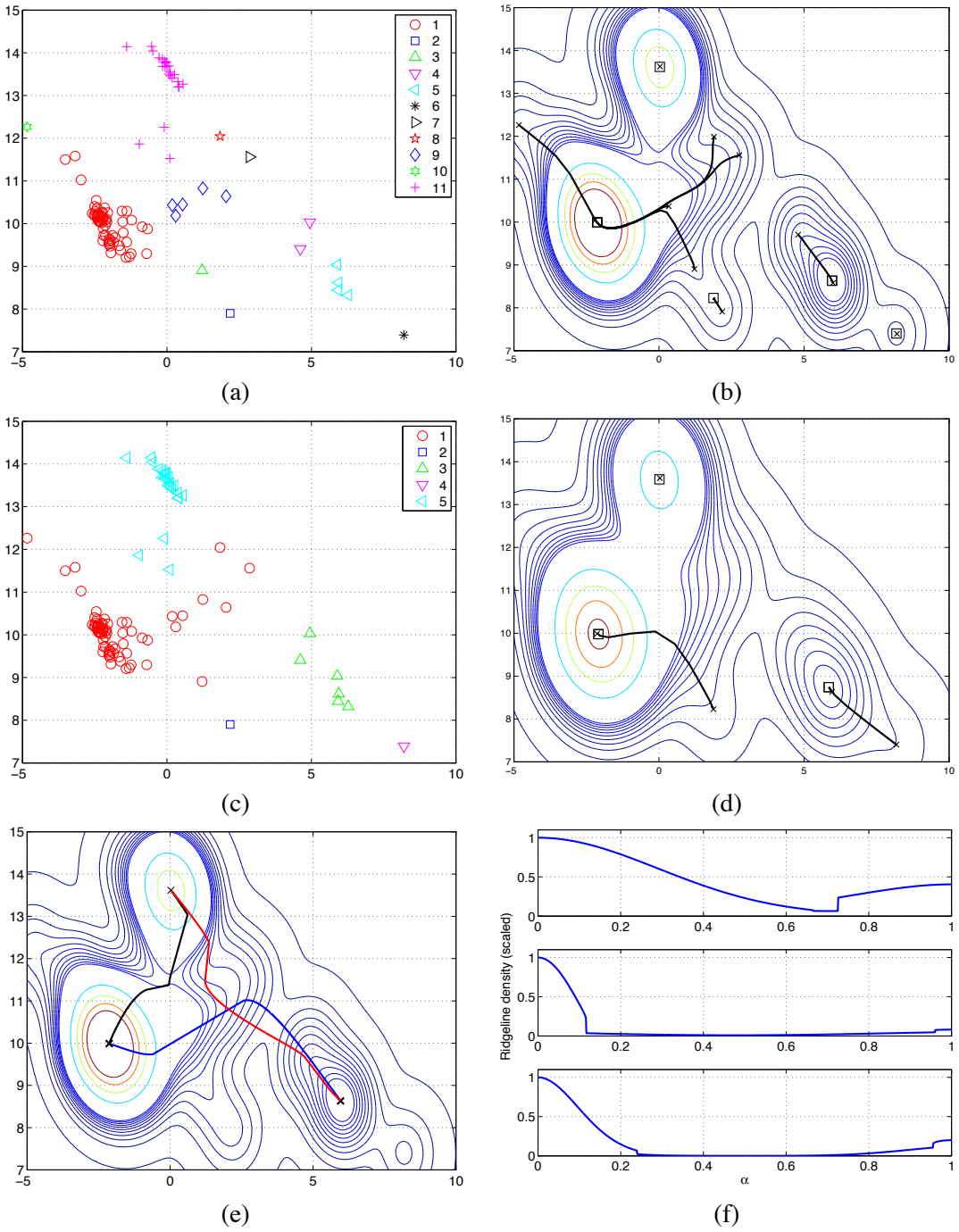


Figure 1: Clustering results for the glass data set obtained from HMAC. (a) The 11 clusters at level 2. (b) The MEM ascending paths from the modes at level 2 (crosses) to the modes at level 3 (squares), and the contours of the density estimate at level 3. (c) The 5 clusters at level 3. (d) The ascending paths from the modes at level 3 (crosses) to those at level 4 (squares) and the contours of the density estimate at level 4. (e) Ridgelines between the 3 major clusters at level 3. (f) The density function along the 3 ridgelines.

define a ridgeline between two unimodal clusters. The REM algorithm is developed to solve the ridgeline.

4.1 Ridgeline

The ridgeline between two clusters with density $g_1(x)$ and $g_2(x)$ is

$$\mathcal{L} = \{x(\alpha) : (1-\alpha)\nabla \log g_1(x) + \alpha \nabla \log g_2(x) = 0, 0 \le \alpha \le 1\} . \tag{3}$$

For a mixture density of the two clusters, $\tilde{g}(x) = \pi_1 g_1(x) + \pi_2 g_2(x)$, $\pi_1 + \pi_2 = 1$, if $\tilde{g}(x) > 0$ for any x, the modes, antimodes (local minimums), and saddle points of $\tilde{g}(x)$ must occur in $\mathcal{L}$ for any prior probability π_1. The locations of these critical points, however, depend on π_1. This fact is referred to as the *critical value theorem* and is proved by Ray and Lindsay (2005).

Remarks:

1. Eq. (3) is precisely the critical point equation for the *exponential tilt* density $g(x|\alpha) = c(\alpha) g_1(x)^{1-\alpha} g_2(x)^{\alpha}$, where $c(\alpha)$ is a normalizing constant. This density family is an exponential family, with sufficient statistic $\log(g_2(x)/g_1(x))$.

2. The set of solutions in $\mathcal{L}$ is, in general, a 1-dimensional manifold; that is, a curve. When both g_1 and g_2 are normal densities, the solution is explicit (see Ray and Lindsay, 2005), and the solutions form a unique one-dimensional curve. More generally, the solutions are possibly a set of curves that pass through the modes of the $g_1(x)$ and $g_2(x)$.

3. If both g_1 and g_2 are unimodal and have convex upper contour sets, it can be proved that the solutions form a unique curve between the modes of g_1 and g_2 respectively. In our discussion, we assume unimodal g_1 and g_2.

Since the local maxima of the exponential tilt function $g(x;\alpha)$ satisfy Eq. (3), we solve (3) by maximizing $\log(g(x|\alpha)) = (1-\alpha)\log g_1(x) + \alpha \log g_2(x)$. In the case when the two cluster densities g_1 and g_2 are themselves mixtures of basic parametric distributions, for example, normal, we develop an ascending algorithm to maximize the function, referred to as the *Ridgeline EM (REM)* Algorithm. For notational brevity, assume that both g_1 and g_2 are mixtures of T parametric distributions:

$$g_i(x) = \sum_{\kappa=1}^{T} \pi_{i,\kappa} h_{i,\kappa}(x) , \quad i = 1,2 .$$

Starting from an initial value $x^{(0)}$, REM updates x by iterating the two steps:

1. Compute

$$p_{i,\kappa} = \pi_{i,\kappa} h_{i,\kappa}(x^{(r)}) / \sum_{j=1}^{T} \pi_{i,j} h_{i,j}(x^{(r)}) , \ \kappa = 1,...,T , \ i = 1,2 .$$

2. Update $x^{(r+1)}$:

$$x^{(r+1)} = \underset{x}{\operatorname{argmax}} (1-\alpha) \sum_{\kappa=1}^{T} p_{1,\kappa} \log h_{1,\kappa}(x) + \alpha \sum_{\kappa=1}^{T} p_{2,\kappa} \log h_{2,\kappa}(x) .$$

REM ensures that $g(x^{(r+1)} \mid \alpha) \geq g(x^{(r)} \mid \alpha)$. Proof is given in Appendix B. As with MEM, we will not rigorously study the convergence properties of the sequence $\{x^{(r)}\}$. In the special case $h_{i,\kappa}(x) = \phi(x \mid \mu_{i,\kappa}, \Sigma)$, the update equation for $x^{(r+1)}$ becomes

$$x^{(r+1)} = (1-\alpha) \sum_{\kappa=1}^{T} p_{1,\kappa}\mu_{1,\kappa} + \alpha \sum_{\kappa=1}^{T} p_{2,\kappa}\mu_{2,\kappa} \, .$$

The Gaussian kernel density estimate belongs to this case.

At the two extreme values $\alpha = 0, 1$, the solutions are the modes of $g_1(x)$ and $g_2(x)$ respectively. We solve $x(\alpha)$ sequentially on a set of grid points $0 = \alpha_0 < \alpha_1 < \cdots < \alpha_\zeta = 1$. First, $x(0) = \text{argmax}_x g_1(x)$ is solved by MEM. For every α_l, $x(\alpha_{l-1})$, previously calculated, is used as initial value to start the iterations in REM.

Suppose two clusters, denoted by z_1 and z_2, have densities g_1 and g_2, and prior probabilities π_1 and π_2 respectively. We define a pairwise *separability* as

$$S(z_1, z_2) = 1 - \frac{\min_\alpha \pi_1 g_1(x(\alpha)) + \pi_2 g_2(x(\alpha))}{\pi_1 g_1(x(0)) + \pi_2 g_2(x(0))} = 1 - \frac{\pi_1 g_1(x(\alpha^*)) + \pi_2 g_2(x(\alpha^*))}{\pi_1 g_1(x(0)) + \pi_2 g_2(x(0))} \tag{4}$$

where $\alpha^* = \text{argmin}_\alpha \pi_1 g_1(x(\alpha)) + \pi_2 g_2(x(\alpha))$. Usually, the prior probabilities π_1 or π_2 are proportional to the cluster sizes. It is obvious that $S(z_1, z_2) \in [0, 1]$. To symmetrize the measure, we define the pairwise *symmetric separability* as $\tilde{S}(z_1, z_2) = \min[S(z_1, z_2), S(z_2, z_1)]$.

By finding $x(\alpha^*)$, we can evaluate the amount of "dip" along the ridgeline. By the critical value theorem, the minimum of $\pi_1 g_1(x) + \pi_2 g_2(x)$, if exists, lies on the ridgeline and therefore must be $x(\alpha^*)$. Hence, if there is a "dip" in the mixture of the two clusters, it will be captured by the ridgeline. According to definition (4), a deeper "dip" leads to higher separability. In our implementation, we approximate α^* by

$$\alpha^* \approx \underset{\alpha_l, 0 \leq l \leq \zeta}{\text{argmin}} \, \pi_1 g_1(x(\alpha_l)) + \pi_2 g_2(x(\alpha_l)) \, ,$$

where $\alpha_l, 0 \leq l \leq \zeta$, are the grid points.

We also define the separability for a single cluster to quantify its overall distinctness from other clusters. Specifically, suppose there are m clusters denoted by z_i, $i = 1, ..., m$. The separability of cluster z_i, denoted by $s(z_i)$ is defined by

$$s(z_i) = \min_{j: 1 \leq j \leq m, j \neq i} S(z_i, z_j) \, .$$

We call a cluster "insignificant" if its separability is below a given threshold ε. In our discussion, $\varepsilon = 0.5$.

Take the glass data set as an example. As shown in Figure 1(c), the points are divided into 5 groups at that level of the clustering hierarchy. Two of the clusters each contain a single sample point, which is far from the other points and forms a mode alone. The separability of these two clusters are weak, respectively 0.00 and 0.30. The three other clusters are highly separable, with separability values 0.94, 0.84 and 0.81. Figure 1(e) shows the ridgelines between any two of the three significant clusters, and Figure 1(f) shows the density function along the ridgelines, normalized to one at the ridgeline end point $x(0)$ or $x(1)$ (depending on whichever is larger).

The task of identifying insignificant clusters in Figure 1(c) is not particularly challenging because the two smallest clusters are "singletons" (containing only one sample). However, cluster size

is not the sole factor affecting separability. Take the clusters in Figure 1(a) as an example. The separability of the 11 clusters and their corresponding sizes (number of points contained) are listed in Table 1. It is shown that although cluster 9 is the third largest cluster, its separability is low. At threshold $\varepsilon = 0.5$, this cluster is insignificant. In contrast, by being far from all the other clusters, cluster 6, a singleton, has separability 0.87. The low separability of cluster 9 is caused by its proximity to cluster 1, the largest cluster which accounts for 60% of the data. Clusters that contain a large portion of data tend to "absorb" surrounding smaller clusters. The attraction of a small cluster to a bigger one depends on its relative size, tightness, distance to the bigger cluster, as well as the orientation of the data masses in the two clusters.

Cluster	1	2	3	4	5	6	7	8	9	10	11
Size	63	1	1	2	4	1	1	1	5	1	25
Separability	0.94	0.41	0.41	0.31	0.68	0.87	0.13	0.13	0.18	0.46	0.92

Table 1: Separability between clusters in the glass data set

4.2 Merging Clusters Based on Separability

To elaborate on the relationships between clusters, we compute the matrix of separability between any pair of clusters. We can potentially use this matrix to decide which clusters can be merged due to weak separation. As discussed previously, one way to merge clusters is to increase the bandwidth of the kernel function used by HMAC. However, an enlarged bandwidth may cause prominent clusters to be clumped while leaving undesirable small "noisy" clusters unchanged. Merging clusters according to the separability measure is one possible approach to eliminate "noisy" clusters without losing important clustering structures found at a small bandwidth. We will show by the glass data that the merging method makes clustering results less sensitive to bandwidth.

Let the clusters in consideration be $\{z_1, z_2, ..., z_m\}$. We denote the pairwise separability between cluster z_i and z_j in short by $S_{i,j}$, where $S_{i,j} = S(z_i, z_j)$. Note in general $S_{i,j} \neq S_{j,i}$. Let a threshold for separability be ε, $0 < \varepsilon < 1$. Let the density function of cluster z_i be $g_i(\cdot)$ and the prior probability be π_i. Denote the weighted mode of each cluster density function by $\delta(z_i) = \pi_i \max g_i(x)$. Since in MEM the mode $\max g_i(x)$ for each cluster z_i is computed when z_i is formed, $\delta(z_i)$ requires no extra computation after clustering. We refer to $\delta(z_i)$ as the *significance index* of cluster z_i.

The main idea of the merging algorithm is to have clusters with a higher significance index absorb other clusters that are not well separated from them and are less dominant (lower significance index). Several issues need to be resolved to avoid arbitrariness in merging. First, a cluster z_i may be weakly separated from several other clusters with higher significance indices. Among those clusters, we let the one from which z_i is worst separated to absorb z_i. Second, two weakly separated clusters z_i and z_j may have the same significance indices, that is, $\delta(z_i) = \delta(z_j)$; and hence it is ambiguous which cluster should be treated as the dominant one. We solve this problem by introducing the concept of *cliques*. The clusters are first grouped into cliques which contain weakly separated z_i's with the same value of $\delta(z_i)$. Clusters in different cliques are ensured to be either well separated or have different significance indices. We then apply the absorbing process to the cliques, without the possibility of encountering the aforementioned ambiguity.

Next, we describe how to form cliques and extend the definition of pairwise separability to cliques. In order to carry out the merging of cliques, a directed graph is constructed upon the cliques based on pairwise separability and the comparison of significance indices.

1. *Tied*: Cluster z_i and z_j are defined to be *tied* if $S(z_i, z_j) < \varepsilon$ and $\delta(z_i) = \delta(z_j)$.

2. *Clique*: Cluster z_i and z_j are in the same *clique* if (a) z_i and z_j are tied, or (b) there is a cluster z_k such that z_i and z_k are in the same clique, and z_j and z_k are in the same clique.

   Remark: the relationship "tied" results in a partition of z_i's. Each group formed by the partition is a clique. Because being tied requires $\delta(z_i) = \delta(z_j)$, in practice, we only observe clusters being tied when they are all singletons.

We now define the *separability between cliques*. Without loss of generality, let clique $c_1 = \{z_1, z_2, ..., z_{m_1}\}$ and $c_2 = \{z_{m_1+1}, ..., z_{m_1+m_2}\}$. Denote the clique-wise separability by $S_c(c_1, c_2)$:

$$S_c(c_1, c_2) \triangleq \min_{1 \le i \le m_1} \min_{m_1+1 \le j \le m_1+m_2} S(z_i, z_j) \, .$$

Since in general, $S(z_i, z_j) \neq S(z_j, z_i)$, the asymmetry carries over to $S_c(c_1, c_2) \neq S_c(c_2, c_1)$. We also denote the significance index of a clique c_i as $\delta(c_i)$. Since all the clusters in c_i have equal significance indices, we let $\delta(c_i) = \delta(z_{i'})$, where $z_{i'}$ is any cluster included in c_i.

Regard each clique as a node in a graph. Suppose there are $\tilde{m}$ cliques $\{c_1, ..., c_{\tilde{m}}\}$. A directed graph is built as follows. For two arbitrary cliques c_i and c_j, a link from c_i to c_j is made if

1. $S_c(c_i, c_j) < \varepsilon$.

2. $S_c(c_i, c_j) = \min_{k \neq i} S_c(c_i, c_k)$ and j is the smallest index among all those j''s that achieve $S_c(c_i, c_{j'}) = \min_{k \neq i} S_c(c_i, c_k)$.

3. $\delta(c_i) < \delta(c_j)$.

A clique c_i is said to be *linked* to c_j if there is a directed edge from c_i to c_j. It is obvious by the second requirement in the link construction that every clique is linked to at most another clique. In Appendix C, it is proved that a graph built by the above rules has no loops. An example graph is illustrated in Figure 2(b). If we disregard the directions of the links, the graph is partitioned into connected subgraphs. In the given example in Figure 2(b), there are four connected subgraphs. The basic idea of the merging algorithm is to let the most dominant clique in one subgraph absorb all the others in the same subgraph.

We call clique c_j the *parent clique* of c_i if there is a directed link from c_i to c_j. In this case, we also say c_i is *directly absorbed* by c_j. By construction, $\delta(c_i) < \delta(c_j)$. More generally, if there is a directed path from c_i to c_j, then c_j is called an *ancestor* of c_i, and c_i is *absorbed* by c_j. Again, we have $\delta(c_i) < \delta(c_j)$ by transitivity. In each connected subgraph containing k nodes, because there is no loop, there are exactly $k-1$ links. Since every node has at most one link originating from it, the $k-1$ links have to originate from $k-1$ different nodes. Therefore, there is precisely 1 node in each connected subgraph that has no link originating from it. This node is called the *head* node (clique) of the connected nodes. It is not difficult to see that the head node is an ancestor for all the other nodes in the subgraph. As a result, the significance index of the head clique is strictly larger

than that of any other clique in the subgraph. In this sense, the head clique dominates all the other connected cliques.

Combining clusters by the above method is the first and main step in our merging algorithm. To account for outliers, the second step in the algorithm employs the notation of *coverage rate*. Outlier points far from all the essential clusters tend to yield high separability and hence will not be merged. In HMAC, to "grab" those outliers, the kernel bandwidth needs to grow so large that under such a bandwidth, many significant clusters are undesirably merged. To address this issue, we find the smallest clusters and mark them as outliers if their total size proportional to the entire data set is below a threshold. For instance, if the *coverage rate* allowed is 95%, this threshold is then 5%.

We now summarize our cluster merging algorithm as follows. We call the merging procedure conducted based on the separability measure stage I merging and that based on coverage rate stage II merging. The two stages do not always have to be concatenated. We can apply each alone. Applying only stage I is equivalent to applying two stages and setting the coverage rate to 100%; applying only stage II is equivalent to setting the threshold of separability to 0.0. Assume the starting clusters are $\{z_1, z_2, ..., z_m\}$.

1. Stage I: merging based on separability.
   (a) Compute the separability matrix $[S_{i,j}]$, $i, j = 1, ..., m$, and the significant index $\delta(z_i)$, $i = 1, ..., m$.
   (b) Form cliques $\{c_1, c_2, ..., c_{\tilde{m}}\}$ based on $[S_{i,j}]$ and $\delta(z_i)$'s, where $\tilde{m} \leq m$. Record the z_i's contained in each clique.
   (c) Construct the directed graph.
   (d) Merge cliques that are in the same connected subgraph. z_i's contained in merged cliques are grouped into one cluster. Denote those merged clusters by $\{\hat{z}_1, \hat{z}_2, ..., \hat{z}_{\hat{m}}\}$, where $\hat{m} \leq \tilde{m}$.
2. Stage II: merging based on coverage rate. Denote the coverage rate by ρ.
   (a) Calculate the sizes of clusters $\{\hat{z}_1, \hat{z}_2, ..., \hat{z}_{\hat{m}}\}$ and denote them by $\hat{n}_i$, $i = 1, ..., \hat{m}$. The size of the whole data set is $n = \sum_{i=1}^{\hat{m}} \hat{n}_i$.
   (b) Sort $\hat{n}_i$, $i = 1, ..., \hat{m}$, in ascending order. Let the sorted sequence be $\hat{n}_{(1)}$, $\hat{n}_{(2)}$, ..., $\hat{n}_{(\hat{m})}$. Let $\hat{n}_{(0)} = 0$. Let k be the largest integer such that $\sum_{i=0}^{k} \hat{n}_{(i)} \leq (1-\rho)n$.
   (c) If $k > 0$, go to the next step. Otherwise, stop and the final clusters are $\{\hat{z}_1, \hat{z}_2, ..., \hat{z}_{\hat{m}}\}$.
   (d) For each $\hat{z}_{(i)}, i = 1, ..., k$, find all the original clusters z_j's that are merged into $\hat{z}_{(i)}$. Denote the index set of the z_j's by $H_{(i)}$. Let $H' = \cup_{i=1}^{k} H_{(i)}$ and $H'' = \cup_{i=k+1}^{\hat{m}} H_{(i)}$.
   (e) For each $\hat{z}_{(i)}, i = 1, ..., k$, find $j^* = \text{argmin}_{j \in H''} \min_{l \in H_{(i)}} S(z_l, z_j)$. Find the cluster $\hat{z}_{j'}$ that contains z_{j^*}. Merge $\hat{z}_{(i)}$ with $\hat{z}_{j'}$. The new clusters obtained are $\{\overline{z}_1, \overline{z}_2, ..., \overline{z}_{\overline{m}}\}$, where $\overline{m} = \hat{m} - k < \hat{m}$.
   (f) Reset $\overline{m} \to \hat{m}$ and $\overline{z}_i \to \hat{z}_i$, $i = 1, ..., \overline{m}$. Go back to step (a).

Unless there is a definite need to assign every data point to one of the major clusters, in certain applications, it may be more preferable to keep the outlier status of some points rather than allocating them to distant clusters.

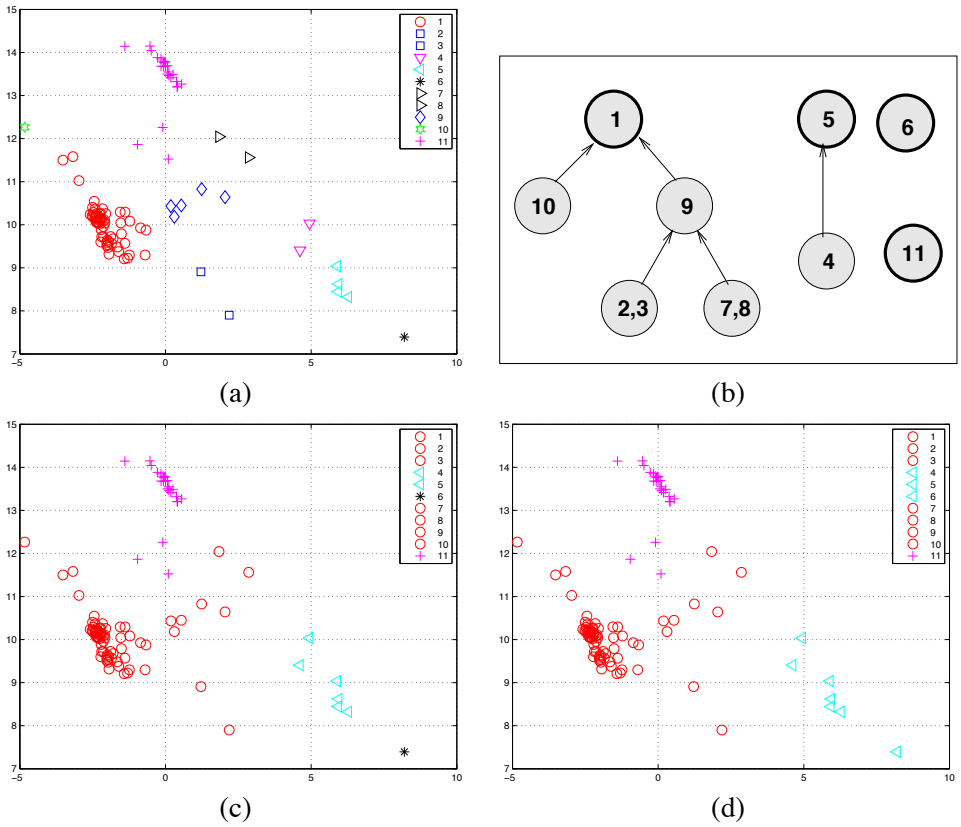


Figure 2: The process of merging clusters for the glass data set. (a) The clustering result after merging clusters in the same cliques. (b) The cliques and directed graph constructed based on separability. (c), (d) The clustering results after stage I and stage II merging respectively.

The first stage merging based on separability is intrinsically connected with linkage-based agglomerative clustering. For details on linkage clustering, see Jain et al. (1999). In a nutshell, linkage clustering forms clusters by progressively merging a pair of current clusters. Initially, every data point is a cluster. The two clusters with the minimum pairwise distance are chosen to merge at each step. The procedure is greedy in nature since minimization is conducted sequentially through every merge. Linkage clustering methods differ by the way between-cluster distance is updated when a new cluster is combined from two smaller ones. For instance, in single linkage, if cluster ξ_2 and ξ_3 are merged into ξ_4, the distance between ξ_1 and ξ_4 is calculated as $d(\xi_1,\xi_4) = \min(d(\xi_1,\xi_2),d(\xi_1,\xi_3))$. If complete linkage, the distance becomes $d(\xi_1,\xi_4) = \max(d(\xi_1,\xi_2),d(\xi_1,\xi_3))$.

Our merging algorithm is a particular kind of linkage clustering where the elements to be clustered are cliques and the distance between the cliques is the separability. The update of this distance for merged groups of cliques is however different from commonly used versions in linkage clustering. The update is the same as single linkage under a certain scenario, but differs in general because of the directed links and the notion of head cliques. We may call this linkage clustering

algorithm *directional single linkage* for reasons that will be self-evident shortly. Consider the above example where ξ_2 and ξ_3 merge into ξ_4. We call ξ more dominant than ξ' if the head clique in ξ has a higher significance index than that in ξ', and denote $\delta(\xi) > \delta(\xi')$. Without loss of generality, assume $\delta(\xi_2) > \delta(\xi_3)$. Then the update of $d(\xi_1, \xi_4)$ and the ordering of $\delta(\xi_1)$ and $\delta(\xi_4)$ (needed for updating the distance) follows three cases:

$$\begin{cases} d(\xi_1,\xi_4) = d(\xi_1,\xi_2), \text{ and } \delta(\xi_1) > \delta(\xi_4), & \text{if } \delta(\xi_1) > \delta(\xi_2) \\ d(\xi_1,\xi_4) = d(\xi_1,\xi_2), \text{ and } \delta(\xi_1) < \delta(\xi_4), & \text{if } \delta(\xi_3) < \delta(\xi_1) < \delta(\xi_2) \\ d(\xi_1,\xi_4) = \min(d(\xi_1,\xi_2), d(\xi_1,\xi_3)), \text{ and } \delta(\xi_1) < \delta(\xi_4), & \text{if } \delta(\xi_1) < \delta(\xi_3) \ . \end{cases}$$

In our proposed merging procedure, we essentially employ a threshold to stop merging when all the between-cluster distances exceed this value. An alternative is to apply the directional single linkage clustering and stop merging when a desired number of clusters is achieved.

We use the glass data set in the previous section to illustrate the merging algorithm. The threshold for separability is set to $\varepsilon = 0.5$ and the coverage rate is $\rho = 95\%$. Apply the algorithm to the 11 clusters formed at the 2nd level of the hierarchical clustering, shown in Figure 1(a). We refer to the clusters as z_1, ..., z_{11}, where the label assignment follows the indication in the figure. The 11 clusters form 9 cliques. Figure 2(a) shows the clustering after merging clusters in the same clique. The square (triangle) symbol used for z_2 (z_7) is now used for both z_2 (z_7) and z_3 (z_8) to indicate that they have been merged. To highlight the relationship between the merged clusters and the original clusters, the list of updated symbols for each original cluster is given in every scatter plot. Figure 2(b) demonstrates the directed graph constructed for the cliques. The z_i's contained in each clique are indicated in the node. Figure 2(c) shows the clustering result after stage I merging. The symbol of the head clique in each connected subgraph is adopted for all the clusters it absorbs. The 4 clusters generated at stage I contain 73, 25, 6, 1 points respectively. At $\rho = 95\%$, only the cluster of size 1 is marked as an outlier cluster, and is merged with the cluster of size 6.

Because clusters with low separability are apt to be grouped together when the kernel bandwidth increases, it is not surprising for the clustering result obtained by the merging algorithm to agree with clustering at a higher level of HMAC. On the other hand, examining separability and identifying outlier clusters enhance the robustness of clustering results, a valuable trait especially in high dimensions. Examples will be shown in Section 6. For practical interest, when equipped with the merging algorithm, we do not need to generate all the hierarchical levels in HMAC until reaching a targeted number of clusters. Instead, we can apply the merging algorithm to a relatively large number of clusters obtained at an early level and reduce the number of clusters to the desired value.

5. Visualization

For clarity, we have used 2-D data to illustrate our new clustering methods, although these methods are not limited by data dimensions. Projection into lower dimensions is needed to visualize clustering results for higher dimensional data. PCA (principal component analysis) is a widely used linear projection method, but it is not designed to reveal clustering structures. We will describe in this section a linear projection method that aims at effectively showing clustering structures. The method is employed in our experiments. We note that visualization is a rich research area in its own right. However, because this topic is beyond the focus of the current paper, we will not discuss it in great depth, nor make thorough comparisons with other methods.

Modal clustering provides us an estimated density function and a prior probability for each cluster. Suppose K clusters are generated. Let the cluster density function of x, $x \in \mathcal{R}^d$, be $g_k(x)$, and the prior probability be $\pi_k, k = 1,2,...,K$. For any $x \in \mathcal{R}^d$, its extent of association with each cluster k is indicated by the posterior probability $p_k(x) \propto \pi_k g_k(x)$. To determine the posterior probabilities $p_k(x)$, under a given set of priors, it suffices to specify the discriminant functions $\log \frac{g_1(x)}{g_K(x)}$, ..., $\log \frac{g_{K-1}(x)}{g_K(x)}$. Without loss of generality, we use $g_K(x)$ as the basis for computing the ratios. Our projection method attempts to find a plane such that $\log \frac{g_k(x)}{g_K(x)}, k = 1,...,K-1$ can be well approximated if only the projection of data into the plane is specified. By preserving the discriminant functions, the posterior probabilities of clusters will remain accurate.

Let the data set be $\{x_1, x_2, ..., x_n\}$, $x_i \in \mathcal{R}^d$. Denote a particular dimension of the data set by $x_{\cdot,l} = (x_{1,l}, x_{2,l}, ..., x_{n,l})^t, l = 1,...,d$. For each $k, k = 1,...,K-1$, the pairs $(x_i, \log \frac{g_k(x_i)}{g_K(x_i)}), i = 1,...,n$, are computed. Let $y_{i,k} = \log \frac{g_i(x_i)}{g_K(x_i)}$. Linear regression is performed based on the pairs $(x_i, y_{i,k}), i = 1,...,n$, to acquire a linear approximation for each discriminant function. Let $\beta_{k,0}, \beta_{k,1}, \beta_{k,2}, ..., \beta_{k,d}$ be the regression coefficients for the kth discriminant function. Denote $\beta_k = (\beta_{k,1}, \beta_{k,2}, ..., \beta_{k,d})^t$ and the fitted values for $\log \frac{g_k(x_i)}{g_K(x_i)}$ by $\hat{y}_{i,k} = \beta_{k,0} + \beta_k^t x_i$. Also denote $\tilde{y}_{i,k} = \beta_k^t x_i = \hat{y}_{i,k} - \beta_{k,0}$. For mathematical tractability, we convert the approximation of the discriminant functions to the approximation of their linearly regressed values $(\hat{y}_{i,1}, \hat{y}_{i,2}, ..., \hat{y}_{i,K-1}), i = 1,...,n$, which is equivalent to approximate $(\tilde{y}_{i,1}, \tilde{y}_{i,2}, ..., \tilde{y}_{i,K-1})$ since the two only differ by a constant. To precisely specify $(\tilde{y}_{i,1}, \tilde{y}_{i,2}, ..., \tilde{y}_{i,K-1})$, we need the projection of x_i onto the $K-1$ directions, $\beta_1, \beta_2, ..., \beta_{K-1}$. If we are restricted to showing the data in a plane and $K-1 > 2$, further projection of $(\tilde{y}_{i,1}, \tilde{y}_{i,2}, ..., \tilde{y}_{i,K-1})$ is needed. At this stage, we employ PCA on the vectors $(\tilde{y}_{i,1}, \tilde{y}_{i,2}, ..., \tilde{y}_{i,K-1})$ (referred to as the discriminant vectors), $i = 1,...,n$, to yield a two-dimensional projection. Suppose the two principal component directions for the discriminant vectors are $\gamma_j = (\gamma_{j,1}, ..., \gamma_{j,K-1})^t$, $j = 1,2$. The two principal components v_j, $j = 1,2$, are

$$\begin{pmatrix} v_{1,j} \\ v_{2,j} \\ \vdots \\ v_{n,j} \end{pmatrix} = \gamma_{j,1} \begin{pmatrix} \tilde{y}_{1,1} \\ \tilde{y}_{2,1} \\ \vdots \\ \tilde{y}_{n,1} \end{pmatrix} + \cdots + \gamma_{j,K-1} \begin{pmatrix} \tilde{y}_{1,K-1} \\ \tilde{y}_{2,K-1} \\ \vdots \\ \tilde{y}_{n,K-1} \end{pmatrix} = \sum_{l=1}^{d} \left[\sum_{k=1}^{K-1} \gamma_{j,k} \beta_{k,l} \right] x_{\cdot,l} \, .$$

To summarize, the two projection directions for x_i are

$$\tau_j = \left(\sum_{k=1}^{K-1} \gamma_{j,k} \beta_{k,1}, \sum_{k=1}^{K-1} \gamma_{j,k} \beta_{k,2}, ..., \sum_{k=1}^{K-1} \gamma_{j,k} \beta_{k,d} \right)^t, \quad j = 1,2 \, . \tag{5}$$

In practice, it may be unnecessary to preserve all the $K-1$ discriminant functions. We can apply the above method to a subset of discriminant functions corresponding to major clusters. The two projection directions in (5) are not guaranteed to be orthogonal, but it is easy to find two orthonormal directions spanning the same plane.

If we use a basis function other than $g_K(x)$, say $g_{k'}(x)$, to form the discriminant functions, the new set of vectors β_k's will span the same linear space as the β_k's obtained with $g_K(x)$. The reason is that the new discriminant functions $\log \frac{g_k(x_i)}{g_{k'}(x_i)}, 1 \le k \le K, k \ne k'$, can be linearly transformed from the previously defined $(y_{i,1}, y_{i,2}, ..., y_{i,K-1})$ by $\log \frac{g_k(x_i)}{g_{k'}(x_i)} = y_{i,k} - y_{i,k'}$, for $k \ne k', K$, $\log \frac{g_K(x_i)}{g_{k'}(x_i)} = -y_{i,k'}$, and linear regression is used on the $y_{i,k}$'s. On the other hand, as the linear transform is not orthonormal, the PCA result is not invariant under the transform and the projection directions τ_j can change.

6. Experiments

We present in this section experimental results of the proposed modal clustering methods on simulated and real data. We also discuss measures for enhancing computational efficiency and describe the application to image segmentation, which may involve clustering millions of data points.

6.1 Simulated Data

We experiment with three simulated examples to illustrate the effectiveness of modal clustering. We start by comparing linkage clustering with mixture modeling using two data sets. This will allow us to illustrate the strengths and weaknesses of these two methods and therefore better demonstrate the tendency of modal clustering to combine their strengths. We then present a study to assess the stability of HMAC over multiple implementations and its performance under increasing dimensions. In this study, comparisons are made with the `Mclust` function in `R`, a state-of-the-art mixture-model-based clustering tool (Fraley and Raftery, 2002, 2006).

For our two data sets, single linkage, complete linkage, and average linkage yield similar results. For brevity, we only show results of average linkage. In average linkage, if cluster z_2 and z_3 are merged into z_4, the distance between z_1 and z_4 is calculated as $d(z_1,z_4) = \frac{n_2}{n_2+n_3} d(z_1,z_2) + \frac{n_3}{n_2+n_3} d(z_1,z_3)$, where n_2 and n_3 are the cluster sizes of z_2 and z_3 respectively. Details on clustering by mixture modeling are referred to Section 1.1. We will also show results of k-means clustering.

The first data set, referred to as the noisy curve data set, contains a half circle and a straight line (or bar) imposed with noise, as shown in Figure 3(a). The circle centers at the origin and has radius 7. The line is a vertical segment between $(13,-8)$ and $(13,0)$. Roughly $\frac{2}{3}$ of the 200 points are uniformly sampled from the half circle and $\frac{1}{3}$ of them uniformly from the bar. Then, independent Gaussian noise with standard deviation 0.5 is added to both the horizontal and vertical directions of each point.

Consider clustering into two groups. The results of average linkage, mixture modeling, and k-means are shown in Figure 3(a), (b), (c). For this example, average linkage partitions the data perfectly into a noisy half circle and bar. Results of mixture modeling and k-means are close. In both cases, nearly one side of the half circle is grouped with the bar. In this example, the mixture model is initialized by the clustering obtained from k-means; and the covariance matrices of the two clusters are not restricted.

The second data set contains 200 samples generated as follows. The data are sampled from two clusters with prior probability $1/3$ and $2/3$ respectively. The first cluster follows a single Gaussian distribution with mean $(6,0)$ and covariance matrix $diag(1.5^2,1.5^2)$. The second cluster is generated by a mixture of two Gaussian components with prior probability $1/5$ and $4/5$, means $(-3,0)$ and $(0,0)$, and covariance matrices $diag(3^2,3^2)$ and $diag(1,1)$ respectively. The two clusters are shown in Figure 4(a). Again, we compare results of average linkage, mixture modeling, and k-means, shown in Figure 4(b), (c), (d). For mixture modeling, we use Mclust with three components and optimally selected covariance structures by BIC. Two of the three clusters generated by Mclust are combined to show the binary grouping. For this example, mixture modeling and k-means yield a partition close to the two original clusters, while average linkage gives highly skewed clusters, one of which contains a very small number of points on the outskirt of the mass of data.

We apply HMAC to both data sets and show the clustering results obtained at the level where two clusters are formed. For the noisy curve data set, HMAC perfectly separates the noisy half circle and the bar, as shown in Figure 3(a). For the second example, shown in Figure 4(e), HMAC yields

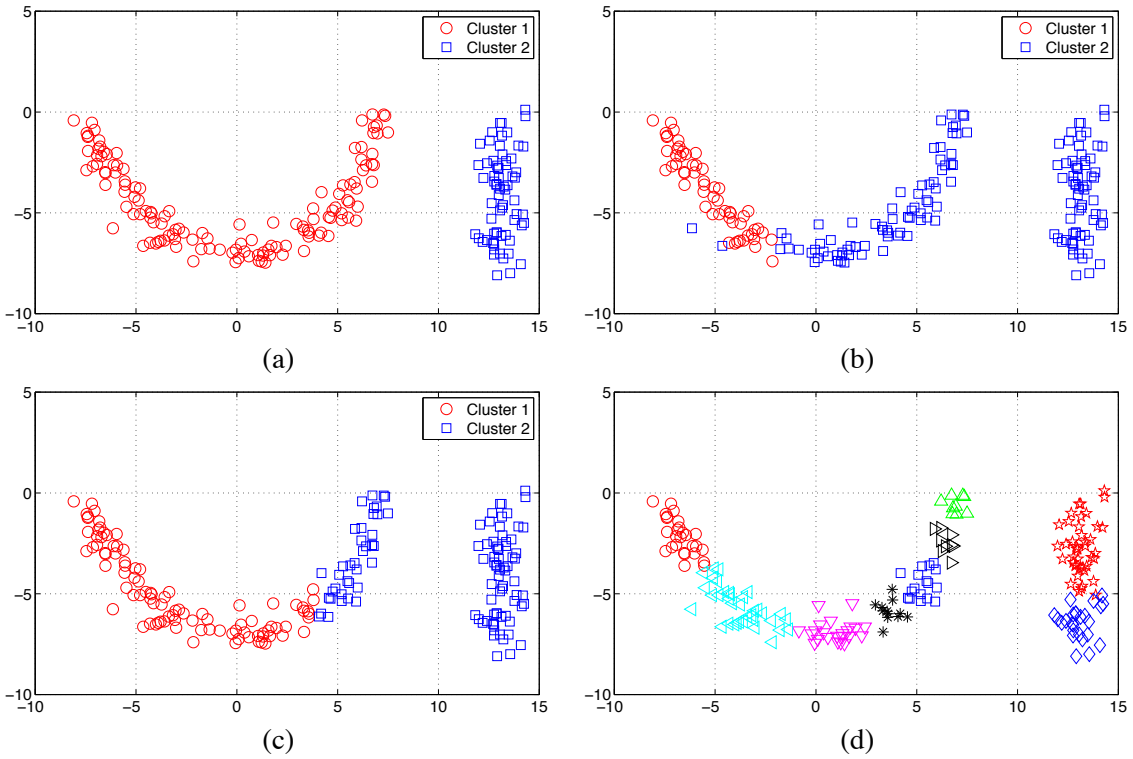


Figure 3: Clustering results for the noisy curve data set. (a) The two original clusters. The two clusters obtained by average linkage clustering and HMAC are identical to the original ones. (b) Clustering by mixture modeling with two Gaussian components. (c) Clustering by k-means. (d) Clustering by HMAC at the first level of the hierarchy.

clusters closest to the original ones among all the methods. Comparing with mixture modeling, HMAC is more robust to the non-Gaussian cluster distributions.

The above two data sets exemplify situations in which either the average linkage clustering or mixture modeling (or k-means) performs well but not both. In the first data set, the two clusters are well separated but seriously violate the assumption of Gaussian distributions. By greedy pairwise merging, average linkage successfully divides the two clusters. In contrast, both mixture modeling and k-means attempt to optimize an overall clustering criterion. Mixture modeling favors elliptical clusters because of the Gaussian assumption, and k-means favors spherical clusters due to extra restrictions on Gaussian components. As a result, one side of the noisy half circle is grouped with the bar to achieve better fitting of Gaussian distributions. On the other hand, mixture modeling and k-means perform significantly better than average linkage in the second example. The greedy pairwise merging in average linkage becomes rather arbitrary when clusters are not well separated.

HMAC demonstrates a blend of traits from average linkage and mixture modeling. When the kernel bandwidth is small, the cluster to which a point is assigned is largely affected by its neighbors. Points close to each other tend to be grouped together, as shown by Figure 3(d) and Figure 4(f). This strong inclination of putting neighboring points in the same cluster is also a feature of average linkage. However, a difference between HMAC and average linkage is that decisions to merge in

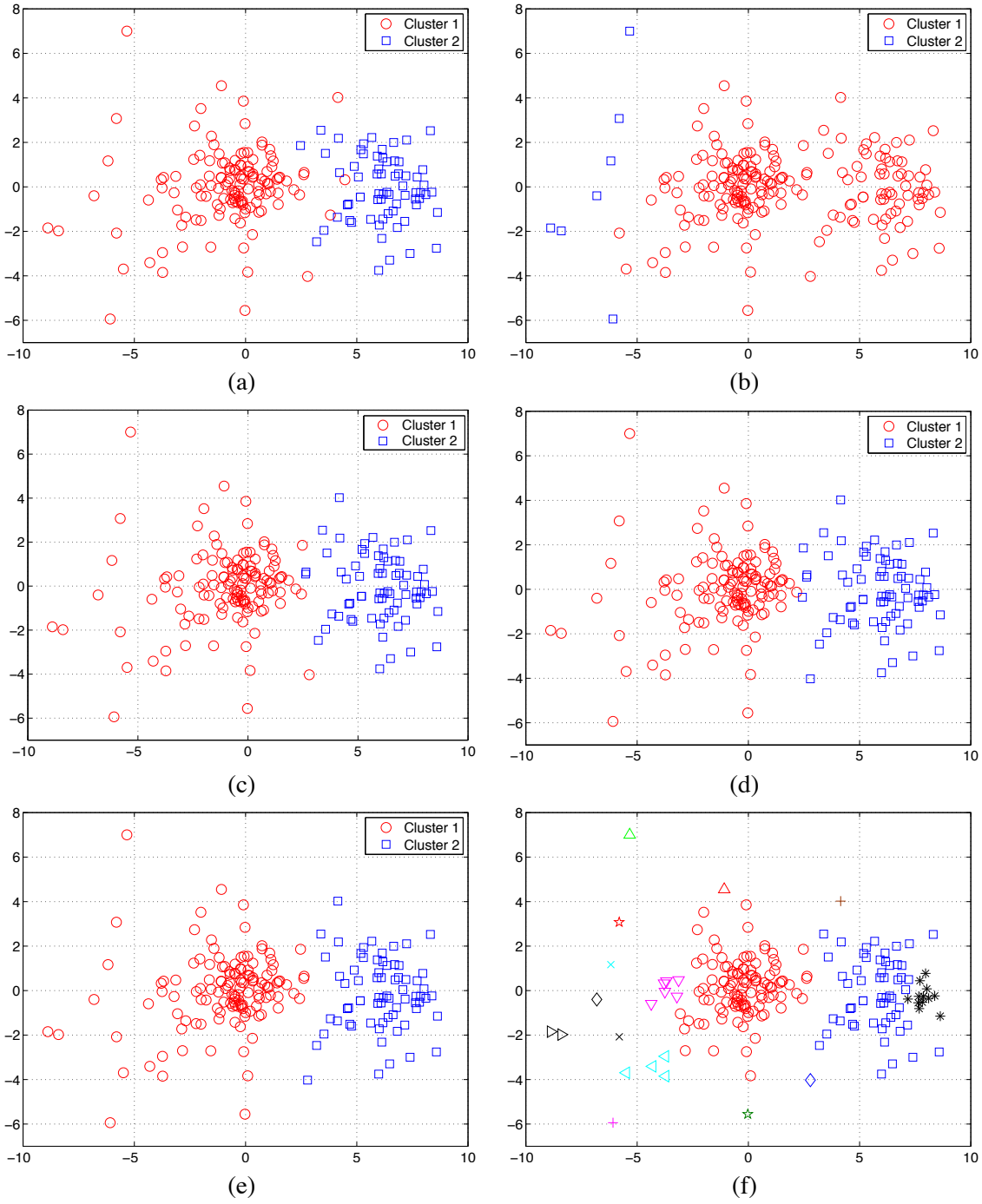


Figure 4: Clustering results for the second simulated data set. (a) The original two clusters. (b) Clustering by average linkage. (c) Clustering by mixture modeling using Mclust with three Gaussian components. Two clusters are merged to show the binary grouping. (d) Clustering by k-means. (e) Clustering by HMAC. (f) Clustering by HMAC at the first level of the hierarchy.

the latter are always local. While in HMAC, when the bandwidth increases, global characteristics of the data become highly influential on clustering results. Hence the clustering result tends to resemble that of mixture modeling (k-means) which enforces a certain kind of global optimality. In spite of this similarity, HMAC is more robust for clusters deviating from Gaussian distributions. In practice, it is usually preferable to ensure very close points are grouped together and in the mean time to generate clusters with global optimality. These two traits, however, are often at odds with each other, a phenomenon discussed in depth by Chipman and Tibshirani (2006).

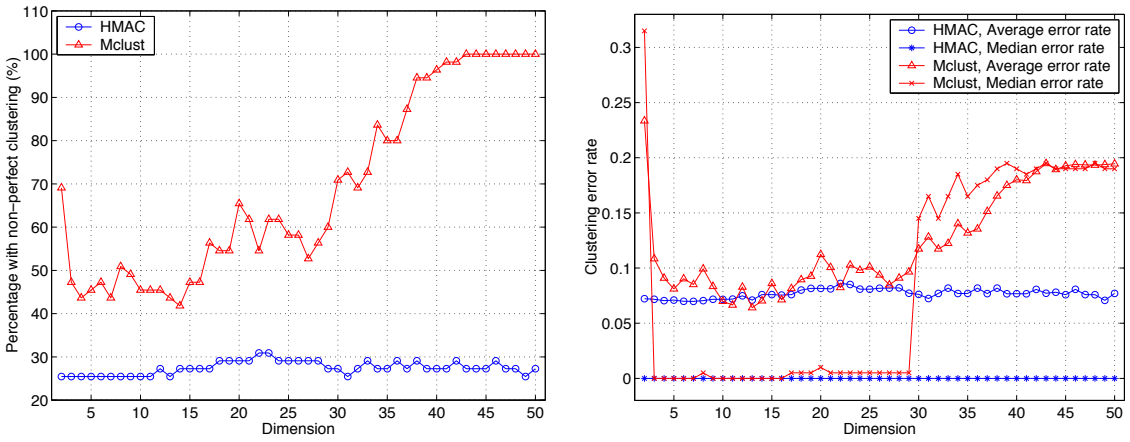
Chipman and Tibshirani (2006) noted that bottom-up agglomerative clustering methods, such as average linkage, tend to generate good small clusters but suffer at extracting a few large clusters. The strengths and weaknesses of top-down clustering methods, such as k-means, are the opposite. A hybrid approach is proposed in that paper to combine the advantages of bottom-up and top-down clustering, which first seeks mutual clusters by bottom-up linkage clustering and then applies top-down clustering to the mutual clusters. HMAC also integrates the strengths of both types of clustering, in a way not as explicit as the hybrid method but more automatically.

To systematically study the performance of HMAC for high dimensional data and its stability over multiple implementations, we conduct the following experiment. We generate 55 random data sets each of dimension 50 and size 200. The first two dimensions of the data follow the distribution of the noisy curve data in the first example described. The other 48 dimensions are independent Gaussian noise all following the normal distribution with mean zero and standard deviation 0.5, same as the noise added to the half circle and line segment in the first two dimensions. As gold standard, we regard data generated by adding noise to the half circle as one cluster and those to the line segment as the other.

Clustering results are obtained for each of the 55 data sets using HMAC and Mclust respectively. For HMAC, the level of the dendrogram yielding two clusters is chosen to create the partition of the data. In another word, the basic version of HMAC is used without the separability based merging of clusters. For Mclust, we set the number of clusters to 2, but allow the algorithm to optimally select the structure of the covariance matrices using BIC. All the structural variations of the covariance matrices provided by Mclust are searched over. In Mclust, the mixture model is initialized using the suggested default option, that is, to initialize the partition of data by an agglomerative hierarchical clustering approach, an extension of linkage clustering based on Gaussian mixtures (Fraley and Raftery, 2006). This initialization may be of advantage especially to data in this study because as shown previously, linkage clustering generates perfect clustering for the noisy curve data set in the first example.

Denote each data set k, $k = 1,2,...,55$, by $A_k = \{x_i^{(k)}, i = 1,...,200\}$, where $x_i^{(k)} = (x_{i,1}^{(k)}, x_{i,2}^{(k)}, ..., x_{i,50}^{(k)})^t$. For each $x_i^{(k)}$, we form a sequence of its lower dimensional vectors: $x_i^{(k,l)} = (x_{i,1}^{(k)}, x_{i,2}^{(k)}, ..., x_{i,l}^{(k)})^t$, $l = 2,3,...,50$. Let $A_{k,l} = \{x_i^{(k,l)}, i = 1,...,200\}$. HMAC and Mclust are applied to every $A_{k,l}$, $l = 2,...,50$, $k = 1,...,55$. Let the clustering error rate obtained by HMAC for $A_{k,l}$ be $r_{k,l}^{(H)}$ and that by Mclust be $r_{k,l}^{(M)}$. We summarize the clustering results in Figure 5.

To assess the variation of clustering performance over multiple implementations, we compute the percentage of the 55 data sets that are not perfectly clustered by the two methods at each dimension $l = 2,...,50$. Figure 5(a) shows the result. For HMAC, this percentage stays between 25% and 32% over all the dimensions. While, for Mclust, the percentage is consistently and substantially higher, and varies greatly across the dimensions. For $l \geq 43$, none of the data sets can be perfectly clustered by Mclust.



(a) (b)

Figure 5: Clustering results obtained by HMAC and Mclust for the 55 high dimensional noisy curve data sets. (a) The percentage of data sets that are not perfectly clustered with increasing dimensions. (b) The average and median of clustering error rates.

To examine clustering accuracy with respect to increasing dimensions, for each $l = 2, ..., 50$, the mean $\bar{r}_{\cdot,l}^{(H)}$, $\bar{r}_{\cdot,l}^{(M)}$, and the median $\tilde{r}_{\cdot,l}^{(H)}$, $\tilde{r}_{\cdot,l}^{(M)}$ over the data sets are computed and shown in Figure 5(b). For HMAC, the mean $\bar{r}_{\cdot,l}^{(H)}$ varies only slightly around 7.5% when the dimension increases, and the median $\tilde{r}_{\cdot,l}^{(H)}$ stays at zero. The error rates obtained from Mclust, on the other hand, change dramatically with the dimension. And for $l = 2$ and $l \geq 30$, both $\bar{r}_{\cdot,l}^{(M)}$ and $\tilde{r}_{\cdot,l}^{(M)}$ are significantly higher than $\bar{r}_{\cdot,l}^{(H)}$. Interestingly, the worst error rate from Mclust is achieved at $l = 2$ with $\bar{r}_{\cdot,2}^{(M)} > 23\%$ rather than at the high end of the range of dimensions. This seems to suggest that the extra noise dimensions help to mollify the effect of non-Gaussian shaped clusters. When $l \geq 30$, $\bar{r}_{\cdot,l}^{(M)}$ and $\tilde{r}_{\cdot,l}^{(M)}$ increase steadily with the dimension and eventually reaches nearly 20%.

Because it is expected that Gaussian components in the mixture model cannot well capture the half circle structure in the data, we have also tested Mclust with 3 components so that the half circle can possibly be characterized by two components. In this case, two of the three clusters need to be combined in order to compute the accuracy with respect to the original two classes. The selection of the two clusters to combine is not trivial. If we use a simple rule of combining the two clusters with the closest pair of centers, the average classification accuracies (over different dimensions) are mostly inferior to those by Mclust with 2 components. Note that the ridgeline based merging procedure in Section 4 may be invoked instead. A detailed examination in this direction is out of the scope of this paper. If we search through all the possible combinations and always choose the one that yields the best classification accuracy, the average accuracies are better than those achieved by HMAC. However, this comparison inherently favors Mclust because the true class labels are used to decide the binary grouping of the 3 components, while HMAC is purely unsupervised.

6.2 Real Data Sets

In this section, we apply HMAC to real data sets, compare our method of visualization described in Section 5 with PCA, and demonstrate the usage of the cluster merging algorithm described in Section 4.2.

6.2.1 GLASS DATA

We first examine the aforementioned glass data set using the entire set of 214 samples and all the original 9 features. Applying HMAC, a dendrogram of 10 levels is obtained, as shown in Figure 6(a). The number of clusters at each level is listed in Table 2. The sizes of the largest 4 clusters at each level are also given in this table. The dendrogram and the table show that 3 clusters containing more than 5 points are formed at the first level. These 3 prominent clusters are retained or augmented up to level 3. At level 4, two of the 3 prominent clusters are merged, leaving 2 prominent clusters which are further merged at level 7. At this level, both the dendrogram and the table suggest the main clustering structure in the data is annihilated by the very large kernel bandwidth. However, the number of clusters generated at level 7 is 7 instead of 1. Except the largest cluster, the other 5 clusters each contain no more than 3 points and in total only 9. They may be considered more appropriately as "outliers" than clusters. At even higher levels, these tiny clusters are merged gradually into the main mass of data.

Level	1	2	3	4	5	6	7	8	9	10
# clusters	29	25	18	15	13	11	7	6	4	3
Size of 1st cluster	160	163	176	177	179	180	205	208	210	211
Size of 2nd cluster	12	12	12	21	21	22	3	2	2	2
Size of 3rd cluster	9	9	9	3	3	3	2	1	1	1
Size of 4th cluster	5	6	2	2	2	2	1	1	1	not exist

Table 2: The clustering results for the full glass data set.

The above discussion suggests that to obtain a given number of clusters, it is not always a good practice to choose a level in the hierarchy that yields the desired number of clusters. We may select a level at which major clusters are merged and outliers are mistaken as plausible clusters. One remedy is to apply the cluster merging algorithm to a larger number of clusters formed at a lower level. We will present results of this approach shortly.

To compare our visualization method with PCA, the clustering results at level 3 are shown in Figure 6(b) and (c) using the two projection methods respectively. Both projections are orthonormal. In our visualization method, the projection only attempts to approximate the discriminant functions between the 3 major clusters. It is obvious that the new visualization method shows the clustering structure better than PCA. The two projection directions derived by our method are used when presenting other clustering results for a clear correspondence between points in different plots.

If we apply our cluster merging algorithm at level 3, we obtain results shown in Figure 6(d) and (e). In Figure 6(d), three clusters are formed by applying stage II merging based on coverage rate. The parameter $\rho = 95\%$. Merging based on separability is not conducted because we attempt to get 3 clusters and the 3 largest clusters at this level already account for close to 95% of the data. This clustering result is much more preferable than the 3 clusters directly generated by HMAC at level

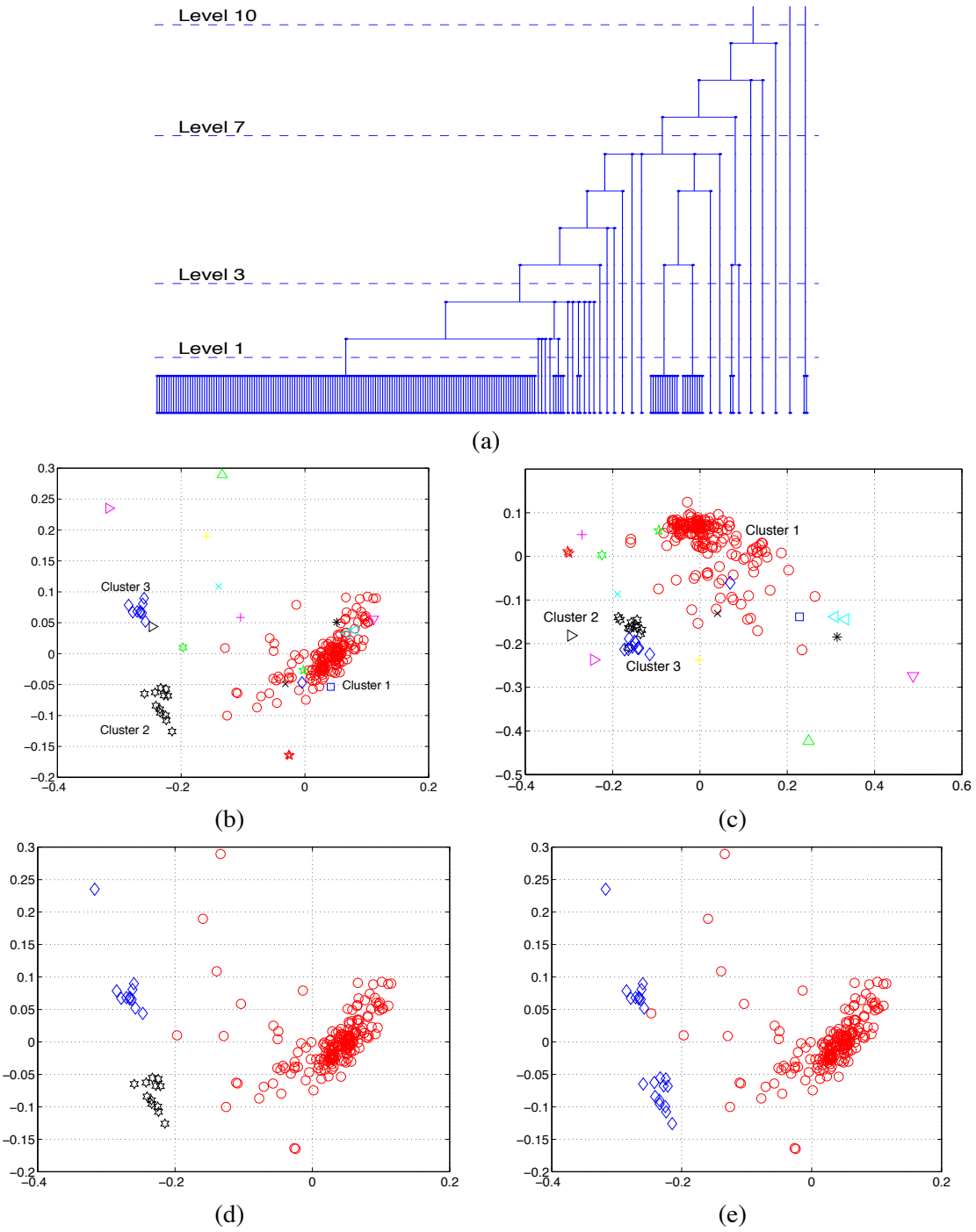


Figure 6: Results for the full glass data set. (a) The dendrogram created by HMAC. (b) Visualization by our method using regression on discriminant functions. (c) Visualization by PCA. (d), (e) Three (two) clusters obtained by applying the merging algorithm to clusters generated by HMAC at level 3.

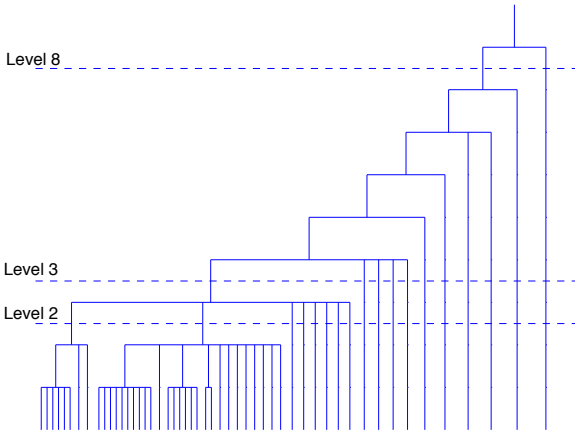


Figure 7: The dendrogram generated by HMAC for the infant data set.

10, containing 211, 2, and 1 points respectively. If we apply merging based on separability with threshold $\varepsilon = 0.4$ (stage I) and merging based on coverage rate with $\rho = 95\%$ (stage II), we obtain two clusters shown in Figure 6(e).

6.2.2 INFANT ATTENTION TIME

The second real data set is provided by Hoben Thomas, funded by the German Research Foundation Grant (Az. Lo 337/19-2), in the Department of Psychology at Penn State. In a study of infants' behavior, 51 infants were tested in two occasions apart by several months. In each occasion, a visual stimulus was given to the infants repeatedly for 11 times, with a fixed amount of time separating the stimuli. An infant's attention time in every stimulus was recorded. We thus have a data set of 51 samples with dimension 22. It is of interest to examine whether the infant data possess clustering structure and the behavior patterns of different groups. It is challenging to cluster this data set because of the high dimensionality and relatively small sample size.

Applying HMAC to the data, we obtain a dendrogram shown in Figure 7. At level 2, two prominent clusters emerge, containing 8 and 27 samples respectively. All the other clusters are singletons. At the next higher level, the two main clusters are merged. Because all the other clusters are singletons, we essentially partition the data into a main group and several outliers. There is no clear clustering structure preserved after level 2. Figure 8(a) and (b) show the clustering results obtained at level 2 using projection directions derived by our visualization method and PCA respectively. Specifically, in our method, the density of the largest cluster is used as the basis to form the discriminant functions of the other clusters. The projection directions are derived to best preserve the discriminant functions of the second and third largest clusters. The separation of the 2 main clusters is reflected better in (a) than (b). The ridgeline between the two major clusters at level 2 is computed, and the density function along this ridgeline is plotted in Figure 8(c). The plot shows that the "valley" between the two clusters is not deep comparing with the peak of the less prominent cluster, indicating weak separation between the clusters.

As the dendrogram suggests, if we need to cluster the infants into two groups, specifically, to assign the singletons into the two main clusters, we cannot simply cut the dendrogram at a level that

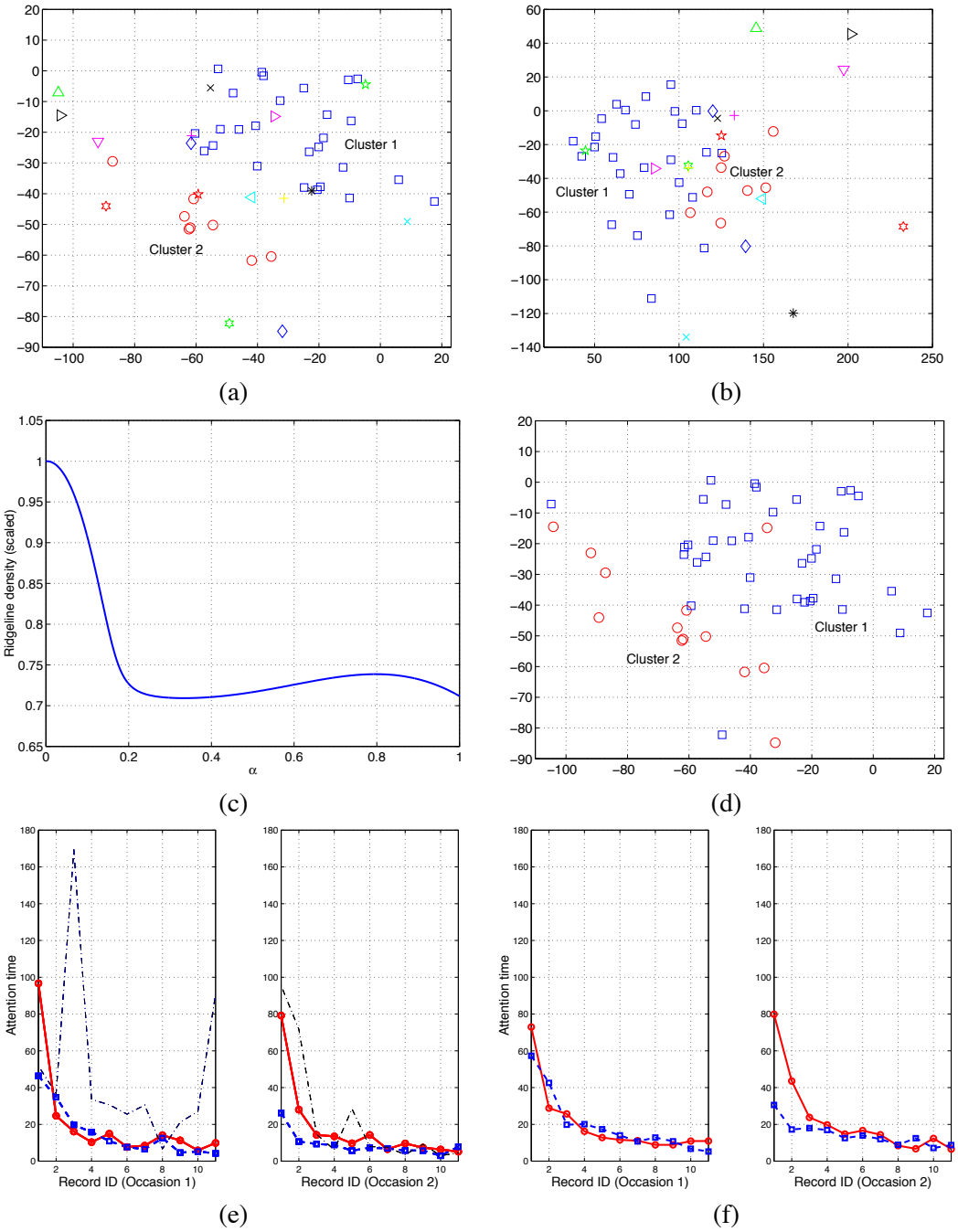


Figure 8: Results for the infant data set. (a) Visualization by our method using regression on discriminant functions. (b) Visualization by PCA. (c) Density function along the ridgeline between the two major clusters at level 2. (d) Two clusters obtained by applying the merging algorithm to clusters generated by HMAC. (e) The modal curves of the two main clusters obtained by HMAC at level 2. Dashed line: cluster 1. Solid line: cluster 2. Dash-dot line: the observation on the 10th infant, who deviates enormously from the cluster modes. (f) The mean curves of the two clusters obtained by fitting a mixture of 2 Gaussian components.

yields two clusters. For this purpose, we use the stage II merging procedure with a coverage rate of 85%. The clustering result is shown in Figure 8(d).

To compare the infants in the two main clusters identified by HMAC, we plot the "modal curves" for both test occasions in Figure 8(e). By modal curve, we refer to the mode of a cluster displayed with the dimension index as the horizontal axis and the value in that dimension as the vertical axis. It is meaningful to view a mode as a curve in this study because the experiments were conducted sequentially and dimension i corresponds to the ith measurement in the sequence. According to Figure 8(e), the main difference between the two groups of infants is their attention time for the first stimulus in each test occasion. The circle cluster exhibits a significantly longer attention time for the initial stimulus in both occasions. According to HMAC, 8 infants belong to the long initial attention time group and 27 infants belong to the short time group. The other 16 infants' data are distinct from both groups. In Figure 8(e), the attention time of the 10th infant is shown by the dash-dot lines. This infant is the most "extreme" outlier which corresponds to the right most branch in the dendrogram of Figure 8(c). The plot shows that this infant behaved very differently from the average in the first test. Instead of gradually decreasing, his attention time jumped to a very high value for the third stimulus and again for the last stimulus. If a clear-cut two groups are desired, Figure 8(d) shows that 13 infants are partitioned into the long time group and 38 into the short time group.

We also perform clustering on this data set by fitting a mixture of two Gaussian components. The EM algorithm is used to estimate the mixture model using k-means clustering to initialize, and the maximum a posterior criterion is used to partition the data. For the two clusters obtained, we display the two Gaussian mean vectors as curves in Figure 8(f). Just as in the clustering by HMAC, one cluster had longer attention time to the initial stimulus in both occasions. However, the difference between the attention time in the first occasion for the two clusters was not substantial. Using mixture modeling, the long time group included 26 infants while the short time group included 25.

6.2.3 NEWSGROUP DATA

In the previous two examples, we cannot quantitatively examine the clustering performance because there are no given class labels for the data points. In the third example, we use a data set containing documents labeled by different topics. This data set is taken from the newsgroup data (Lang, 1995). Each instance corresponds to a document in one of the two computer related topics: `comp.os.ms-windows.misc` (class 1) and `comp.windows.x` (class 2). The numbers of instances in the two classes are close: 975 (class 1) and 997 (class 2). The raw data for each document simply contain the words appeared in this document and their counts of occurrences. We process the data by stemming the words and employing two stages of dimension reduction. For details on the dimension reduction procedure, which relies mainly on two-way mixture modeling, we refer to (Li and Zha, 2006). In summary, we represent every document by a 10-dimensional vector, where each dimension is the total number of occurrences for a chosen collection of words. We then normalize the vector such that each dimension becomes the frequency of the corresponding set of words.

Due to the large data size, it is unrealistic to show the dendrogram generated by HMAC directly. Instead, we provide a summarized version of the dendrogram emphasizing prominent clusters in Figure 9(a). In order to retain as much information as possible in the summary, a cluster will be regarded "prominent" and shown individually if it contains at least 3% of the total data. This requirement for showing a cluster is rather mild. The number in each node box in the tree indicates

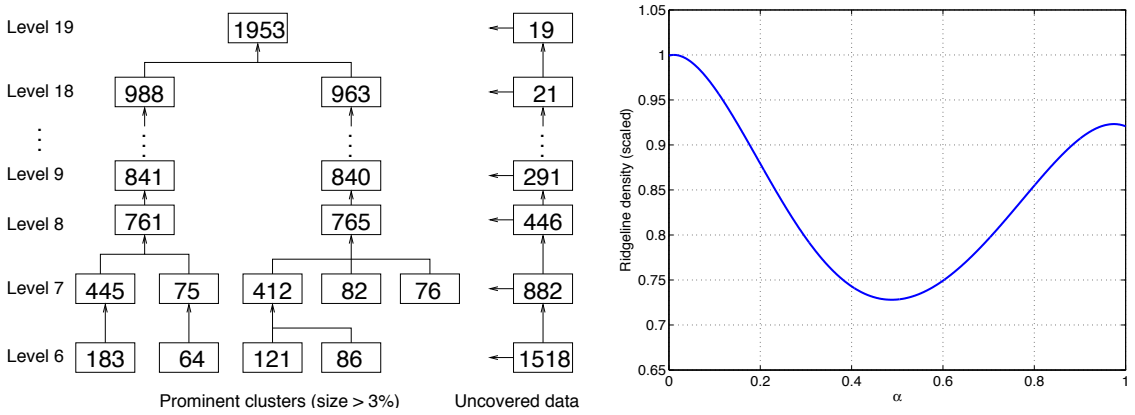


Figure 9: Results for the document data set. (a) The summarized dendrogram showing the prominent clusters. (b) The density function along the ridgeline between the two major clusters at level 12.

the cluster size. The number of data points not covered by the prominent clusters at any level is shown in the box to the right of the dendrogram. As shown in the figure, prominent clusters do not appear until level 6. At level 6, the four prominent clusters are very small. About 77% of the data are scattered in tiny clusters. At level 8, two major clusters emerge, each accounting for nearly 39% of the data. The rest data do not form any prominent clusters. The two major clusters remain through level 8 to level 18, and each absorbs more data points at every increased level. For brevity, we omit showing the sizes of these two clusters between level 9 and 18. At level 19, the two major clusters are merged, and there are 19 outlier points not absorbed into the main mass of data. This dendrogram strongly suggests there are two major clusters in this data set because before level 8, the clusters formed are too small and the percentage of data not covered by the clusters is too high. If we allow 5% points to lie outside prominent clusters, we can choose level 12 in the dendrogram. At level 12, the two clusters are of size 950 and 932. To examine the separation of the two clusters at this level, we calculate the ridgeline between them and plot the density function along the ridgeline in Figure 9(b). The mode heights of the two clusters are close, and the cluster separation is strong.

Level	8	9	10	11	12	13	14	15	16	17	18
Correct (%)	73.5	80.4	85.4	87.7	89.0	90.2	91.0	91.2	91.5	91.6	91.7
Incorrect (%)	3.9	4.9	5.7	6.1	6.4	6.5	6.8	6.8	6.9	7.1	7.2
Uncovered (%)	22.6	14.8	8.9	6.2	4.6	3.2	2.2	2.0	1.6	1.2	1.0

Table 3: The clustering accuracy of HMAC for the document data set. The percentages of points that are correctly clustered, incorrectly clustered, and not covered by the two major clusters are listed.

From level 8 to 18, we compute the percentages of points that are correctly clustered, incorrectly clustered, and not covered by the two major clusters. Table 3 provides the result. At level 18, 91.7% data points are correctly clustered. For comparison, we apply Mclust to the same data set. If we specify the range for the number of clusters as 2 to 10 and let Mclust choose the best number and the best covariance structure using BIC, the number of clusters chosen is 3. The sizes of the 3 clusters are 763, 668, and 541. The first two clusters are highly pure in the sense of containing points from the same class. The third cluster however contains about 40% class 1 points and 60% class 2. If we label the third cluster as class 2, the overall clustering accuracy is 87.6%. On the other hand, if we fix the number of clusters at 2 and run Mclust, the clustering accuracy becomes 94%.

6.2.4 DISCUSSION ON PARAMETER SELECTION

As shown by the above examples of real data clustering, it is often not obvious which level in the dendrogram produced by HMAC should be used to yield the final clustering. This is a general issue faced by agglomerative clustering approaches. Prior information about data or our implicit assumptions about good clusters can play an important role in this decision. One common assumption we make is that a valid cluster ought not be too small. Under this principle, we declare a level of the dendrogram too low (small bandwidth) if all the clusters are small and a level too high (large bandwidth) if a very large portion of the data (e.g., 95%) belong to a single cluster. For the intermediate acceptable levels in the dendrogram, we have designed the coverage rate mechanism to ensure that very small clusters are excluded. The chosen coverage rate reflects the amount of outliers one believes to exist. Despite the inevitable subjectiveness of this value, the coverage rate affects only the grouping of a small percentage of outlier points.

A more profound effect on the clustering result comes from the merging procedure based on separability which involves choosing a separability threshold. As we have discussed in Section 4.2, the merging process based on separability is essentially the directional single linkage clustering which stops when all the between-cluster separability measures are above the threshold. The threshold directly determines the final number of clusters, but is irrelevant to the full clustering hierarchy generated by the directional single linkage. Hence in situations where we have a targeted number of clusters to create, we can avoid choosing the threshold and simply stop the directional single linkage when the given number is reached. Of course, if at a certain level of the dendrogram, there exist precisely the targeted number of valid clusters, we can use that level directly rather than applying merging on clusters at a lower level. Whether the HMAC dendrogram clearly suggests the right number of clusters can be highly data dependent. For instance, in the document clustering example, the dendrogram in Figure 9(a) strongly suggests two clusters because at all the acceptable levels there are two reasonably large clusters. On the other hand, for the glass data set with results shown in Figure 6, it is not so clear-cut whether there are three or two clusters.

Another choice we need to make in HMAC is the sequence of kernel bandwidths, $\sigma_1 < \sigma_2 < \cdots < \sigma_\eta$. It is found empirically that as long as the grid of bandwidths is sufficiently fine, the prominent clusters created are not sensitive to the exact sequence. Major clustering structures often remain over a wide range of bandwidths. We have always used a uniformly spaced sequence of bandwidths in our experiments. If precisely two clusters merge at every increased level of a dendrogram, that is, the number of clusters decreases exactly by one, the dendrogram will have n levels, where n is the data size. We call such a dendrogram *all-size* since the clustering into any number of groups smaller than n appears at a certain level. We rarely observe an all-size dendrogram generated

by HMAC. On the other hand, by using extremely refined bandwidths, we indeed obtained all-size dendrograms for the example in Section 3.2 and the infant data set.

We note that linkage clustering by construction generates the all-size dendrogram. However, it is not necessarily a good practice to simply choose a level in the dendrogram that yields the desired number of clusters, as we have discussed previously for the dendrogram of HMAC. Hence, the fact that the dendrogram generated by HMAC is usually not all-size raises little concern. In practice, since the targeted number of clusters is normally much smaller than the data size, it is easy to find a relatively low level at which the number of clusters exceeds the target. We can then apply the separability based directional single linkage clustering at that level, and achieve any number of clusters smaller than the starting value.

6.3 Image Segmentation

To demonstrate the applicability of HMAC to computationally intensive tasks, we develop an image segmentation algorithm based on HMAC. A basic approach to image segmentation is to cluster the pixel color components and label pixels in the same cluster as one region (Li and Gray, 2000). This approach partitions images into regions that are relatively homogeneous. Examples of segmentation via clustering are shown in Figure 10. We employed the speeding up method described in Section 3.3. Our image segmentation method comprises the following steps: (a) Apply k-center algorithm to cluster image pixels into a given number of groups. This number is significantly larger than the desired number of regions. In particular, we set it to 100. (b) Form a data set $\{x_1,...,x_n\}$, $n = 100$, where x_i is the mean of the vectors assigned to the ith group by k-center clustering. For each x_i, assign weight w_i, where w_i is the percentage of pixels assigned to x_i. (c) Apply the weighted version of HMAC to the data set. The only difference lies in the formula for the kernel density estimator. In the weighted version, $f(x) = \sum_{i=1}^{n} w_i \phi(x \mid x_i, D(\sigma^2))$. (d) Starting from the first level of the dendrogram formed by HMAC, apply the cluster merging algorithm described in Section 4.2. If the number of clusters after merging is smaller than or equal to the given targeted number of segments, stop and output the clustering results at this level. Otherwise, repeat the merging process at the next higher level of the dendrogram. For brevity, we simply refer to this segmentation algorithm as HMAC.

To assess the computational efficiency of HMAC, we experiment with 100 digital photo images randomly selected from the Corel image database (Wang et al., 2001). Every image is of size 256×384 or 384×256. The experiments were conducted on a 1.2GHz Sun UltraSparc processor. We compare the segmentation time of HMAC and k-means. On average, it takes 4.41 seconds to segment an image using HMAC. The average number of segmented regions is 6.0. For k-means clustering, we experimented with both dynamically determining and fixing the number of segmented regions. In the dynamic case, the average number of regions generated per image by thresholding is 5.5. The average segmentation time for each image is 4.43 seconds, roughly equal to the time of HMAC. However, the computation time of k-means increases if more regions are formed for an image. If we fix the number of segmented regions to 6.0, the average segmentation time is 4.87 seconds per image.

In terms of segmentation results, whether HMAC or k-means is preferred is application dependent. K-means clusters the pixels and computes the centroid vector for each cluster according to the criterion of minimizing the mean squared distance between the original vectors and the centroid vectors. HMAC, however, finds the modal vectors, at which the kernel density estimator achieves

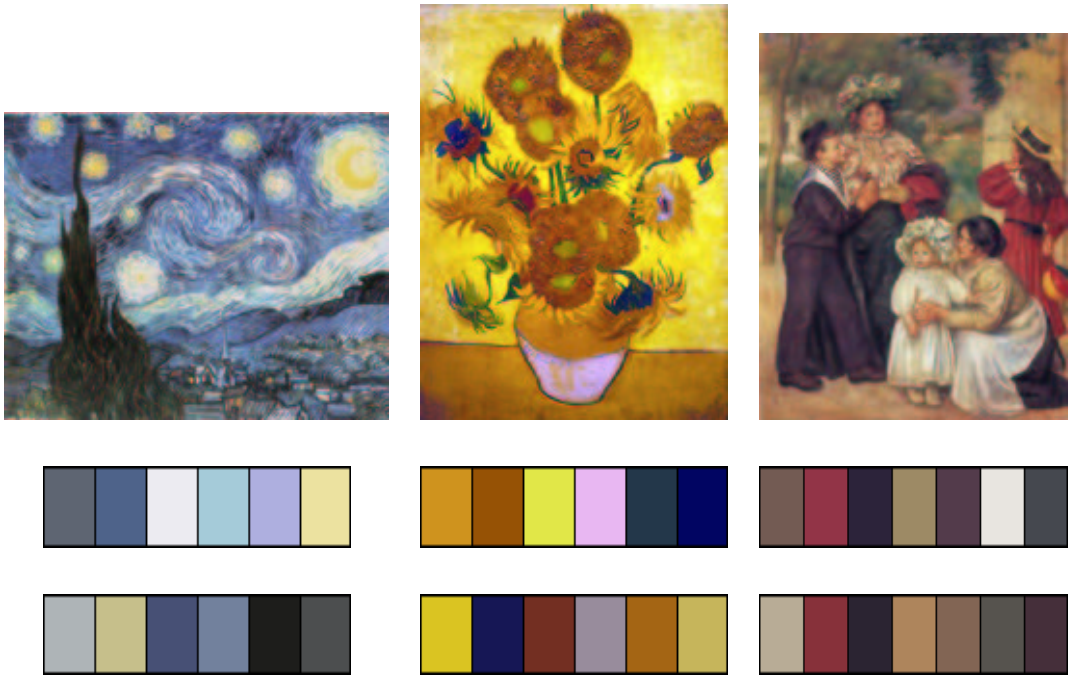


Figure 10: Segmentation results. First row: Original images. Second row: mode colors of the clusters generated by HMAC. Third row: mean colors of the clusters generated by k-means.

a local maxima. These vectors are peaks of density bumps. They are significant in the sense of possessing locally maximum density, but may not be the best approximation to the original vectors in an average sense. Figure 10 shows the representative colors extracted by HMAC and k-means for several impressionism paintings. For HMAC, the mode color vector of each cluster is shown as a color bar; and for K-means, the mean vector of each cluster is shown. The representative colors generated by k-means tend to be "muddier" due to averaging. Those by HMAC retain the true colors better, for instance, the white color of the stars in the first picture and the purplish pink of the vase in the second. On the other hand, HMAC may ignore certain colors that either are not distinct enough from others or do not contain enough pixels. For the purpose of finding the main palette of a painting, HMAC may be more preferable.

7. Conclusion

In this paper, we have introduced an EM-style algorithm, namely, Modal EM (MEM), for finding local maxima of mixture densities. For a given data set, we model the density of the data non-parametrically using kernel functions. Clustering is performed by associating each point to a mode identified by MEM with initialization at this point. A hierarchical clustering algorithm, HMAC, is developed by gradually increasing the bandwidth of the kernel functions and by recursively treating modes acquired at a smaller bandwidth as points to be clustered when a larger bandwidth is used.

The Ridgeline EM (REM) algorithm is developed to find the ridgeline between the density bumps of two clusters. A separability measure between two clusters is defined based on the ridgeline, which takes comprehensive consideration of the exact densities of the clusters. A cluster merging method based on pairwise separability is developed, which addresses the competing factors of using a small bandwidth to retain major clustering structures and using a large one to achieve a low number of clusters.

The HMAC clustering algorithm and its combination with the cluster merging algorithm are tested using both simulated and real data sets. Experiments show that our algorithm tends to unite merits of linkage clustering and mixture-model-based clustering. Applications to both simulated and real data also show that the algorithm works robustly with high dimensional data or clusters deviating substantially from Gaussian distributions. Both of these cases pose difficulty for parametric mixture modeling.

A C package at `http://www.stat.psu.edu/~jiali/hmac` is developed, which includes the implementation of the HMAC algorithm, REM for computing ridgelines, and the separability/ coverage rate based merging algorithm.

There are several directions that can be pursued in the future to strengthen the framework of modal clustering. In the current work, we use a fixed bandwidth for the kernel density functions. We can explore ways to make the bandwidth vary with the location of the data because there may not exist a single bandwidth suitable for the entire data set. Moreover, in this paper, we have discussed an approach to best visualize clusters in lower dimensions. A related and interesting question is to find a lower dimensional subspace in which the data form well separated modal clusters. We expect that the optimization of the subspace needs to be conducted as an integrated part of the modal clustering procedure.

Acknowledgments

We would like to thank the reviewers and the associate editor for insightful comments and constructive suggestions. We also thank Hongyuan Zha at Georgia Institute of Technology for pointing out some useful references, and Hoben Thomas at The Pennsylvania State University for providing us a real data set. Jia Li and Bruce Lindsay's research is supported by the National Science Foundation.

Appendix A.

We prove the ascending property of the MEM algorithm. Let the mixture density be $f(x) = \sum_{k=1}^{K} \pi_k f_k(x)$. Denote the value of x at the rth iteration of MEM by $x^{(r)}$. We need to show $f(x^{(r+1)}) \geq f(x^{(r)})$, or equivalently, $\log f(x^{(r+1)} \geq \log f(x^{(r)})$.

Let us introduce the latent discrete random variable $J \in \{1,2,...,K\}$ with prior probabilities $P(J=k) = \pi_k$. Assume that the conditional density of X given $J=k$ is $f_k(x)$. Then the marginal

distribution of X is $f(x)$, as specified above. Define the following functions:

$$\begin{aligned} L(x) &= \log f(x), \\ Q(x' \mid x) &= \sum_{k=1}^{K} p_k(x) \log \pi_k f_k(x'), \\ H(x' \mid x) &= Q(x' \mid x) - L(x') = \sum_{k=1}^{K} p_k(x) \log p_k(x') \end{aligned}$$

where $p_k(x) = P(J = k \mid X = x) = \frac{\pi_k f_k(x)}{f(x)}$ is the posterior probability of J being k given x. Denote the posterior probability mass function (pmf) given x by $\mathbf{p}(x) = (p_1(x), p_2(x), ..., p_K(x))$.

Because $H(x \mid x) - H(x' \mid x) = D(\mathbf{p}(x) \parallel \mathbf{p}(x'))$ and relative entropy $D(\cdot \parallel \cdot)$ is always nonnegative (Cover and Thomas, 1991), $H(x' \mid x) \leq H(x \mid x)$ for any pair of x and x'. On the other hand, according to the MEM algorithm,

$$\begin{aligned} x^{(r+1)} &= \operatorname*{argmax}_x \sum_{k=1}^{K} p_k(x^{(r)}) \log f_k(x) \\ &= \operatorname*{argmax}_x \left(\sum_{k=1}^{K} p_k(x^{(r)}) \log \pi_k + \sum_{k=1}^{K} p_k(x^{(r)}) \log f_k(x) \right) \\ &= \operatorname*{argmax}_x Q(x \mid x^{(r)}). \end{aligned}$$

Hence, $Q(x^{(r+1)} \mid x^{(r)}) \geq Q(x^{(r)} \mid x^{(r)})$. Finally, we prove the ascending property:

$$L(x^{(r+1)}) = Q(x^{(r+1)} \mid x^{(r)}) - H(x^{(r+1)} \mid x^{(r)}) \geq Q(x^{(r)} \mid x^{(r)}) - H(x^{(r)} \mid x^{(r)}) = L(x^{(r)}) \, .$$

Appendix B.

Recall that the Ridgeline EM algorithm aims at maximizing $\log g(x \mid \alpha) = (1-\alpha) \log g_1(x) + \alpha \log g_2(x)$, where $0 \leq \alpha \leq 1$ and $g_1(x)$ and $g_2(x)$ are two mixture densities $g_i(x) = \sum_{\kappa=1}^{T} \pi_{i,\kappa} h_{i,\kappa}(x)$, $i = 1, 2$. We prove here the ascending property of this algorithm, as described in Section 4.1.

Following the definitions in Appendix A, we form functions $L_i(x)$, $Q_i(x' \mid x)$, and $H_i(x'|x)$ for densities $g_i(x)$, $i = 1, 2$, respectively. Specifically, $L_i(x) = \log g_i(x)$, $Q_i(x' \mid x) = \sum_{\kappa=1}^{T} p_{i,\kappa}(x) \log \pi_{i,\kappa} h_{i,\kappa}(x')$, and $H_i(x' \mid x) = \sum_{\kappa=1}^{T} p_{i,\kappa}(x) \log p_{i,\kappa}(x')$, where $p_{i,\kappa}(x) = \pi_{i,\kappa} h_{i,\kappa}(x) / g_i(x)$. Note that, based on the proof in Appendix A, we have $L_i(x') = Q_i(x' \mid x) - H_i(x' \mid x)$ and $H_i(x \mid x) \geq H_i(x' \mid x)$. We now define

$$\begin{aligned} \tilde{L}(x) &= (1-\alpha) L_1(x) + \alpha L_2(x) \, , \\ \tilde{Q}(x' \mid x) &= (1-\alpha) Q_1(x' \mid x) + \alpha Q_2(x' \mid x) \, , \\ \tilde{H}(x' \mid x) &= (1-\alpha) H_1(x' \mid x) + \alpha H_2(x' \mid x) \, . \end{aligned}$$

According to the Ridgeline EM algorithm, $x^{(r+1)} = \operatorname{argmax}_{x'} \tilde{Q}(x' \mid x^{(r)})$. Hence $\tilde{Q}(x^{(r+1)} \mid x^{(r)}) \geq \tilde{Q}(x^{(r)} \mid x^{(r)})$. Also, it is obvious that $\tilde{H}(x^{(r)} \mid x^{(r)}) \geq \tilde{H}(x^{(r+1)} \mid x^{(r)})$. Finally, we prove the ascending property:

$$\tilde{L}(x^{(r+1)}) = \tilde{Q}(x^{(r+1)} \mid x^{(r)}) - \tilde{H}(x^{(r+1)} \mid x^{(r)}) \geq \tilde{Q}(x^{(r)} \mid x^{(r)}) - \tilde{H}(x^{(r)} \mid x^{(r)}) = \tilde{L}(x^{(r)}) \, .$$

Appendix C.

We prove here the graph constructed in Section 4.2, where every clique corresponds to a node, has no loop. We denote a node (i.e., a clique) by c_i. By construction, a directed edge from c_i to c_j exists if the following conditions are satisfied:

1. $S_c(c_i, c_j) < \varepsilon$.
2. $S_c(c_i, c_j) = \min_{k \neq i} S_c(c_i, c_k)$ and j is the smallest index among all those j''s that achieve $S_c(c_i, c_{j'}) = \min_{k \neq i} S_c(c_i, c_k)$.
3. $\delta(c_i) < \delta(c_j)$.

We refer to the three conditions as Condition 1, 2, 3.

We prove the non-existence of loops by contradiction. We will first show that if there is a loop in the graph, this loop is directed. Then, we will prove that a directed loop cannot exist. Without loss of generality, assume that there is a loop connecting nodes $\{c_1, c_2, ..., c_k\}$ sequentially. The edges in the loop are $\{e_{1,2}, e_{2,3}, ..., e_{k-1,k}, e_{k,1}\}$, where $e_{i,j}$ connects c_i and c_j. Let $head(e_{i,i+1})$ indicate the node from which edge $e_{i,i+1}$ starts. Obviously, $head(e_{i,i+1}) = c_i$ or c_{i+1}.

Without loss of generality, let $head(e_{k,1}) = c_k$. By Condition 2, every node can have almost one edge starting from it. Hence $head(e_{k-1,k}) = c_{k-1}$. For an arbitrary $j > 1$, assume that for $i = j, j+1, ..., k-1$, we have $head(e_{i,i+1}) = c_i$. Since $head(e_{j,j+1}) = c_j$, again by Condition 2, $head(e_{j-1,j}) = c_{j-1}$. Thus, for $i = j-1, j, ..., k-1$, we have $head(e_{i,i+1}) = c_i$. By induction, for any $i = 1, ..., k-1$, we have $head(e_{i,i+1}) = c_i$. Therefore, the loop connecting $\{c_1, c_2, ..., c_k\}$ is directed.

By Condition 3 of the graph construction procedure, if $head(e_{i,j}) = c_i$, then $\delta_i < \delta_j$. Thus, if there is a directed loop connecting nodes $\{c_1, c_2, ..., c_k\}$ and $head(e_{i,i+1}) = c_i$, we get the contradiction: $\delta_1 < \delta_2 < \cdots < \delta_k < \delta_1$. This proves that there is no loop (regardless of directed or not) in the graph.

References

A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6:1345-1382, 2005.

J. D. Banfield and A. E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803-821, 1993.

C. Blake, E. Keogh, and C. J. Merz. *UCI repository of machine learning databases*. *http://www.ics.uci.edu/mlearn/MLRepository.html*, Dept. of Information and Computer Science, UCI, 1998.

G. Celeux and G. Govaert. Comparison of the mixture and the classification maximum likelihood in cluster analysis. *Journal of Statistical Computation & Simulation*, 47:127-146, 1993.

S. V. Chakravarthy and J. Ghosh. Scale-based clustering using the radial basis function network. *IEEE Trans. Neural Networks*, 7(5):1250-1261, 1996.

M.-Y. Cheng, P. Hall, and J. A. Hartigan. Estimating gradient trees. *A Festschrift for Herman Rubin, IMS Lecture Notes Monogr. Ser.*, 45:237-49, Inst. Math. Statist., Beachwood, OH, 2004.

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

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 1991.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal Royal Statistics Society*, 39(1):1-21, 1977.

B. S. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. Oxford University Press US, 2001.

C. Fraley and A. E. Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97:611-631, 2002.

C. Fraley and A. E. Raftery. MCLUST Version 3 for R: Normal mixture modeling and model-based clustering. *Technical Report*, no. 504, Department of Statistics, University of Washington, 2006.

T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoret. Comp. Sci.*, 38(22):293-306, 1985.

J. C. Gower and G. J. S. Ross. Minimum spanning trees and single linkage cluster analysis. *Applied Statistics*, 18(1):54-64, 1969.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., NJ, USA, 1988.

A. K. Jain , M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264-323, 1999.

D. Joshi, J. Li, and J. Z. Wang. A computationally efficient approach to the estimation of two- and three-dimensional hidden Markov models. *IEEE Transactions on Image Processing*, 15(7):1871-1886, 2006.

J. R. Kettenring. The practice of cluster analysis. *Journal of Classification*, 23(1):3-30, 2006.

K. Lang. NewsWeeder: Learning to filter netnews. In *Proceeding of the International Conference on Machine Learning*, pages 331-339, 1995.

Y. Leung, J.-S. Zhang, and Z.-B. Xu. Clustering by scale-space filtering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(12):1396-1410, 2000.

J. Li. Two-scale image retrieval with significant meta-information feedback. In *Proc. ACM Multimedia*, pages 499-502, Singapore, November 2005.

J. Li. Clustering based on a multi-layer mixture model. *Journal of Computational and Graphical Statistics*, 14(3):547-568, 2005.

J. Li and R. M. Gray. *Image Segmentation and Compression Using Hidden Markov Models*. Springer, 2000.

J. Li and J. Z. Wang. Real-time computerized annotation of pictures. In *Proc. ACM Multimedia Conference*, pages 911-920, ACM, Santa Barbara, CA, October 2006.

J. Li and H. Zha. Two-way Poisson mixture models for simultaneous document classification and word clustering. *Computational Statistics and Data Analysis*, 50(1):163-180, 2006.

G. J. McLachlan and D. Peel. *Finite Mixture Models*. New York: Wiley, 2000.

M. C. Minnotte and D. W. Scott. The mode tree: A tool for visualization of nonparametric density features. *Journal of Computational and Graphical Statistics*, 2(1):51-68, 1993.

M. C. Minnotte, D. J. Marchette, and E. J. Wegman. The bumpy road to the mode forest. *Journal of Computational and Graphical Statistics*, 7(2):239-51, 1998.

N. R. Pal and S. K. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26:1277-94, 1993.

A. Pothen, H. D. Simon, and K. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430-452, 1990.

S. Ray and B. G. Lindsay. The topography of multivariate normal mixtures. *Annals of Statistics*, 33(5):2042-2065, 2005.

S. J. Roberts. Parametric and nonparametric unsupervised clustering analysis. *Pattern Recognition*, 30(2):261-272, 1997.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888-905, 2000.

J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):947-963, 2001.

R. Wilson and M. Spann. A new approach to clustering. *Pattern Recognition*, 23(12):1413-25, 1990.

C. F. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95-103, 1983.