Separation Index, Variable Selection and Sequential Algorithm for Cluster Analysis

by

Weiliang Qiu

B.Sc., Beijing Polytechnic University, 1996

M.Sc., Beijing Polytechnic University, 1999

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

(Department of Statistics)

we accept this thesis as conforming
to the required standard

The University of British Columbia

September 2004

Abstract

This thesis considers four important issues in cluster analysis: cluster validation, estimation of the number of clusters, variable weighting/selection, and generation of random clusters.

Any clustering method can partition data into several subclusters. Hence it is important to have a method to validate obtained partitions. We propose a cluster separation index to address the cluster validation problem. This separation index is based on projecting the data in the two clusters into a one-dimensional space, in which the two clusters have the maximum separation. The separation index directly measures the magnitude of gap between pair of clusters, is easy to compute and interpret, and has the scale equivariance property.

The ultimate goal of cluster analysis is to determine if there exist patterns (clusters) in multivariate data sets or not. If clusters exist, then we would like to determine how many there are in the data set. We propose a sequential clustering (SEQCLUST) method that produces a sequence of estimated number of clusters based on varying input parameters. The most frequently occurring estimates in the sequence lead to a point estimate of the number of clusters with an interval estimate.

For a given data set, some variables may be more important than others to be used to recover the cluster structure. Some variables, called noisy variables, may even mask cluster structures. It is necessary to downweight or eliminate the effects of noisy variables. We investigate when noisy variables will mask cluster structures, and propose a weight-vector averaging idea and a new noisy-variable-detection method, which does not require the specification of the true number of clusters.

Simulation study is an important tool to assess and compare performances of clustering methods. The qualities of simulated data sets depend on cluster generating algorithms. We propose a design to generate simulated clusters so that the distances of simulated clusters to their neighboring clusters can be controlled and that the shapes, diameters and orientations of the simulated clusters can be arbitrary.

We also propose low-dimensional visualization methods and a method to determine the partial memberships of data points that are near boundaries among clusters.

Contents

List of Tables

List of Figures

Notation, Abbreviations and Conventions

We follow the widely used conventions throughout the thesis. Latin upper-case letters X, Y, Z, with or without subscripts, are used for random variables. Latin upper-case letters A, B, D, are used for matrices. Bold Latin upper-case letters (e.g. $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$) are used as random vectors. Greek lower-case letters (e.g. α, β, γ), with or without subscripts, are used for parameters in models. Bold Greek lower-case letters (e.g. $\boldsymbol{\beta}$, $\boldsymbol{\theta}$, $\boldsymbol{\eta}$) stand for parameter vectors. For a vector a matrix, the transpose is indicated with a superscript of T. All vectors are column vectors. The following is a table of symbols and abbreviations used throughout the thesis.

Symbol	Meaning
n	the number of objects
p	the number of variables
p_1	the number of non-noisy variables
p_2	the number of noisy variables
k_0	the number of clusters
i	the index for the i-th object
j	the index for the j-th variable
k	the index for the k-th cluster
ϕ	the probability density function of the standard normal distribution
Φ	the cumulative distribution function of the standard normal distribution
$\sim$	"distributed as" or "has its distribution in "
rv	random variable
MLE	maximum likelihood estimate (estimation)
sd	standard deviation
$X\|Y$	rv X conditional on rv Y

Acknowledgements

Foremost I would like to thank my research supervisor, Dr. Harry Joe, for his guidance in the development of my thesis, for his excellent advice and support and for his great patience. Without him, this thesis would never have been completed. And I feel honored to have had the chance to work with him.

I am very grateful to Dr. William J. Welch and Dr. Ruben H. Zamar for their careful reading of the manuscript, their valuable comments and suggestions.

Many thanks to Dr. Nancy Heckman, Dr. John A. Petkau and Dr. Lang Wu for their encouragement and help, both academically and personally.

Also I would like to thank Dr. Jenny Bryan for her valuable comments and suggestions on the proposal of my thesis.

I am also very grateful to Dr. David Brydges, Dr. Raymond Ng, Dr. Paul Gustafson, and Dr. Peter Hooper for their invaluable comments, advices and suggestions to this thesis.

I am honored that Dr. Ramanathan Gnanadesikan, Dr.Jon Kettenring, and Dr. Hugh Chipman kindly replied my e-mail requests about cluster analysis and offered me many advices and suggestions.

Special thanks to Christine Graham, Rhoda Morgan, and Elaine Salameh for all their help with administrative matters.

Also I would like to thank the department for giving me the great opportunity to study at this world-wide famous institute.

Furthermore I thank my fellow graduate students: Steven Wang, Rong Zhu, Sijin Wen, YinShan Zhao, Rachel M. Altman, Jochen Brumm, Justin Harrington, Aihua Pu, Lei Han, Wei Liu, Ruth Zhang, Mike Danilov, Hossain Md. Shahadut, Wei Wang, Fei Yuan and all of the others.

I cannot express how grateful I am to my parents, Weisheng Qiu and FengLan Li, for their continual love, encouragement and support. My parents have helped me through many a difficult moment. I also need to thank my brother, Weiyang Qiu, for always understanding me.

Finally, I would like to thank my wife, Kunling Wu, for her tireless love, patience and support.

WEILIANG QIU

The University of British Columbia
September 2004

To my parents and my wife

Chapter 1

Introduction

Cluster analysis is an exploratory tool to detect the natural groups (or clusters) in data sets so that within a cluster, objects are "similar" to each other and that between clusters, objects are "dissimilar" to each other. Cluster analysis can be applied to many fields of study, such as life sciences, behavioral and social sciences, earth sciences, medicine, engineering sciences, and information and policy sciences (Anderberg, 1973).

Cluster analysis has a fairly long history, but research is still needed for things like comparing partitions from different clustering methods, deciding on the number of clusters, variable weighting and selection for forming clusters. This thesis makes new research contributions to these challenges in cluster analysis.

1.1 Challenges

Clustering is a quite challenging problem since we do not know in advance how many clusters exist in a data set and there is no unified definition of "cluster". For example, for the data set illustrated in Figure 1.1, either a 2-cluster partition or a 6-cluster partition seems fine. It depends on subject knowledge about the data set to decide if the final number of clusters is 2 or 6. Again it depends on subject knowledge to decide which similarity measure is appropriate. Another difficulty is how to define "similarity"? For example, in Figure 1.2, is point A closer to point C or to point B?

For the example in Figure 1.2, we can visualize the cluster structure so that we can choose a suitable similarity measure and check if the obtained partition is appropriate or not. However in

Figure 1.1: Is the number of clusters 2 or 6?

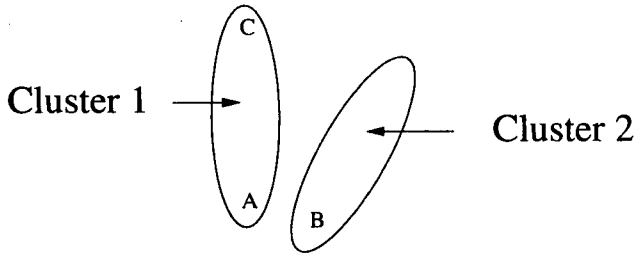


Figure 1.2: Is the point A is closer to the point C or to the point B?

real applications, data sets are usually in high dimensional space so that we could not verify the clustering results by eye.

Furthermore, there are noisy variables, outliers, missing values, and measurement errors which make the discovery of the cluster structure (the number of clusters and the partition) more difficult.

Many methods have been proposed to deal with these difficulties. However these problems still are not solved to satisfaction.

For example, for a specific data set, many internal validation methods have been proposed to check the appropriateness of a partition (the number of clusters and the partition) of the data set by using information obtained strictly from within the clustering process. Recent reviews on this area can be found in Halkidi et al. (2001) and Kim et al. (2003). As Kim et al. (2003) pointed out the main limitation of these validation methods is that they focused on only the compactness. However, compactness might not be a good measure to check the appropriateness of a partition. Some validation indexes emphasize external isolation (between-cluster variation). However those indexes focused only on the distance between cluster centroids and hence have limitations in their ability to provide a meaningful interpretation of structure in the data.

To systematically compare the performances of different clustering algorithms, simulated data sets play an important role since their cluster structures are known and can be controlled. Important features of simulated data sets include that (1) the distances between clusters could be controlled; (2) the shapes, diameters, and orientations of clusters could be arbitrary. However, there seems to be no systematic research as to how to generate simulated data sets which have both these features.

Estimation of the number of clusters is an important and challenging issue. In real problems, subject-matter knowledge can provide only a rough range of the number of clusters. Many methods have been proposed to estimate the number of clusters. The problem is still not solved to satisfaction.

For a given data set, some variables may be more important than others to be used to recover the cluster structure. Some variables, called noisy variables, may even mask the cluster structure. It is necessary to downweight or eliminate the effects of noisy variables. Most existing variable weighting/selection methods require the specification of the true number of clusters. What are the effects if the specified number of clusters is not correct? There seems no research to address this important issue. Moreover most existing methods are heuristic.

1.2 Scope

This dissertation concerns the following topics:

- Cluster validation
- Generation of simulated clusters.
- Estimation of the number of clusters
- Elimination or down-weighting the effect of noisy variables

We address these issues for the most common situation for which clustering methods are used: (1) clusters are all convex in shape; (2) variables are all continuous type; and (3) there are no missing values in data sets. We use Euclidean distance to measure the dissimilarity among data points. With these assumptions, cluster analysis methods essentially optimize some criterion, for

example, minimize within cluster dispersion. We will not consider clustering based on connectivity to handle non-convex-shaped clusters.

There is a trade-off of speed versus quality in cluster analysis. Our goals are (1) to get good quality partitions at reasonable speed; (2) to propose methods that work well and quickly for moderate size samples (cluster sizes of order several thousands and number of clusters of order 2 to 20).

1.3 Major Contributions

The new major contributions are

1. We propose a cluster validation/separation index which can be used to check the appropriateness of a partition and to compare different partitions. This validation index is different from existing internal validation indexes in that it directly measures the magnitude of the sparse area between pairs of clusters and is easy to interpret. Under certain conditions, this separation index is affine invariant and is easy to calculate.

2. We propose a weight-vector averaging idea and a variable weighting/selection method. Our approach is different from existing methods in that we can show theoretically that under certain conditions, the population version of the new variable weighting/selection method assigns zero weights to noisy variables and the obtained weights are scale equivariant. Combined with the weight-vector averaging idea we propose, the new variable weighting/selection method does not require the specification of the true number of clusters. A preliminary theoretical validation of this approach is given in this dissertation.

3. We propose a design to generate simulated clusters so that the distances of simulated clusters to their neighbor clusters can be controlled and that the shapes, diameters and orientations of the simulated clusters can be arbitrary.

4. We propose a low dimensional visualization method to help validate partitions. This method is different than existing methods in that it projects two clusters at a time and that the two clusters have the maximum separation along the orthogonal projection directions.

5. We propose a two-step method to assign partial memberships for "boundary points". It differs from fuzzy clustering methods in that it has the property of assigning partial membership only to points at boundaries of clusters.

6. We improve the ISODATA clustering method which simultaneously estimates the number of clusters and obtains the partition of a data set. The key improvements are the merging and splitting criteria.

7. We study the above issues with a theoretical basis.

1.4 Outline

The outline of chapters is as follows:

Chapter 2: Introduction of a new validation/separation index with its three applications: (1) cluster validation; (2) low dimensional visualization; and (3) assigning partial memberships.

Chapter 3: Proposal of a new design to generate simulated clusters.

Chapter 4: Proposal of a sequential clustering method to improve the ISODATA clustering method.

Chapter 5: Proposal of a new variable weighting/selection method.

Chapter 6: Summary and future research.

Chapter 2

Separation Index and Partial Membership for Clustering

2.1 Introduction

Cluster analysis is an exploratory tool to detect the natural groups (or clusters) in data sets. Two fundamental questions in cluster analysis are (1) how many clusters exist in a data set? (2) how do we get a good partition of the data set? Many methods have been proposed to address these two issues. Simulated data sets and real data sets with known cluster structures (including the number of clusters and the partition) have been used to check and compare the performances of these methods. However for a data set whose true cluster structure is unknown, how do we know if the estimated cluster structure or partition is appropriate or not? And if we apply different methods to the same data set, how do we know which partition is better? These are the main issues that we are going to address in this chapter. More specifically, we try to answer the following three questions:

1. Given the number of clusters, how do we compare the partitions obtained by applying different clustering methods to the same data set.

2. Given a clustering method, how do we check if the specified number of clusters is appropriate or not?

3. How do we compare different partitions with different numbers of clusters?

Cluster analysis usually is used as the first step to other data analysis, such as data reduction, classification, prediction based on groups, and model fitting (Ball 1971). The performances of these further analyses depend on the quality of the results of cluster analysis. So it is necessary to check the appropriateness of the partitions.

If we have subject-matter knowledge of the data set, then we might have some idea about the type of cluster structure. For the same type of cluster structure, there are many number-of-cluster-estimation methods and clustering methods available. If the results of these methods are similar to each other, then we can arbitrarily choose one. However if the results obtained by these methods are not consistent, then we need a method to compare their performances without the knowledge of the true cluster structure. If data points are in (projected) low dimensional space (e.g. one, two, or three dimension), then we can visualize the partitions and check which partition is better. However in cluster analysis, data points are usually in high dimensional space. So in general we could not visualize the partitions except for some special cases where the cluster structure in high dimensional space can be observed in (projected) low dimensional space. This again suggests the need for methods to check and compare the partitions.

In this chapter, we propose a cluster separation index to address the cluster validation problem. This separation index is based on the gap between pair of clusters whereas other previous inter-cluster measures do not directly quantify the magnitude of the gap.

It is also easy to interpret the value of the separation index we propose. A negative value of the separation index indicates that two clusters are overlapping; a zero value indicates that two clusters are touching; and a positive value indicates that two clusters are separated. The separation index we propose is easy to compute and has scale invariance property.

The separation index not only can be used to check and compare the partitions, but also has many other applications. Based on this separation index, we develop low dimensional projection methods to help visualize the distance between a pair of clusters. In real problems, it is common that clusters are close to each other. It makes sense to give a measure to indicate to which extent a data point belongs to a cluster. We develop such a measure based on the separation index.

We first give a literature review on cluster validation indexes in Section 2.2. Then in Section 2.3, we give the motivation, definition, and properties of the separation index. In Section 2.4, we propose methods to check and compare partitions based on the separation index we propose.

One simulated data set and one real data set will be used to illustrate the performance of the validation methods. Low dimensional projections based on the separation index are described in Section 2.5. The application of the projections to determine a partial membership of data points is given in Section 2.6. Discussion is given in Section 2.7.

2.2 Related Ideas

Many internal criterion measures have been proposed to validate the cluster structures (e.g. Milligan 1981; Halkidi et al. 2001; Lin and Chen 2002; Kim et al. 2003). An internal criterion measure assesses the significance of a partition using information obtained strictly from within the clustering process (Milligan, 1981). A better value of the internal criterion measure corresponds to a better partition. Milligan (1981) compared the performances of thirty such internal criterion measures and found that a subset of 6 internal criterion measures (Gamma, $\underline{C}$ index, Point-Biserial, Tau, $\underline{W}/\underline{B}$ and $\underline{G}(+)$) which have relatively better performances.

The main motivation of many internal criterion measures studied in Milligan (1981), such as $\underline{C}$ `index`, `Point-Biserial` and $\underline{W}/\underline{B}$, is to measure the compactness of clusters which can be described by internal cohesion and external isolation. Internal cohesion means that objects within the same cluster should be similar to each other at least within the local metric. External isolation means that objects in one cluster should be separated from objects in another cluster by fairly empty areas of space (Cormack 1971; Everitt 1974). The "true" cluster structure should be internally cohesive and externally isolated. The internal cohesion is usually measured by the within-cluster sum of squares and the external isolation is usually measured by the between-cluster sum of squares.

However compactness may not be a good measure to check the appropriateness of a partition or to compare the performances of partitions. In this situation, we think that external isolation is better than compactness. That is, the more externally isolated a partition is, the better the partition is. To illustrate this point, we generate a small data set with two clusters from bivariate normal distributions $\mathrm{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, where

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_2 = \begin{pmatrix} 10 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 100 \end{pmatrix}.$$

Each cluster contains 100 data points. The 2-cluster partitions obtained by the clustering methods `clara` (Kaufman and Rousseeuw 1990) and `Mclust` (Fraley and Raftery, 2002a, b) are shown in the left and right panels of Figure 2.1 respectively. The values of $\underline{C}$ `index`, `Point-Biserial` and

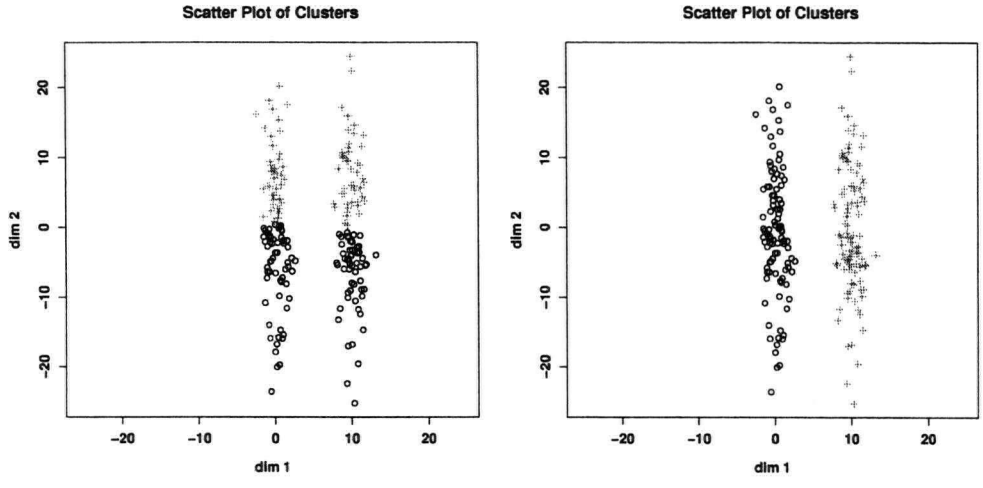


Figure 2.1: Left Panel: The 2-cluster partition obtained by `clara`. Right Panel: The 2-cluster partition obtained by `Mclust`. The circles represent cluster 1 and the symbol "+"'s represent cluster 2.

$\underline{W/B}$ are shown in Table 2.1. Smaller values of these internal criterion measure values indicate

Table 2.1: The values of internal criterion measures

	$\underline{C}$ `index`	`Point-Biserial`	$\underline{W/B}$
`clara`	3.12	2.66	207.17
`Mclust`	21.34	58.48	685.31

more compact cluster structures. By eye, we can see that the partition obtained by `Mclust` is better than the partition obtained by `clara` since two clusters obtained by `Mclust` are more isolated than those obtained by `clara`. However, the values of $\underline{C}$ `index`, `Point-Biserial` and $\underline{W/B}$ indicate that the partition obtained by `clara` is more compact than that obtained by `Mclust`. Note that this result would be quite different if we scale the variables. The issue on variable weighting/selection will be addressed in Chapter 5.

Not like other cluster validation methods involving a single summary value for a partition, Lin and Chen (2002) proposed a cluster validation method involving a summary value, called `cohesion`, for each pair of clusters. A low cohesion index value indicates well-separated cluster

structures. The cohesion index is in fact a variant of the average distance between two clusters. Lin and Chen (2002) used `joinability` instead of the usual distance to measure the closeness of one data point in one cluster to other cluster. However the average distance between two clusters may not directly measure how far apart two clusters are. For the previous example, the cohesion index values for the partitions obtained by `clara` and `Mclust` are 0.37 and 0.56 respectively. That is, based on the cohesion index values, the two clusters obtained by `clara` are more separated than those by `Mclust`. This contradicts the intuition from Figure 2.1.

Höppner et al. (1999, page 191) mentioned a separation index to measure the overall compactness of clusters. The separation index is defined as

$$D = \min_{i=1,\ldots,k_0} \left\{ \min_{j=1,\ldots,k_0, j\neq i} \left[\frac{d(C_i, C_j)}{\max_{k=1,\ldots,k_0} \operatorname{diam}(C_k)} \right] \right\}$$

where C_k is cluster k,

$$\operatorname{diam}(C_k) = \max\{d(\mathbf{y}_i, \mathbf{y}_j) | \mathbf{y}_i, \mathbf{y}_j \in C_k\},$$

and $d(\mathbf{y}_i, \mathbf{y}_j)$ is a distance function, and

$$d(C_i, C_j) = \min\{d(\mathbf{y}_i, \mathbf{y}_j) | \mathbf{y}_i \in C_i, \mathbf{y}_j \in C_j\}.$$

Actually

$$D_{ij} = \frac{d(C_i, C_j)}{\max_{k=1,\ldots,k_0} \operatorname{diam}(C_k)}$$

is a measure of distance between two clusters, i.e., D_{ij} is the the nearest distance between the two clusters normalized by the maximum cluster diameter. We can see that the separation index D emphasizes more the external isolation. For the previous example, the D values for the partitions obtained by `clara` and `Mclust` are 0.017 and 0.109 respectively. The inequality $0.109 > 0.017$ indicates that the partition obtained by `Mclust` is better than the partition obtained by `clara`. However D is sensitive to outliers.

The separation index mentioned in Höppner et al. (1999, page 191), the cohesion index proposed by Lin and Chen (2002), and the internal criterion measures studied in Milligan (1981) are for "hard partitions" — the membership of a data point to a cluster can only take values 1 or 0. Many cluster validation indexes have been proposed for "soft" or fuzzy partitions — the membership of a data point to a cluster can be any value between 0 and 1 depending on the distance of the data point to the cluster. A recent review of these validation indexes can be found in Halkidi

et al. (2001) and Kim et al. (2003). Kim et al. (2003) pointed out that the main limitation of these indexes, such as partition coefficient and partition entropy (Bezdek 1974a, b), is that they focused on only the compactness. Some indexes, such as Xie and Beni's index (Xie and Beni 1991) and CWB index (Rezaee et al. 1998), emphasize external isolation. However those indexes focused only on the distance between cluster centroids and hence are limited in their ability to provide a meaningful interpretation of structure in the data. For example, in Figure 2.2, the distance between the two cluster centers in the upper panel is the same as that in the lower panel. However, the two clusters are separated in the upper panel while the two clusters are overlapped in the lower panel.

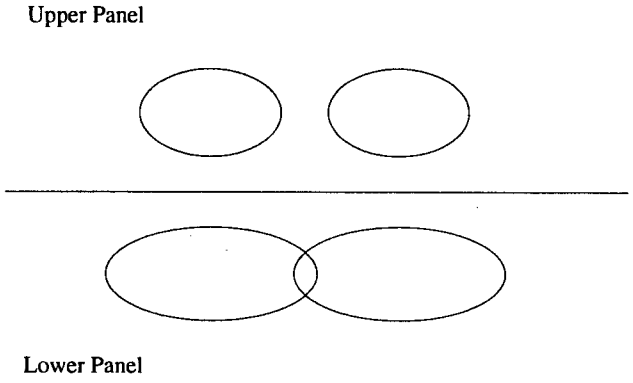


Figure 2.2: An example illustrates that the distance between cluster centers is not a good measure of the external isolation.

Kim et al. (2003) proposed a fuzzy validation index based on inter-cluster proximity which measures the degree of overlap between clusters. A low index value indicates well-partitioned clusters. This index emphasizes the external isolation and has meaningful interpretation. However this index is designed specifically for the partitions obtained by fuzzy clustering algorithms.

2.3 A Separation Index

In this section, we propose a geometric approach to get cluster separation indexes for any two clusters in a partition. A separation index matrix is then a summary of a partition. Our new cluster separation index directly measures the magnitude of the gap or sparse area between pair of clusters. This separation index is easy to compute and interpret, and has the scale invariance property. Also, the projections associated with separation index lead to a method to determine the partial memberships of data points that are near boundaries among clusters.

2.3.1 Motivation and Definition

The motivation for the separation index we propose is based on the observation that two sets of data points are regarded as two distinct clusters only if there exists a gap or sparse area between these two sets. So it is natural to measure the degree of separation between two clusters based on the gap or sparse area between them. The larger the gap is, the more separated the two clusters are. If there is no gap or sparse area between the two clusters, then there is doubt about two distinct clusters. For example, there are two obvious clusters in the left panel of Figure 2.3 because there exists a gap area between the two clusters. We doubt that there exist two clusters in the right panel of Figure 2.3 because there is no gap or sparse area between the two clusters, i.e. the density of data points along the boundary of the two clusters are relatively high. The left panel of Figure 2.3 also shows that the separation index based on the minimum distance between the two clusters (i.e. distance between point A and B) might not be a good measure of the gap area.

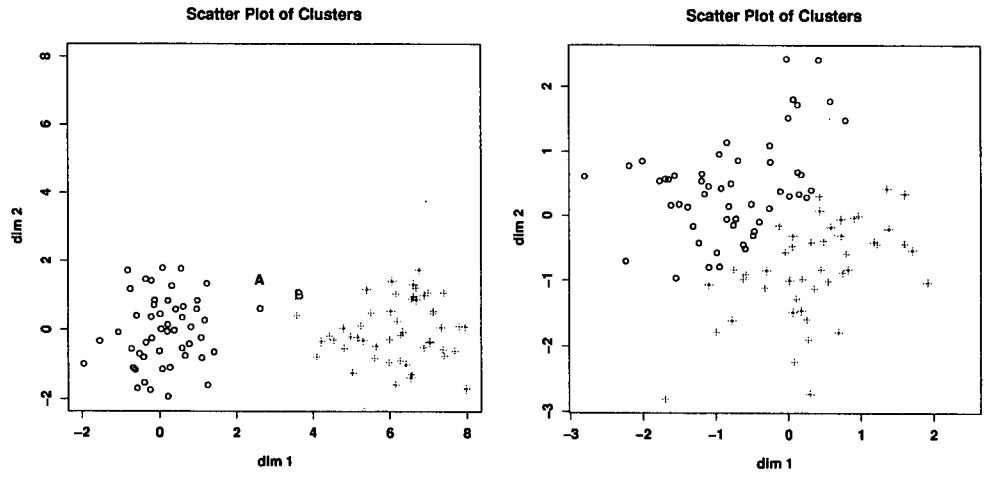


Figure 2.3: Left panel: There exist two obvious clusters. Right Panel: Cluster structure not obvious. Circles represent cluster 1 and the symbol + represents cluster 2.

We first consider how to define a good measure of the magnitude of the gap for two clusters in one dimensional space. Denote x_{1i}, $i = 1, \ldots, n_1$, as n_1 data points in cluster 1 and x_{2j}, $j = 1, \ldots, n_2$, as n_2 data points in cluster 2. One possible measure of the magnitude of the gap between the two clusters is

$$J = L_2(\alpha/2) - U_1(\alpha/2), \tag{2.3.1}$$

where $L_i(\alpha/2)$ and $U_i(\alpha/2)$ are the sample lower and the upper $\alpha/2$ quantile of cluster i (we assume that cluster 1 is on the left-hand side of cluster 2). Figure 2.4 illustrates that the separation index J^* can summarize the gap area between the two clusters. It is less sensitive to outliers than indexes based on $\min_j x_{2j} - \max_i x_{1i}$, such as Dun and Dun-like indexes (Halkidi et al. 2001). The parameter

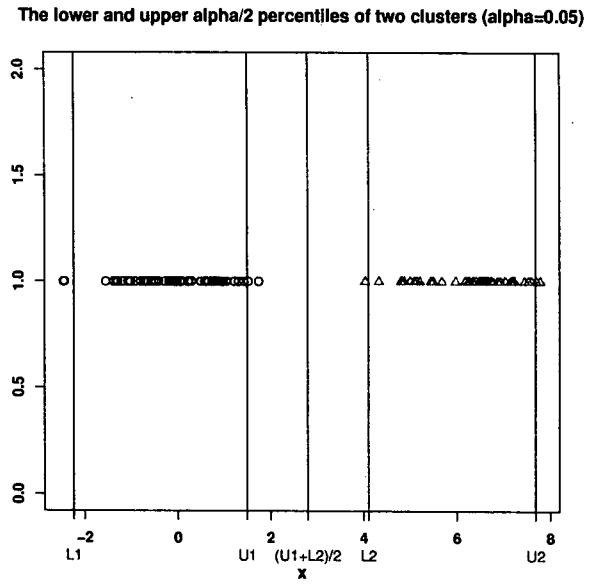


Figure 2.4: The separation index $J = L_2(\alpha/2) - U_1(\alpha/2)$ can capture the gap area between the two clusters.

α can be regarded as a tuning parameter to reflect the percentage in the one tail that might be outlying.

Based on the separation index J, we also can define a separating point $S_0 = [U_1(\alpha/2) + L_2(\alpha/2)]/2$ so that the data points on the left-hand side of S_0 are treated as from cluster 1 and data points on the right-hand side of S_0 are treated as from cluster 2.

For multivariate data, we can first find a projection direction such that the two cluster projections have the largest separation and then we use the value of the separation index for the projected data as a measure of the magnitude of the gap between the two clusters. Figure 2.5 illustrates this idea. The separating point S_0 in the projected space corresponds to a separating hyperplane in the original space.

One limitation of the separation index J defined in Formula (2.3.1) is that it does not consider the variations within the two clusters. Figure 2.6 gives an example where the value of the

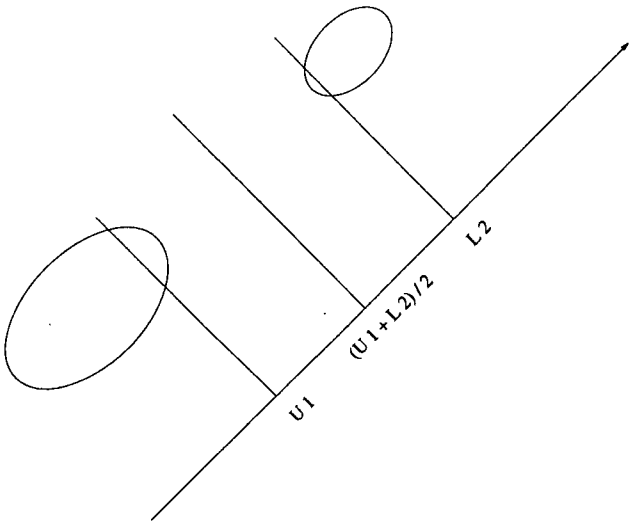


Figure 2.5: For high dimensional data, we can use the value of the separation index J for the 1-dimensional projected data as a measure of the magnitude of the gap between the two clusters.

separation index J between two clusters in the upper panel is the same as that in the lower panel, while intuitively the two clusters in the lower panel are more separated than those in the upper panel. One possible way to overcome this limitation is to normalize the separation index J with

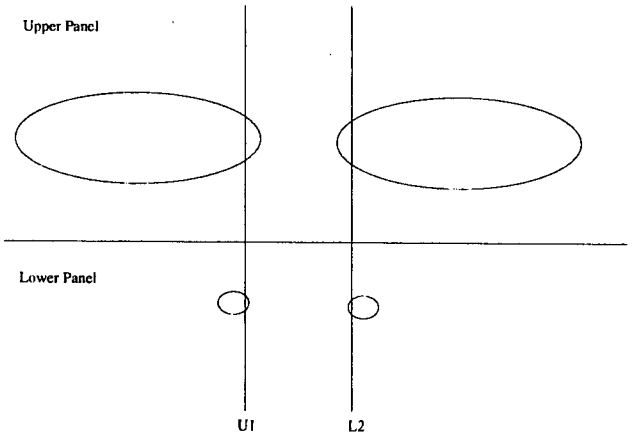


Figure 2.6: An example illustrates a limitation of the separation index J.

$U_2(\alpha/2) - L_1(\alpha/2)$, that is,

$$J^* = \frac{L_2(\alpha/2) - U_1(\alpha/2)}{U_2(\alpha/2) - L_1(\alpha/2)}. \tag{2.3.2}$$

The tuning parameter α reflects the percentage in the two tails that might be outlying (see Figure 2.7). That is, we don't want the few points that might be in the middle between "boundaries"

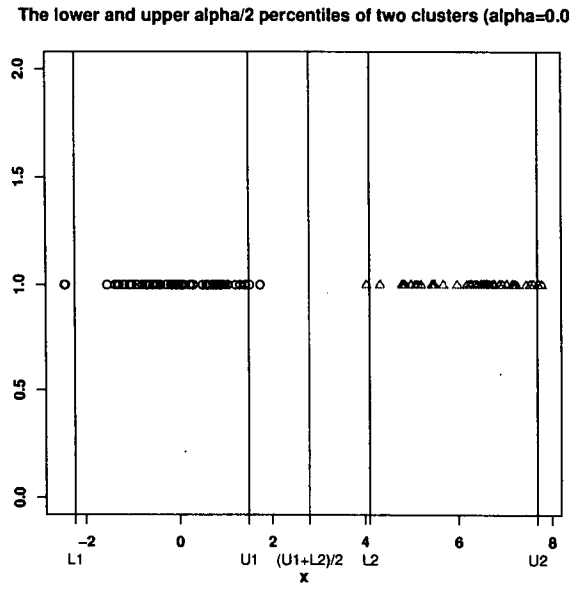
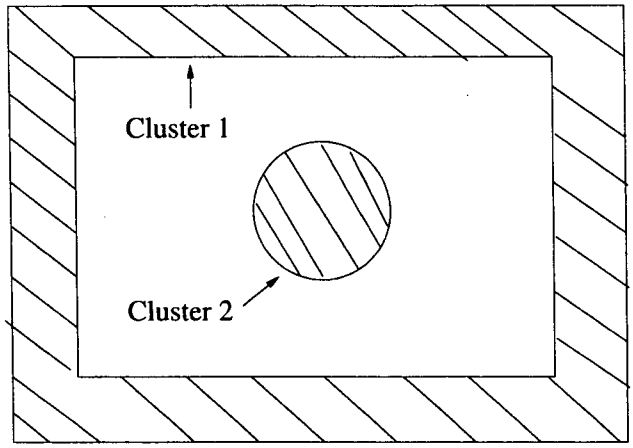


Figure 2.7: The normalized separation index $J^* = [L_2(\alpha/2) - U_1(\alpha/2)]/[U_2(\alpha/2) - L_1(\alpha/2)]$ takes account of both the external isolation and within-cluster variation. The tuning parameter α reflects the percentage in the two tails that might be outlying.

of two clusters or points that are extremely far from the separating hyperplane to have an influence on the separation index. For example, $\alpha = 0.01$ means that for each cluster, the separation index allows for $100(\alpha/2)\% = 0.5\%$ of the projected data points to lie in the extreme two tails away from the hyperplane or may be "overlapping" with the other cluster in the middle.

The implicit assumption for the separation indexes J and J^* is that clusters are convex in shape. For example, the separation indexes J and J^* are not suitable for a case like Figure 2.8 since cluster 1 is not convex. This assumption makes sense and most clustering methods explicitly or implicitly require this assumption. Clustering methods typically find clusters that are separated by hyperplanes.

To use the separation index J^* to validate partitions, we need to assume that there are gap or sparse areas among clusters although we can calculate J^* for heavily overlapped cluster structures with meaningful interpretations. For example, for the data points on the right panel of Figure 2.3, we could not know if there is only one cluster or there exist two heavily overlapped clusters, without other information.



Cluster 1 is not convex–shaped

Figure 2.8: An example that the separation indexes J and J^* would not work.

2.3.2 Optimal Projection Direction

To determine an "optimal" way of choosing the projection direction $\mathbf{a}$ when calculating the separation index J^*, we use a population version of (2.3.2). Suppose cluster 1 and cluster 2 are realizations of random samples from distributions F_1, F_2 respectively. Let $\mathbf{X}_1 \sim F_1$, $\mathbf{X}_2 \sim F_2$. For a vector $\mathbf{a}$, let G_1, G_2 be the univariate distributions of $\mathbf{a}^T\mathbf{X}_1, \mathbf{a}^T\mathbf{X}_2$. Suppose the sign of $\mathbf{a}$ is such that $\mathbf{a}^T\mathbf{X}_1 < \mathbf{a}^T\mathbf{X}_2$ with high probability (this is the theoretical equivalent of a good separating hyperplane direction). The population version of (2.3.2) is

$$\frac{G_2^{-1}(\alpha/2) - G_1^{-1}(1-\alpha/2)}{G_2^{-1}(1-\alpha/2) - G_1^{-1}(\alpha/2)}, \tag{2.3.3}$$

where $G_i^{-1}(\alpha/2)$, $i = 1, 2$, are the lower quantiles of G_i for $i = 1, 2$.

If F_i corresponds to $N(\boldsymbol{\theta}_i, \boldsymbol{\Sigma}_i)$ for $i = 1, 2$, then G_i corresponds to $N(\mathbf{a}^T\boldsymbol{\theta}_i, \mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a})$ and

$$G_i^{-1}(\alpha/2) = \mathbf{a}^T\boldsymbol{\theta}_i - z_{\alpha/2}\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}}\,, \quad G_i^{-1}(1-\alpha/2) = \mathbf{a}^T\boldsymbol{\theta}_i + z_{\alpha/2}\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}}\,.$$

(2.3.3) becomes

$$J^*(\mathbf{a}) = \frac{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - z_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + z_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}. \tag{2.3.4}$$

For ease of computation, we will use (2.3.4) as a means to choose the separating hyperplane or projection direction even if the true distributions are not multivariate normal. For a partition from a clustering method, $\boldsymbol{\theta}_i, \boldsymbol{\Sigma}_i$ are the sample mean vector and covariance matrix for cluster i.

One reason for using (2.3.4) as a criterion for choosing $\mathbf{a}$ is that there is a simple iterative algorithm to find the $\mathbf{a}$ that maximizes (2.3.4), and properties of this optimal $\mathbf{a}$ can be studied.

From (2.3.4), J^* satisfies $J^*(c\boldsymbol{a}) = J^*(\boldsymbol{a})$ where $c > 0$ is a constant scalar. So we only need consider the direction $\boldsymbol{a} \in A$, where

$$A = \{\mathbf{a} : \mathbf{a}^T\mathbf{a} = 1 \text{ and } \mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) \geq 0\}.$$

From (2.3.4), we also can see that $J^* \in [-1, 1)$ for all $\mathbf{a} \in A$. In fact, if $a \geq 0$ and $b > 0$, then $-1 \leq (a-b)/(a+b) < 1$ is equivalent to $a \geq 0$ and $b \geq 0$. If $\mathbf{a} \in A$, then $a = \mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) \geq 0$ and $b = z_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right) > 0$. Hence $J^* \in [-1, 1)$ for $\mathbf{a} \in A$. When $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$, then $J^* = -1$. When $L_2 = U_1$, then $J^* = 0$, i.e., the two clusters would be considered as just touching each other. It is easy to interpret the value of the separation index J^*. A negative value of the separation index indicates that two clusters are overlapping; a zero value indicates that two clusters are touching; and a positive value indicates that two clusters are separated (see Figure 2.9).

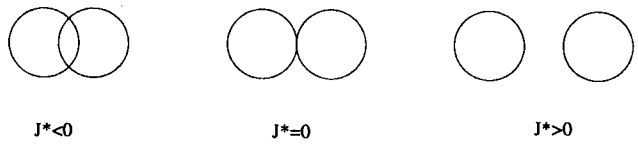


Figure 2.9: Interpretation of the value of the separation index J^*.

Denote $\mu_i = \mathbf{a}^T\boldsymbol{\theta}_i$ and $\sigma_i = \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}}$, $i = 1, 2$. If everything is unchanged except we move the two clusters farther apart along the projection direction, then $J^* \to 1$ as $\mu_2 - \mu_1 \to \infty$. In fact, J^* is a monotone increasing function of $\mu_2 - \mu_1$ if $\mu_2 - \mu_1 \geq 0$ (with α, σ_1 and σ_2 fixed). Also J^* is a monotone decreasing function of $z_{\alpha/2}(\sigma_1 + \sigma_2)$ if $\mu_2 - \mu_1$ is fixed and positive.

Theorem 2.3.1 *The $\boldsymbol{a} \in A$ that maximizes $J^*(\mathbf{a})$ in (2.3.4) satisfies*

$$c \cdot \left(\frac{\boldsymbol{\Sigma}_1}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_1\boldsymbol{a}}} + \frac{\boldsymbol{\Sigma}_2}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_2\boldsymbol{a}}}\right)\boldsymbol{a} = (\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1), \tag{2.3.5}$$

where

$$c = \frac{\boldsymbol{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_1\boldsymbol{a}} + \sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_2\boldsymbol{a}}} \tag{2.3.6}$$

[**Proof:**] Define the functions $d(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)$ and $\boldsymbol{D}(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)$ as

$$d(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2) = \sqrt{\boldsymbol{a}^T \boldsymbol{\Sigma}_1 \boldsymbol{a}} + \sqrt{\boldsymbol{a}^T \boldsymbol{\Sigma}_2 \boldsymbol{a}},$$

$$\boldsymbol{D}(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2) = \frac{\boldsymbol{\Sigma}_1}{\sqrt{\boldsymbol{a}^T \boldsymbol{\Sigma}_1 \boldsymbol{a}}} + \frac{\boldsymbol{\Sigma}_2}{\sqrt{\boldsymbol{a}^T \boldsymbol{\Sigma}_2 \boldsymbol{a}}}.$$

By taking the first derivative of J^* in (2.3.4), we can get

$$\frac{\partial J^*(\boldsymbol{a})}{\partial \boldsymbol{a}} = \frac{2 z_{\alpha/2}}{\left[\boldsymbol{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + z_{\alpha/2} d(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)\right]^2} \cdot \left\{ d(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - \boldsymbol{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) \boldsymbol{D}(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2) \boldsymbol{a} \right\}. \tag{2.3.7}$$

Let $\partial J^* / \partial \boldsymbol{a} = 0$ to solve for the optimal $\mathbf{a}$. We get

$$(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) = \frac{\boldsymbol{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)}{d(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2)} \boldsymbol{D}(\boldsymbol{a}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2) \boldsymbol{a}.$$

The solution satisfying (2.3.5) corresponds to a maximum because for a vector $\boldsymbol{q}$ that is orthogonal to $\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1$, (2.3.4) becomes $J^*(\boldsymbol{q}) = -1$, indicating that $\boldsymbol{q}$ leads to a projection that does not separate the two clusters (projections of cluster 2 within the range of projections of cluster 1). □

Corollary 2.3.2 *If $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$, then the optimal projection direction satisfies*

$$\boldsymbol{a} \propto \boldsymbol{\Sigma}^{-1}(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1). \tag{2.3.8}$$

In particular, if $\boldsymbol{\Sigma} = \boldsymbol{I}$, then

$$\boldsymbol{a} \propto (\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1). \tag{2.3.9}$$

The direction $\boldsymbol{\Sigma}^{-1}(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)$ is the well known Fisher's discriminant direction.

Rewriting (2.3.5) with $\boldsymbol{a}$ by itself on the left-hand side, we propose the following fixed-point iterative algorithm to calculate the optimal direction $\boldsymbol{a}$.

1. Get an initial estimate of $\boldsymbol{a}$ from (2.3.9).
2. Normalize $\boldsymbol{a}$, i.e. $\boldsymbol{a} \leftarrow \boldsymbol{a}/\|\boldsymbol{a}\|$.
3. Set $t = 1$, $\boldsymbol{a}^{(t)} \leftarrow \boldsymbol{a}$.
4. Update $\boldsymbol{a}$ by the formula

$$\boldsymbol{a}^{(t+1)} = \boldsymbol{D}^{-1}\left(\boldsymbol{a}^{(t)}; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2\right)(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1). \tag{2.3.10}$$

5. Normalize $\boldsymbol{a}^{(t+1)}$, i.e. $\boldsymbol{a}^{(t+1)} \leftarrow \boldsymbol{a}^{(t+1)}/\|\boldsymbol{a}^{(t+1)}\|$. Stop if $\|\boldsymbol{a}^{(t+1)} - \boldsymbol{a}^{(t)}\|^2 < \epsilon$, where ϵ is a small positive number, 10^{-4} say. Otherwise, increment $t \leftarrow t+1$ and go back to step 4.

Note that we can drop the scalar c in (2.3.10) because of the constrain $\boldsymbol{a}^T\boldsymbol{a} = 1$. If the matrix $\boldsymbol{D}$ is singular, we can use the Moore-Penrose generalized inverse (Wang and Chow, 1994). It is theoretically intractable to prove the convergence for our problem. But empirical experience shows that the convergence will be achieved within a few iterations. There might exist local maximum points. So there is no guarantee the algorithm converges to a global optimum. Empirical experience shows that two groups of data points have good separation along the projection directions obtained by the algorithm.

From the equation (2.3.5), we can see that the optimal direction $\boldsymbol{a}^*$ does not depend on α and $Z_{\alpha/2}$. Hence Theorem 2.3.1 still holds for more general elliptically contoured distributions, and the separation index could be defined as

$$J^{**} = \frac{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - q_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + q_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}, \tag{2.3.11}$$

where $q_{\alpha/2}$ is the upper $\alpha/2$ percentile of the standardized univariate margin of the elliptically contoured distribution. In fact, if F_i corresponds to a elliptically contoured distribution with mean vector $\boldsymbol{\theta}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$ for $i = 1, 2$, then the lower and upper $\alpha/2$ percentiles of the univariate distributions G_i are

$$G_i^{-1}(\alpha/2) = \mathbf{a}^T\boldsymbol{\theta}_i - q_{\alpha/2}\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}}, \quad G_i^{-1}(1-\alpha/2) = \mathbf{a}^T\boldsymbol{\theta}_i + q_{\alpha/2}\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}}.$$

(2.3.3) becomes (2.3.11).

Theorem 2.3.3 *[Affine invariance:] If we make the transformation $\boldsymbol{Y}_i = A\boldsymbol{X}_i + \boldsymbol{b}$, then the value of the separation index is unchanged and $\boldsymbol{a}^*_{\boldsymbol{Y}_i} = \left(A^{-1}\right)^T \boldsymbol{a}^*_{\boldsymbol{X}_i}$, $i = 1, 2$.*

Proof: From

$$J^*(\mathbf{a}_Y) = \frac{\mathbf{a}_Y^T A(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - z_{\alpha/2}\left(\sqrt{\mathbf{a}_Y^T A\boldsymbol{\Sigma}_1 A^T\mathbf{a}_Y} + \sqrt{\mathbf{a}_Y^T A\boldsymbol{\Sigma}_2 A^T\mathbf{a}_Y}\right)}{\mathbf{a}_Y^T A(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + z_{\alpha/2}\left(\sqrt{\mathbf{a}_Y^T A\boldsymbol{\Sigma}_1 A^T\mathbf{a}_Y} + \sqrt{\mathbf{a}_Y^T A\boldsymbol{\Sigma}_2 A^T\mathbf{a}_Y}\right)},$$

we can get $J^*(\boldsymbol{a}_Y) = J^*(\boldsymbol{a}_X)$ and $\boldsymbol{a}^*_{\boldsymbol{Y}_i} = \left(A^{-1}\right)^T \boldsymbol{a}^*_{\boldsymbol{X}_i}$, $i = 1, 2$. □

2.4 Comparisons Based on the Separation Index

As we mentioned in Section 2.1, we try to answer the following questions:

Question 1: Given the number of clusters, how do we compare the partitions obtained by applying different clustering methods to the same data set.

Question 2: Given a clustering method, how do we check if the specified number of clusters is appropriate or not?

Question 3: How do we compare different partitions (both the number of clusters and the partitions may be different)?

In this section, we propose a solution for these three questions based on the pairwise separation index matrices $\zeta_\ell = (J_{ij}^{*\ell})_{k_\ell \times k_\ell}$, $\ell = 1, \ldots, M$, where M is the number of partitions considered, k_ℓ is the number of clusters in the ℓ-th partition, $J_{ij}^{*\ell}$ is the separation index between cluster i and j $(i \neq j)$ in the ℓ-th partition and J_{ii}^{*} is defined as -1. This solution is based on the following criteria:

Criterion 1: Given a specified number of clusters, the t-th partition would be the best if its minimum separation index value is the largest. That is,

$$\min_{i<j} J_{ij}^{*t} = \max_{\ell=1,\ldots,M} \min_{i<j} J_{ij}^{*\ell}.$$

Criterion 2: Given a clustering method, if the minimum separation index value $\min_{i<j} J_{ij}^{*t}$ of the t-th partition is negative or close to zero, then the corresponding number of clusters might be under- or over-specified. Figure 2.10 illustrates the case where the number of clusters is under-specified and the minimum separation index would be negative.

Criterion 3: Suppose that we want to compare two different partitions P_1 and P_2. If the minimum separation index values, $\min_{i<j} J_{ij}^{*\ell}$, $\ell = 1, 2$, of the two partitions have the same sign, then P_1 is better than P_2 when the average separation index value $(\sum_{i<j} J_{ij}^{*1}/(k_1(k_1-1)/2))$ of P_1 is larger than that $(\sum_{i<j} J_{ij}^{*2}/(k_2(k_2-1)/2))$ of P_2. Otherwise, P_1 is better than P_2 when the minimum separation index value of P_1 is positive.

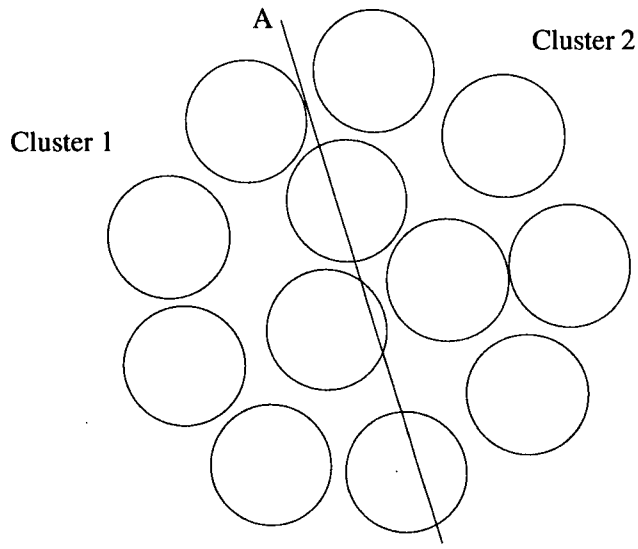


Figure 2.10: The line A splits 12 clusters into two subclusters. The number 2 of clusters is under-specified and the value of the separation index J^* would be negative.

Criterion 3 uses average instead of max minimum separation index to avoid cases like that shown in Figure 2.11. The four circles in Figure 2.11 illustrate 4 clusters. The left panel shows a 2-cluster partition while the right panel shows a 4-cluster partition. The minimum separation index of the 2-cluster partition is larger than that of the 4-cluster partition. However the 4-cluster partition seems better than that of 2-cluster partition. By using average separation index, we might avoid this problem.

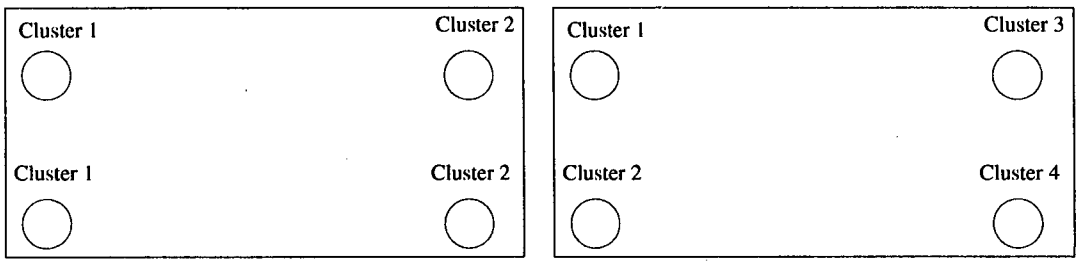


Figure 2.11: Using average separation index is better than using max minimum separation index if the numbers of clusters of two partitions are different.

The above set of criteria is only one possible way to use the separation index matrix to answer Questions 1, 2 and 3. Other functions of the separation index matrix could be used. For example, instead of using the average separation index value, we can use penalized average separation index value to take into account the number of clusters.

We use a simulated data set and a real one to illustrate the separation indexes and the criteria. The simulated data set is in two dimensions for ease of illustration, and the real data set has 13 dimensions.

2.4.1 A Simulated Data Set

In the simulated data set, there are two clusters generated from two bivariate normal distributions with mean vectors and covariance matrices

$$\boldsymbol{\theta}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 1.86 & 2.65 \\ 2.65 & 9.14 \end{pmatrix}, \quad \boldsymbol{\theta}_2 = \begin{pmatrix} 7 \\ 2 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 3.62 & 1.90 \\ 1.90 & 2.38 \end{pmatrix}.$$

Each cluster has 200 data points.

The left panel and right panel of Figure 2.12 show 2-cluster partitions of the data set, obtained by `clara` and `Mclust` respectively, which we denote by P_{p1} and P_{m1} respectively. We can see that the two clusters are quite close and detectable by eye. The separation index values (based on cluster mean vectors and covariance matrices) of the partitions P_{p1} and P_{m1} are $J_{12}^{*p} = 8 \times 10^{-5}$ and $J_{12}^{*m} = 0.12$ respectively (with $\alpha = 0.05$). $J_{12}^{*p} = 0.00$ indicates that the two clusters are touching and $J_{12}^{*m} = 0.12$ indicates that the two clusters are separated. According to **Criterion** 1, the partition P_{m1} is better than P_{p1}. By visualizing the two partitions from Figure 2.12, we can see that P_{m1} is better P_{p1}.

Suppose that we use `Mclust` to get three different partitions with the number of clusters 2, 3, and 4 respectively. The three partitions are shown in Figure 2.13. Denote these three partitions as $P_m^{(2)}$, $P_m^{(3)}$, and $P_m^{(4)}$ respectively. The separation index matrices of these partitions are:

$$\zeta_2^m = \begin{pmatrix} -1.00 & 0.12 \\ 0.12 & -1.00 \end{pmatrix}, \zeta_3^m = \begin{pmatrix} -1.00 & -0.11 & 0.11 \\ -0.11 & -1.00 & 0.28 \\ 0.11 & 0.28 & -1.00 \end{pmatrix}, \zeta_4^m = \begin{pmatrix} -1.00 & -0.12 & 0.33 & 0.12 \\ -0.12 & -1.00 & 0.41 & 0.28 \\ 0.33 & 0.41 & -1.00 & -0.16 \\ 0.12 & 0.28 & -0.16 & -1.00 \end{pmatrix},$$

respectively. According to **Criterion** 2, 3-cluster and 4-cluster clusters are not as good as the 2-cluster partition.

Note that usually a clustering method will not produce overlapping subclusters if we use the clustering method to split a compact cluster. Unlike touching clusters, these subclusters have high density on the boundaries between clusters. That is, the distributions of these subclusters are skewed to the boundaries between clusters and hence the cluster centers are closer to the boundaries

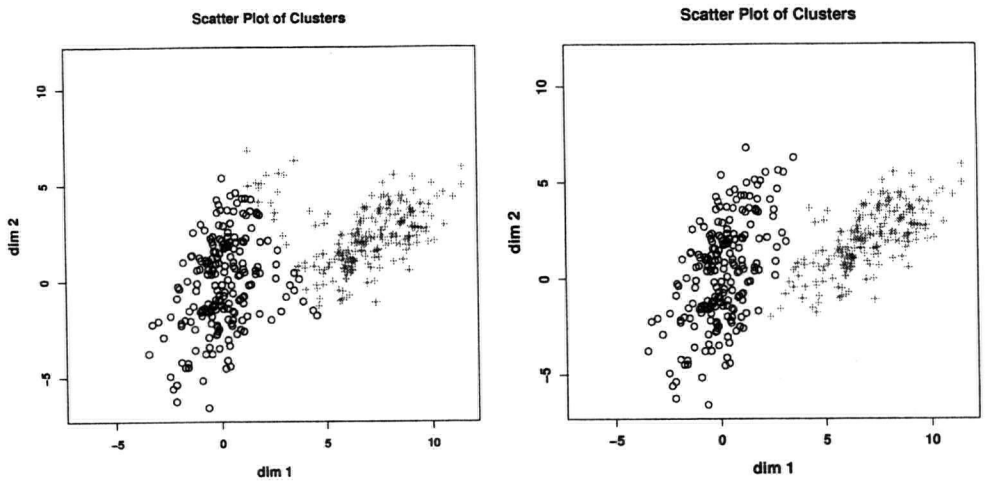


Figure 2.12: Two-cluster partitions of the simulated data set. Left panel: By `clara`. Right Panel: By `Mclust`. The circles are for points from cluster 1 and the +'s are for points from cluster 2.

between clusters. Therefore when we use the formula (2.3.4) to calculate the separation indexes, the values are negative instead of close to zero. Although, the subclusters can not be normally distributed in this case, it is an advantage to use the formula (2.3.4) instead of the formula (2.3.3). In this case, the formula (2.3.3) produces small positive separation index values and makes it difficult to distinguish if the subclusters are touching or if the number of clusters is over-specified.

If we want to compare the partitions P_{p1} and $P_m^{(3)}$, we can apply **Criterion** 3. The average separation index values for the partitions P_{p1} and $P_m^{(3)}$ are 0.00 and 0.09 respectively. Hence according to **Criterion** 3, $P_m^{(3)}$ is better than P_{p1}. By comparing the left panel of Figure 2.12 and the middle panel of Figure 2.13, we see that in the partition P_{p1} some points in the true cluster 2 are mistakenly grouped into cluster 1 and some points in cluster 1 are mistakenly grouped into cluster 2. Although the partition $P_m^{(3)}$ divides cluster 1 into two subclusters, there seems no obvious misclassification. So intuitively, $P_m^{(3)}$ is better than P_{p1}.

2.4.2 A Real Data Set

For the simulated data set in the previous subsection, we can easily visualize the cluster structure since clusters are in the two-dimensional space. Real data sets are usually high-dimensional and we may not be able to visualize the cluster structure from pairwise scatterplots. The importance of the separation index will be clear in these cases. To illustrate this, we use the wine data set,

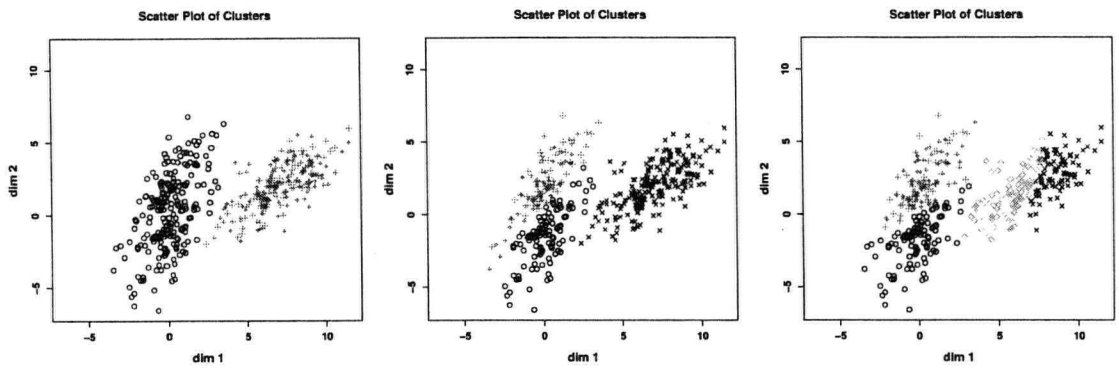


Figure 2.13: Three partitions obtained by `Mclust` for the simulated data set. Left Panel: 2-cluster partition; middle panel: 3-cluster partition; right panel: 4-cluster partition. The symbols "$\circ$", "$+$", "$\times$", and "$\diamond$" are for points from cluster 1, 2, 3, and 4 respectively.

available from the UCI Machine Learning site (Blake and Merz, 1998). There are 178 observations, 13 variables and 3 classes which represent 13 different chemical constituents of 178 Italian wines derived from 3 different cultivars.

The pairwise scatterplots of the 13 variables do not show obvious 3-cluster or 3-class structure. The pairwise scatterplots of 4 randomly selected variables (variables 2, 6, 11, and 13) are shown in Figure 2.14. Figure 2.14 also shows that the ranges of the variables are quite different. For variable 11, the range is $[0.48, 1.71]$, while the range of variable 13 is $[278, 1680]$. So we standardize the 13 variables before we do further analysis so that the variance of each variable is 1.

Given that the number of classes is 3, we obtain two partitions by using `clara` and `Mclust` respectively. Denote the two partitions as $P_{p,w}$ and $P_{m,w}$. The separation index matrices for partitions $P_{p,w}$ and $P_{m,w}$ (with $\alpha = 0.05$) are

$$\zeta_w^p = \begin{pmatrix} -1.00 & 0.04 & 0.41 \\ 0.04 & -1.00 & 0.14 \\ 0.41 & 0.14 & -1.00 \end{pmatrix}, \quad \zeta_w^m = \begin{pmatrix} -1.00 & 0.10 & 0.44 \\ 0.10 & -1.00 & 0.14 \\ 0.44 & 0.14 & -1.00 \end{pmatrix}.$$

Since the minimum separation index value 0.04 (except diagonal elements) of ζ_w^p is smaller than that of ζ_w^m (0.10), we think $P_{m,w}$ is better than $P_{p,w}$ according to **Criterion** 1. By comparing $P_{p,w}$ and $P_{m,w}$ with the true partition of the wine data, we find that the misclassification rates of $P_{p,w}$ and $P_{m,w}$ are 11 and 5 respectively. This demonstrates that the comparison result obtained by applying **Criterion** 1 is good.

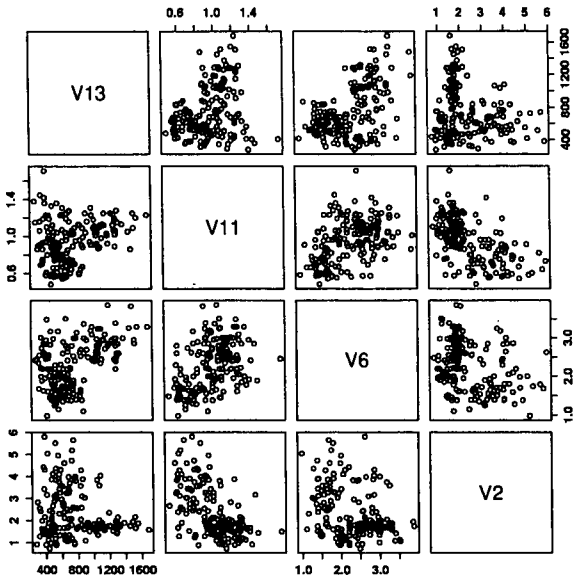


Figure 2.14: The pairwise scatter plot of 4 randomly selected variables (variables 2, 6, 11 and 13) for the wine data set.

For the clustering method `clara` with 2 to 4 specified clusters, the resulting matrices of separation indexes (with $\alpha = 0.05$) are:

$$\begin{pmatrix} -1 & 0.06 \\ 0.06 & -1 \end{pmatrix}, \quad \begin{pmatrix} -1 & 0.04 & 0.41 \\ 0.04 & -1 & 0.14 \\ 0.41 & 0.14 & -1 \end{pmatrix}, \quad \begin{pmatrix} -1 & -0.13 & 0.29 & 0.46 \\ -0.13 & -1 & 0.03 & 0.39 \\ 0.29 & 0.03 & -1 & 0.15 \\ 0.46 & 0.39 & 0.15 & -1 \end{pmatrix}$$

The numbers close to zero or less than zero in the 4-cluster partition suggest that choosing $k_0 = 4$ was too large (but when clustering data with known classes, there is always the possibility that one of the classes will show up as more than one cluster). The results for the 2-cluster and 3-cluster partitions suggest that there are 2 groups that are "close" and a third group is farther away.

2.5 Low Dimensional Visualization

The separation index matrix is a summary of closeness among clusters. If all indexes are large, then we know clusters are quite far apart from each other. However, we might have trouble understanding the cluster structure from the separation index matrix if some indexes are close to zero. In this

case, it is useful to produce a 1-, or 2-dimensional projection, such as scatter plot of the first two principal components, so that we can view the cluster structure.

In this section, we propose a new method to produce low dimensional visualization of clusters.

Since we are interested in checking the appropriateness of a partition, we only need to study the pairs of clusters which we are interested in, e.g. the nearest two clusters. It is natural to project data points of the two clusters along the optimal projection direction $\boldsymbol{a}^*$. If we want a two-dimensional projection, then we can find the second direction along which the projected data points on the hyperplane orthogonal to the direction $\boldsymbol{a}^*$ have the largest separation. Figure 2.15 shows an example of this kind of one- and two-dimensional visualization. The data set is the simulated

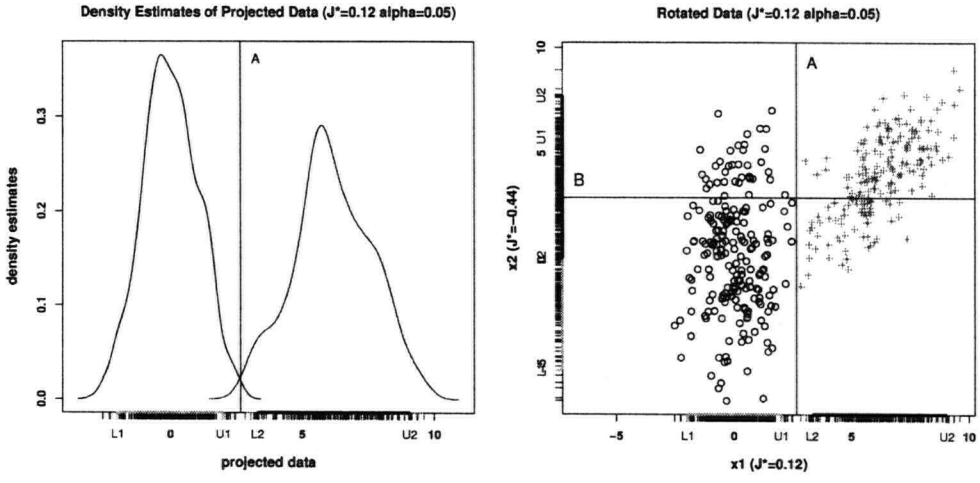


Figure 2.15: Left panel: 1-dimensional visualization of the simulated data set. Right panel: 2-dimensional visualization of the simulated data sets. The 2-cluster partition is obtained by `Mclust`.

data set used in Section 2.4. The 2-cluster partition is obtained by `Mclust`. The ticks along the axes show the distributions/densities of the two clusters along the projection directions. L_i and U_i, $i = 1, 2$, are also labeled on the axes. The separating points A and B are also marked in the plots. For the one-dimensional visualization, kernel density estimates of the two projected clusters are also shown in the plot to indicate the concentrations and variations of the two projected clusters. Both the one- and two-dimensional visualizations indicate that there exists a sparse area between the two clusters obtained by `Mclust` for the simulated data set.

Figure 2.16 and Figure 2.17 respectively show the pairwise one- and two-dimensional vi-

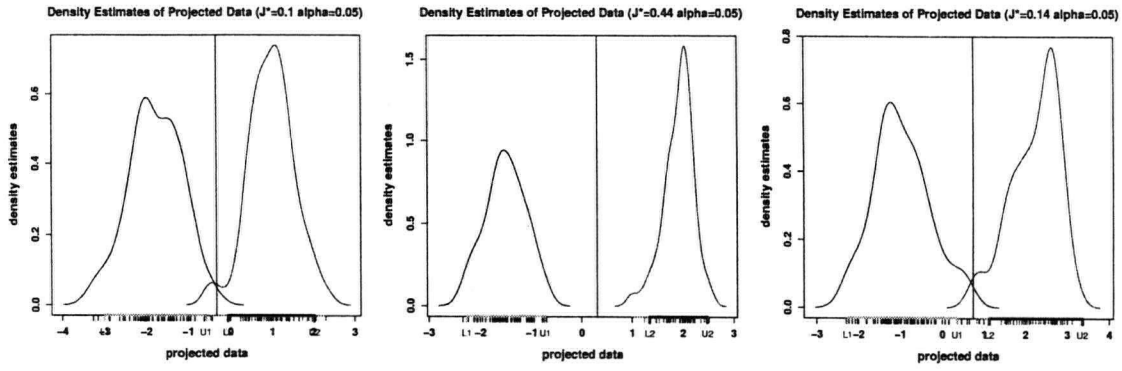


Figure 2.16: Pairwise 1-dimensional visualizations of the 3-cluster partition obtained by `Mclust` for the wine data. Left panel: cluster 1 vs cluster 2. Middle panel: cluster 1 vs cluster 3. Right panel: cluster 2 vs cluster 3.

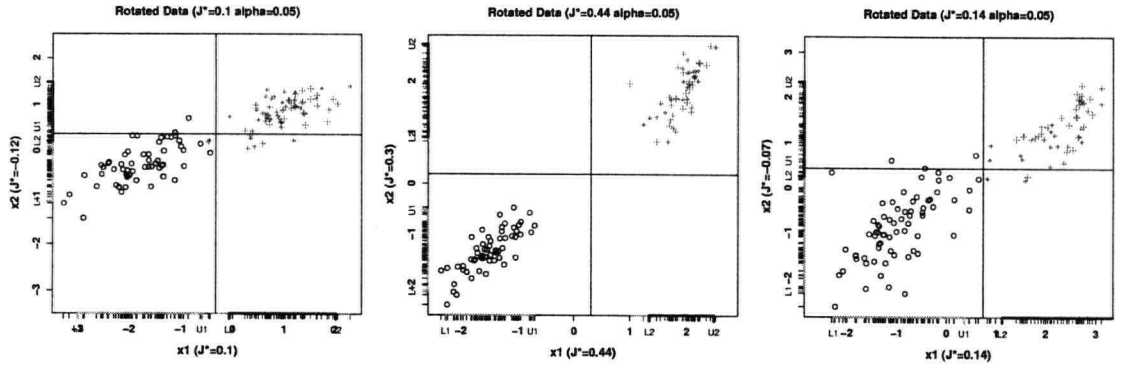


Figure 2.17: Pairwise 2-dimensional visualizations of the 3-cluster partition obtained by `Mclust` for the wine data. Left panel: cluster 1 vs cluster 2. Middle panel: cluster 1 vs cluster 3. Right panel: cluster 2 vs cluster 3.

sualizations of the three clusters obtained by `Mclust` for the wine data set. The one- and two-dimensional visualizations clearly suggest that clusters 1 and 2 and clusters 2 and 3 are separated but "close" to each other; clusters 1 and 3 are well-separated.

2.6 Partial Membership

The separation indexes for a cluster partition indicate how far apart the clusters are. If some clusters are close to each other, then the points near the boundaries between clusters might be "misclassified" in the sense that they partly fit with more than one cluster. We use the idea of partial membership to indicate points which are close to the boundaries among two or more clusters.

If there are many points at these boundaries, then the boundaries between clusters are more vague.

For partial membership, each point is assigned a value between 0 and 1 for each cluster, with a total sum of 1 for the values. The vague points are those which do not have a value of 1 for one cluster (and 0 for the rest).

There are a few fuzzy clustering methods in the literature to determine the "optimal" partial memberships of the data, but they do not necessarily have the property of assigning partial membership only to points at boundaries of clusters. The fuzzy c-means algorithm is a classic fuzzy clustering algorithm (Bezdek 1981; Höppner et al. 1999). It obtains the partial memberships and the partition of data points simultaneously by iteratively solving the following minimization problem:

$$\min_{H \times V} \sum_{k=1}^{n} \sum_{i=1}^{k_0} [h_i(\mathbf{y}_k)]^m \, \|\mathbf{y}_k - \boldsymbol{v}_i\|^2,$$

such that

$$h_i(\mathbf{y}) \in [0,1], \quad \sum_{i=1}^{k_0} h_i(\mathbf{y}) = 1.$$

where k_0 is the number of clusters, n is the number of data points, m is a real number larger than 1, $h_i(\mathbf{y})$ is the membership function of cluster i for the data point $\mathbf{y}$, $H = \{h_1, \ldots, h_{k_0}\}$ is the set of fuzzy membership functions,

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{n} [h_i(\mathbf{y}_k)]^m \mathbf{y}_k}{\sum_{k=1}^{n} [h_i(\mathbf{y}_k)]^m}$$

is the fuzzy mean of the i-th cluster, and $V = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{k_0}\}$ is the set of fuzzy cluster means.

Kaufman and Rousseeuw (1990, page 190) proposed a fuzzy clustering algorithm, called **fanny**, which has the same form of the objective function (with $m = 2$) as that of the fuzzy c-means algorithm except that **fanny** uses distance instead of squared distance to measure the difference between two data points.

One desired property of partial membership is that only points near boundaries between clusters have membership values that are not equal to 0 or 1. In the special case of two clusters in one-dimensional space, the larger the variable value is, the smaller should be its partial membership for belonging to cluster 1 if cluster 1 is to the left of cluster 2. Both the fuzzy c-means algorithm and fanny do not have this property. Figure 2.18 shows a plot of partial membership values for cluster 1 versus the data points in one-dimensional space. The partial memberships are obtained by **fanny**. The two clusters are generated from two univariate normal distributions N (0, 1) and N (4, 1)

respectively. Each cluster has 200 data points. The memberships of cluster 1 of the right-end data points increase as data points move further away from cluster 1, and this does not match intuition. Note that convexity is implicitly assumed for this desired property, while fuzzy clustering methods such as fuzzy c-means and `fanny` do not require this assumption.

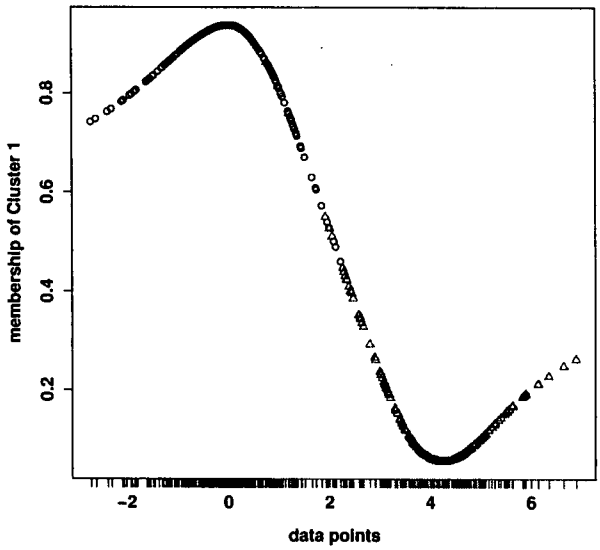


Figure 2.18: Plot of partial memberships of cluster 1 versus the data points. The circles indicate cluster 1 while the triangles indicate cluster 2. The ticks along the axes indicate the distributions of two clusters. The two clusters are generated from the univariate normal distributions $N(0,1)$ and $N(4,1)$ respectively. Each cluster has 200 data points. The memberships are obtained by `fanny`.

In this section, we propose a two-step method to assign partial memberships. In the first step, we obtain a "hard" partition from a clustering method. Then for each pair of clusters, we project them along the optimal projection direction described in Section 2.3 and determine a partial membership for the pair. Then the membership values are reweighted based on all pairs of clusters to obtain the overall partial memberships. For a given pair of clusters, points closer to the "separating" hyperplane corresponding to the optimal projection will be assigned membership values that are closer to 0.5, and points far from the hyperplane will be assigned a membership value of 1 for the cluster it was found to be in.

In discriminant analysis, if the data were a mixture of two densities (classes):

$$f(\mathbf{y}) = \pi_1 f_1(\mathbf{y}) + \pi_2 f_2(\mathbf{y}), \quad \pi_1 + \pi_2 = 1, \quad \pi_1 \geq 0, \pi_2 \geq 0,$$

then given a future $\mathbf{y}$ value, the membership probabilities for the two classes are

$$h_i(\mathbf{y}) = \frac{\pi_i f_i(\mathbf{y})}{\sum_{j=1}^{2} \pi_j f_j(\mathbf{y})}, \quad i = 1, 2. \tag{4.1}$$

We think of the one-dimensional projections from clusters i_1, i_2 as being the realization of the mixture of two univariate distributions with density:

$$f(y) = [\pi_{i_1} f_{i_1:i_2}(y) + \pi_{i_2} f_{i_2:i_1}(y)]/[\pi_{i_1} + \pi_{i_2}], \quad \pi_{i_1} + \pi_{i_2} = 1, \quad \pi_{i_1} \geq 0, \pi_{i_2} \geq 0,$$

where π_i is the relative frequency for cluster i, and $f_{i_a,i_b}(y)$ is the density of the projections for cluster i_a based on the projection direction for the cluster pairs indexed by i_a, i_b. Then we could have a pairwise membership similar to (4.1). To get actual numbers, we need values of $f_{i_1:i_2}(y), f_{i_2:i_1}(y)$ based on the data. Since we mainly are interested in identifying vague points, we will use univariate normal densities for $f_{i_1:i_2}(y), f_{i_2:i_1}(y)$ based on the sample means and variances of the projections, even though sometimes this will not be a good approximation.

Suppose the mean vector and covariance matrix of cluster i for a "hard" partition are $\boldsymbol{\theta}_i$ and $\boldsymbol{\Sigma}_i$, and the projection direction for cluster i_1 vs i_2 is $\mathbf{a}_{i_1,i_2}$, for $i_1 \neq i_2$. For a point $\mathbf{y}$ in cluster i, define the pairwise partial membership for cluster j, as

$$h_j^*(\mathbf{y}) = \begin{cases} \dfrac{\pi_j \phi\left(\mathbf{a}_{ij}^T(\mathbf{y}-\boldsymbol{\theta}_j)/\sqrt{\mathbf{a}_{ij}^T \boldsymbol{\Sigma}_j \mathbf{a}_{ij}}\right)}{\pi_i \phi\left(\mathbf{a}_{ij}^T(\mathbf{y}-\boldsymbol{\theta}_i)/\sqrt{\mathbf{a}_{ij}^T \boldsymbol{\Sigma}_i \mathbf{a}_{ij}}\right) + \pi_j \phi\left(\mathbf{a}_{ij}^T(\mathbf{y}-\boldsymbol{\theta}_j)/\sqrt{\mathbf{a}_{ij}^T \boldsymbol{\Sigma}_j \mathbf{a}_{ij}}\right)}, & j \neq i \\ \frac{1}{k_0-1} \sum_{\ell \neq i} [1 - h_\ell^*(\mathbf{y})], & j = i, \end{cases}$$

where ϕ is the density function of the standard normal distribution and $h_i^*(\mathbf{y})$ is interpreted as the average amount assigned to cluster i in the pairwise comparisons. When all k_0 clusters are considered, these are revised to

$$h_j(\mathbf{y}) = \frac{h_j^*(\mathbf{y})}{\sum_{l=1}^{k_0} h_l^*(\mathbf{y})}. \tag{2.6.1}$$

For $\mathbf{y}$ in cluster i, then one should have $h_i(\mathbf{y}) > h_j(\mathbf{y})$, $j \neq i$.

If the point $\mathbf{y}$ is in cluster i_1 and near the boundary of cluster i_2, and far from the other clusters, then one would have $h_j^*(\mathbf{y}) \approx 0$, $j \neq i_1, i_2$, and $h_{i_1}(\mathbf{y}) \approx h_{i_1}^*(\mathbf{y})$, $h_{i_2}(\mathbf{y}) \approx h_{i_2}^*(\mathbf{y})$.

As a numerical example, suppose $k_0 = 3$, and $\mathbf{y}$ is in cluster 1, and near the boundaries of clusters 2 and 3. If $h_2^*(\mathbf{y}) = 0.4$, $h_3^*(\mathbf{y}) = 0.3$, then the membership values are $h_1(\mathbf{y}) = 0.65/1.35$, $h_2(\mathbf{y}) = 0.4/1.35$, $h_3(\mathbf{y}) = 0.3/1.35$.

Because of the approximation with normal densities (more generally one could use some density estimation methods), and we only need a rough idea of which points are near boundaries of clusters, we consider a point to be vague if

$$\max_j h_j(\mathbf{y}) \leq c$$

where c might be around 0.9.

The partial memberships obtained by Formula (2.6.1) have the desired property: points closer to the "separating" hyperplane corresponding to the optimal projection will be assigned membership values that are closer to 0.5, and points far from the hyperplane will be assigned a membership value of 1 for the cluster it was found to be in. For the simulated data set described at the beginning of this section, the partial memberships obtained by Formula 2.6.1 are shown in Figure 2.19.

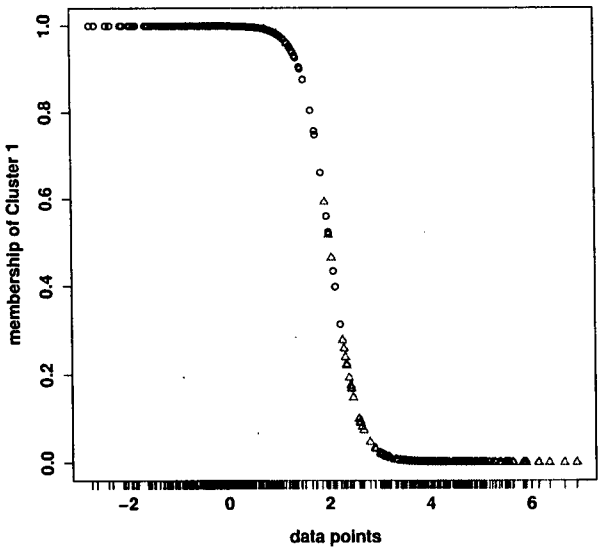


Figure 2.19: Plot of partial memberships of cluster 1 versus the data points. The circles indicate cluster 1 while the triangles indicate cluster 2. The ticks along the axes indicate the distributions of two clusters. The partial memberships are obtained by the two-step method we propose.

Note that the two-step method we propose is different from some existing fuzzy clustering methods in that fuzzy clustering simultaneously assigns partial cluster memberships and finds cluster localization by solving an optimization problem, while the membership assignment in our

two-step method is not optimal according to a given criterion. Our two-step method is a direct approach to assign partial membership while the fuzzy clustering methods like fuzzy c-means use an indirect approach.

In the following we use the two examples in the previous section to illustrate the performance of the partial membership function we proposed.

Figure 2.20 shows the partial memberships of cluster 1 of the two clusters obtained by `clara` for the simulated data set described in Subsection 2.4.1. The data points shown in the plot are not on the original 2-dimensional space. Instead, they are on the projected 2-dimensional space based on the method described in Appendix A. The points with membership values which are between 0 and 1 represent the vague points at the shared boundary of the clusters.

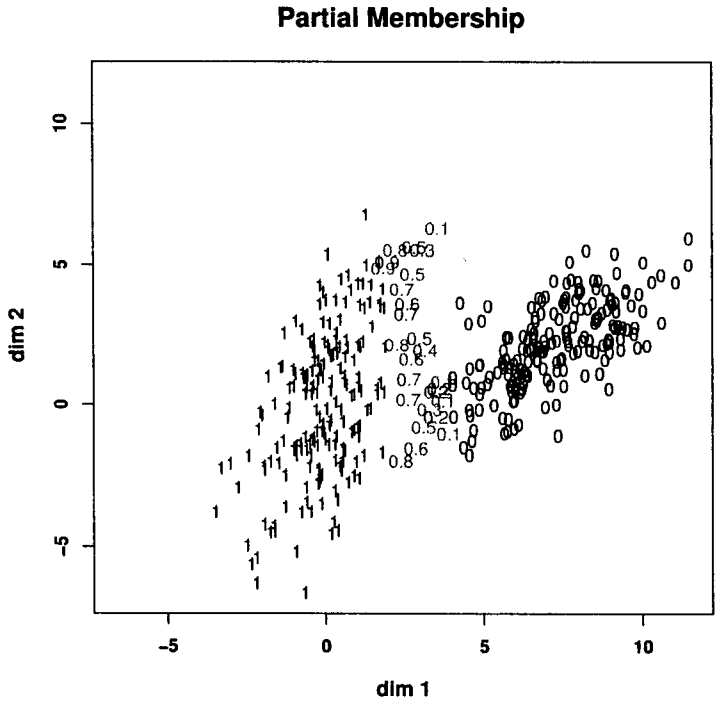


Figure 2.20: Partial membership for the simulated data set. Partition is obtained by `clara`.

The partial memberships of clusters 1, 2, and 3 obtained by `clara` for the wine data set are shown in Figure 2.21. Cluster 3 is well separated from cluster 1 and 2, so there are only two data points in cluster 3 whose memberships are between 0.1 and 0.9.

The true classes of the wine data set are known. To illustrate the relationship of vagueness

and misclassification, we check which data points are misclassified in the partition obtained by clara. We found that some of vague points are misclassified, while some are not. For example, data points $\mathbf{y}_{66}$, $\mathbf{y}_{72}$, $\mathbf{y}_{84}$, $\mathbf{y}_{99}$, and $\mathbf{y}_{111}$ are all vague points and misclassified points. Data points $\mathbf{y}_{66}$, $\mathbf{y}_{72}$ and $\mathbf{y}_{99}$, $\mathbf{y}_{111}$ belong to class 2 but are in `clara` cluster 1. Data point $\mathbf{y}_{84}$ belongs to class 2 but are in `clara` cluster 3. The memberships of cluster 1 for data points $\mathbf{y}_{66}$, $\mathbf{y}_{72}$ $\mathbf{y}_{99}$, and $\mathbf{y}_{111}$ are 0.73, 0.66, 0.84 and 0.65 respectively. The membership of cluster 3 for data point $\mathbf{y}_{84}$ is 0.65.

We can see that our approach reasonably captures the vague points at shared cluster boundaries. The partial membership values for these vague points are reasonable.

2.7 Discussion

We have proposed a separation index between pair of clusters to measure the separation distance between them; this applies to any pair of clusters obtained from a partition using a clustering method. A separation index matrix is used as a summary of a partition, and partitions from different clustering methods can be compared based on their separation index matrices.

The population version of Formula (2.3.4) is used as motivation for the method. For data and a partition obtained from a clustering method, the cluster sample covariance matrices are used. We view the separation indexes as summary of a partition, and they are not necessarily estimates of population parameters.

Based on the projections associated with the separation indexes we have also proposed a two-step method to obtain partial membership values for points which are at the boundaries among clusters.

In addition, the separation index we propose have many other applications. We can develop a sequential clustering algorithm by merging and splitting clusters to simultaneously estimate the number of clusters and obtain a "hard" partition; for examples two clusters with near zero separation index could be merged, and a large cluster could be split into two parts and the separation index of the subclusters can be computed. This application is implemented in Chapter 4.

It might also be possible to obtain a partition by maximizing the minimum separation index. For example, the stopping rule of the `kmeans` algorithm is that we stop reallocate data points until the average within-cluster variation no longer decreases. We might modify it so that we stop

reallocate data points until the minimum separation index among clusters no longer increases. This can be a future research topic.

Moreover, we can develop a method to generate random clusters with different amounts of separation and with arbitrary covariance matrices. This application is implemented in Chapter 3.

The optimal projection direction associated to the separation index can be used to do 2-class discriminant analysis. Actually, the optimal projection direction generalizes the famous Fisher's discriminant direction to allow different class covariance matrices.

In classification, researchers (e.g. Rosenblatt 1958) have proposed a separating hyperplane to distinguish two groups. The aim is to minimize the distance of misclassified points to the separating hyperplane. This kind of separating hyperplane is not unique. So researchers such as Vapnik (1996) find a hyperplane that cuts two groups and maximize the minimum distance over all points to the hyperplane. Clustering is generally used without knowledge of true classes, so our optimal separating hyperplane tries to maximize the difference in projections of two clusters, rather than minimize misclassification. Moreover, Vapnik's (1996) method would not work for two classes which are not linear separable. The separating hyperplane associated to the separation index J^* can work for linear non-separable cases.

Support vector machine generalizes Vapnik's (1996) separating hyperplane idea to handle linear non-separable cases and to separate non-linear boundaries by enlarging the feature space (Hastie et al. 2001). A possible future research topic is to generalize the separation index J^* so that it can measure the magnitude of the gap or sparse area between two non-convex-shaped clusters.

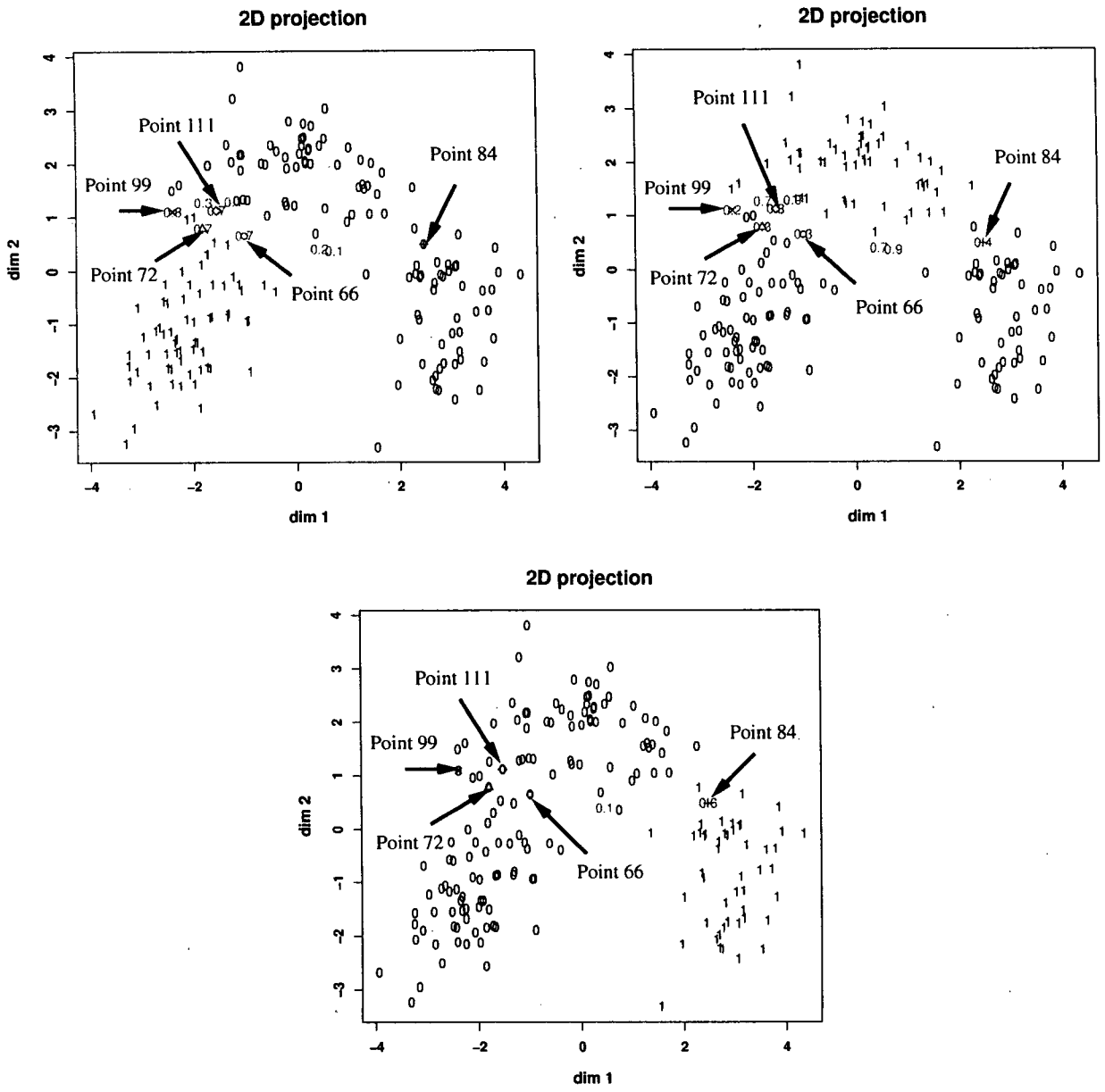


Figure 2.21: The membership scatterplots of the scaled and centralized wine data set in the projected 2-dimensional space described in Appendix A. The "hard" partition is obtained by `clara`. The top-left, top-right and bottom panels show membership for clusters $1, 2$, and 3 respectively.

Chapter 3

Generation of Random Clusters

3.1 Introduction

Many clustering methods have been proposed and new methods continue to appear in the cluster analysis literature. However, some methods have an ad hoc nature and hence it is difficult to study their theoretical properties. Some methods do have some nice theoretical properties, but need some specific assumptions. It is usually difficult to validate these assumptions. Even if these assumptions can be easily checked, we still need a way to check the performances of these methods when the assumptions are not satisfied. Hence numerical evaluation techniques are needed.

One way to numerically evaluate the performances of clustering methods is to apply the methods to real data sets whose class or cluster structures are known. The greater the agreement is between the true partition and the partition obtained by the clustering method, the better performance the clustering method has. However it is usually difficult to obtain real data sets with known cluster structures. Moreover, real data sets may contain noisy variables, outliers, measurement errors, and/or missing values. These problems are usually not easy to be detected and removed effectively. Hence we could not know if the bad performance is due to these problems or due to the method itself. Furthermore, there are no replications for real data sets. So we could not know if the partition produced by a clustering method is by chance or not.

Simulated data sets do not have these drawbacks. They have known cluster structures and are easy to generate. Moreover, we can control the noise and produce as many replicates as we want. Furthermore we also can determine in which situations a clustering method works well and

in which situations it does not work well. Hence simulated data sets are usually used to evaluate the performances of clustering methods.

The qualities of simulated data sets depend on cluster generating algorithms. Many algorithms have been proposed (e.g. Milligan, 1985; Gnanadesikan, et al. 1995; Zhang, et al. 1996; Guha, et al. 1998; Tibshirani, et al. 2001). It seems that Milligan (1985) is the most systematic in addressing the problem of cluster generating. Milligan generated clusters from an experimental design point of view: the factors include the number of clusters, the number of dimensions (variables), sizes of clusters, outliers, noisy variables, and measurement errors. In the design, cluster centers and boundaries are generated one dimension at a time. Cluster boundaries are separated by a random quantity in the first dimension. However there is no constraint on the isolation among clusters in other dimensions. Multivariate normal distributions with diagonal covariance matrices are used to generate data points. Data points are truncated if they fall outside the cluster boundaries.

The main limitation of Milligan's (1985) method is that the degree of separation among clusters is not controlled. The degree of separation is one of the most important factors to check the performances of clustering methods. For data sets with close cluster structures, we expect most clustering methods could not work well. If a clustering method could work well for closely-spaced clusters, then it is reasonable to believe that this method is better than other clustering methods. If clusters are well-separated from each other, then we expect all clustering methods could work well. If a method does not work well for well-separated clusters, it is reasonable to believe that its performance is worse than other clustering methods. Therefore it is desirable to control the degree of separation. To the best of our knowledge, no existing cluster generating algorithms consider the degree of separation as a factor.

The cluster structures generated by Milligan (1985) are also not challenging since clusters are separated in the first dimension and we can detect the cluster structures from pairwise scatter plots. Furthermore the covariance matrices of clusters are diagonal which is usually not true in real data sets. It is quite common in real data sets that clusters are overlapped in all dimensions and covariance matrices are not diagonal.

By a random rotation, we can eliminate the property in Milligan (1985) that clusters are separated only in the first dimension. It is also possible to allow the covariance matrices to have different shapes, diameters and orientations. However it is not easy to control the degree of sepa-

ration. At first thought, we can control the degree of separation by specifying the lengths of the gaps between clusters in the first dimension rather than randomly generating the lengths of the gaps. However this requires that the covariance matrices of clusters be diagonal. And we still can not totally control the degree of separation among clusters since there is no control at all on the isolation in other dimensions.

In this chapter, we improve the cluster generation method proposed in Milligan (1985) so that the degree of separation among clusters could be set to a specified value while the cluster covariance matrices can be arbitrary positive definite matrices. The cluster structures produced by our algorithm have the following desired features:

- The theoretical degrees of separation among clusters can be set to a specified value, based on a separation index.
- No constraint is imposed on the isolation among clusters in each dimension.
- The covariance matrices can have different shapes, diameters and orientations.
- The full cluster structures generally will not be detected simply from pairwise scatter plots.
- Noisy variables and outliers can be imposed to make the cluster structures harder to be recovered.

The structure of this chapter is as follows: The overall cluster generating algorithm is listed in Section 3.2. We give a quantitative description of the degree of separation in Section 3.3. In Section 3.4, we discuss how to allocate the mean vectors or centers. We discuss how to generate covariance matrices in Section 3.5. In Section 3.6, we propose a method to generate noisy variables. Random rotation is a technique used in our cluster generating algorithm to produce clusters so that cluster structures might not be detected from pairwise scatter plots. We describe the random rotation technique in Section 3.7. In Section 3.8, we propose a method to generate outliers. In Section 3.9, we propose a factorial design so that the simulated data sets can be used to systematically study the performances of clustering methods. The verification of the simulated data sets and the discussion are given in Section 3.10. Finally, Section 3.11 contains a summary and proposes possible future research topics.

3.2 Overall Algorithm for Generation of Random Clusters

The main idea of our algorithm for generation of random clusters is to allocate cluster centers to the vertexes of an equilateral simplex. Then we adjust the length of the simplex edges so that the minimum separation among clusters is equal to the specified value J_0. Finally we scale covariance matrices (but keep their shapes, diameters and orientations) so that the separations between clusters and their nearest neighboring clusters are also equal to the specified value J_0. The degree of separation is based on the separation index we proposed in Chapter 2. Other separation indexes can be used. Data points are generated from a mixture of elliptically contoured distributions (for which the univariate margin is fixed up to location and scale), which include multivariate normal as a special case.

In this section, we give the overall algorithm for generation of random clusters. We will describe the details later. The overall algorithm is given below:

Step 1 Input the number of dimensions p, the number of clusters k_0, the degree of separation J_0 for neighboring clusters, the tuning parameter α for the separation index, the number of noisy variables p_2, the lower/upper eigenvalue parameters $\lambda_{\min}$ and $\lambda_{\max}$ for random covariance matrices, and the range of cluster sizes $[n_L,\ n_U]$.

Step 2 Generate cluster centers and random covariance matrices in the p_1 non-noisy dimensions so that neighboring clusters have theoretical separation index J_0 (Details are given in Sections 3.4 and 3.5).

Step 3 Generate sizes of each cluster randomly from the range $[n_L,\ n_U]$ and generate memberships of each data point.

Step 4 Generate the mean vector and covariance matrix of the noisy variables (Details are given in Section 3.6).

Step 5 Apply a random rotation to the cluster means and covariance matrices in Step 2 (Details are given in Section 3.7).

Step 6 From Step 4 and 5, we have cluster means and covariance matrices for all k_0 clusters.

Step 7 Generate random vectors for each of the k_0 clusters from a given family of elliptically contoured distributions.

Step 8 Calculate the theoretical separation index matrices and projection directions via the theoretical mean vectors and covariance matrices.

Step 9 Calculate the sample separation index matrices and projection directions via the sample mean vectors and covariance matrices.

Step 10 Generate outliers. The memberships of outliers are assigned as zero (Details are given in Section 3.8).

3.3 Degree of Separation

The key concept in the algorithm is the degree of separation. In this section, we propose a quantitative description of the degree of separation based on the separation index we proposed in Chapter 2 (see Definition 2.3.4).

If two clusters are generated from two elliptically contoured distributions with mean vector $\boldsymbol{\theta}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$ for $k = 1, 2$, and a common characteristic generator [1], then the theoretical separation index between the two clusters is

$$J_{12}^* = \frac{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - q_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + q_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_1\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)},$$

where $\alpha \in (0, 0.5)$ is a tuning parameter indicating the percentage of data in the extremes to downweight, $q_{\alpha/2}$ is the upper $\alpha/2$ percentile of the standardized univariate margin of elliptically contoured distribution, and $\mathbf{a}$ is the optimal projection direction which maximizes J_{12}^*.

Definition 3.3.1 *Cluster k_2 is the* ***nearest neighboring cluster*** *of cluster k_1 if the separation index J_{k_1,k_2}^* between cluster k_1 and cluster k_2 is the smallest among the pairwise separation indexes of cluster k_1 and other clusters. That is,*

$$J_{k_1,k_2}^* = \min_{k=1,\ldots,k_0, k\neq k_1} J_{k_1,k}^*,$$

where k_0 is the number of clusters. We denote $J_{k_1\,\min}^*$ as the separation index between cluster k_1 and its nearest neighboring cluster.

[1] Characteristic generator is term used in Fang et al. (1990)

The degree of separation then can be measured by the separation indexes $J^*_{k\,\min}$, $k = 1, \ldots, k_0$. If $J^*_{k\,\min}$, $k = 1, \ldots, k_0$, are all close to zero, then the cluster structure is close. If $J^*_{k\,\min}$, $k = 1, \ldots, k_0$, are all quite large, then the cluster structure is well-separated. However it is difficult to make a clear-cut decision on whether a cluster structure is "close", "separated", or "well-separated". For the factorial design in Section 3.9, we regard a cluster structure as close if $J^*_{k\,\min} = 0.010$, $k = 1, \ldots, k_0$, as separated if $J^*_{k\,\min} = 0.210$, $k = 1, \ldots, k_0$, and as well-separated if $J^*_{k\,\min} = 0.342$, $k = 1, \ldots, k_0$. The value 0.010 is the separation index between two clusters, which are generated from two univariate normal distributions $\mathrm{N}(0, 1)$ and $\mathrm{N}(A, 1)$, where $A = 4$. The values 0.210 and 0.342 are the separation indexes when $A = 6$ and $A = 8$ respectively. The tuning parameter α is equal to 0.05 when calculating these separation indexes. Figure 3.1 shows that the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 4)$ are close to each other, the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 6)$ are separated from each other, and the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 8)$ are well-separated from each other.

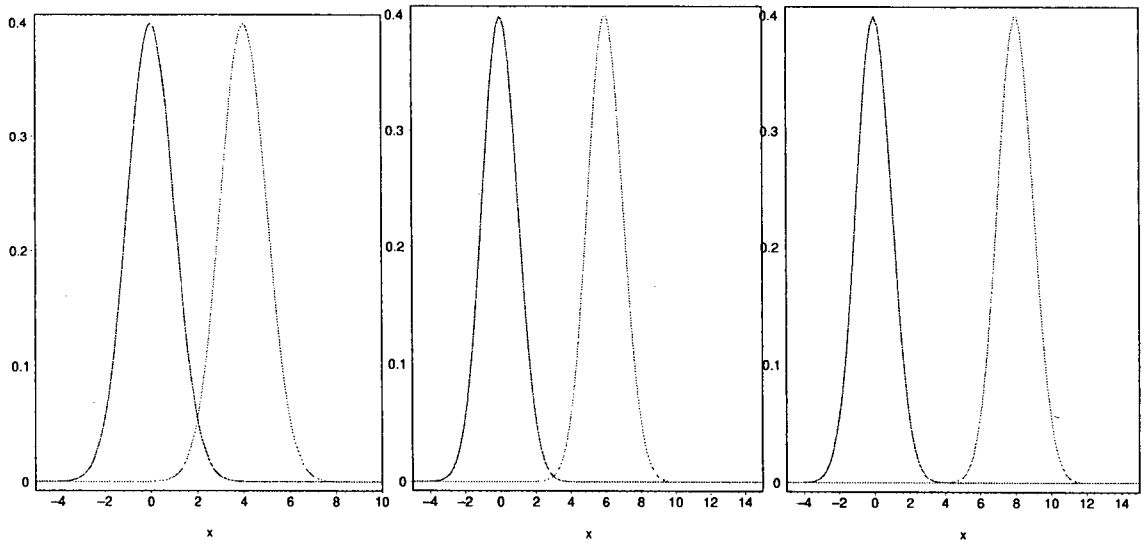


Figure 3.1: The left panel shows the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 4)$. The middle panel shows the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 6)$. And the right panel shows the densities of $\mathrm{N}(0, 0)$ and $\mathrm{N}(0, 8)$.

3.4 Allocating Cluster Centers

One key to generating cluster structures with specified degree J_0 of separation is the allocation of the cluster centers. If there are only two clusters, then it is easy to allocate the cluster centers so

that the separation index is equal to J_0. However, if there are more than two clusters, then it is not easy to allocate the cluster centers so that the separation indexes ($J^*_{k\,\min}$, $k = 1, \ldots, k_0$, where k_0 is the number of clusters) between clusters and their nearest neighboring clusters are equal to J_0, except for the trivial case where the cluster covariance matrices are all equal to a multiple of the identity matrix.

For the trivial cases, we can first generate a simplex in a p_1-dimensional space whose edges have equal length L, where p_1 is the number of non-noisy variables. Then we allocate cluster centers to the vertexes of the simplex. Finally, we adjust the length L so that $J^*_{k\,\min} = J_0$, $k = 1, \ldots, k_0$. If the number k_0 of clusters is smaller than or equal to $p_1 + 1$, then we can take the first k_0 vertexes as cluster centers. If k_0 is larger than $p_1 + 1$, then we can construct several simplexes and connect them together until all cluster centers are allocated. Figure 3.2 illustrates the locations of the five vertexes of two connected simplexes in a two-dimensional space.

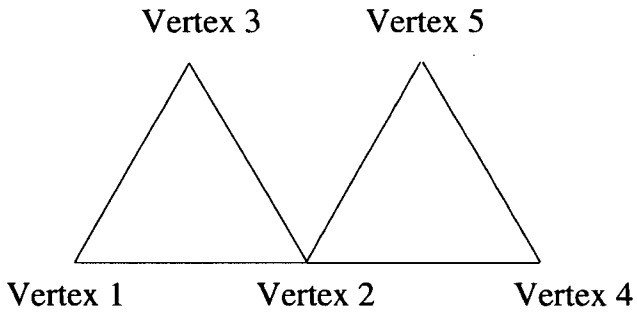


Figure 3.2: The five vertexes of two connected simplexes in a two-dimensional space

A **simplex** in a p_1-dimensional space contains $p_1 + 1$ vertexes $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_{p_1+1}$. These $p_1 + 1$ vertexes are linearly dependent. However all its proper subsets are linearly independent. That is, there exists a non-zero vector $\boldsymbol{x} = (x_1, \ldots, x_{p_1+1})^T$ for the homogeneous linear equation

$$\sum_{i=1}^{p_1+1} x_i \boldsymbol{v}_i = \mathbf{0}_{p_1},$$

where $\mathbf{0}_{p_1}$ is a $p_1 \times 1$ vector whose elements are all zero, x_i, $i = 1, \ldots, p_1 + 1$, are scalars. And for any subset $\{\boldsymbol{v}_{\ell_j} : j = 1, \ldots, m\}$, $m \in \{1, \ldots, p_1\}$, the only solution for the homogeneous linear equation

$$\sum_{j=1}^{m} x_j \boldsymbol{v}_{\ell_j} = \mathbf{0}_{p_1}$$

is $\boldsymbol{x} = \mathbf{0}_m$.

A simplex is a line segment, a triangle, and a tetrahedron in one-, two-, and three-dimensional space respectively. The lengths of the simplex edges are not necessarily equal.

For the non-trivial cases where not all eigenvalues of cluster covariance matrices are equal, we can still construct simplexes with equilateral edges to allocate cluster centers. However we can only make sure the minimum separation index among clusters is equal to J_0 and can not control other separation indexes. To make sure $J^*_{j\min} = J_0$, $j = 1, \ldots, k_0$, we have to scale the covariance matrices (but keep the cluster shape and orientation).

With following cluster-center-allocation algorithm, we can obtain mean vectors and covariance matrices of clusters so that the theoretical separation indexes $J^*_{k\min}$, $k = 1, \ldots, k_0$, are all equal to J_0.

Cluster-Center-Allocation Algorithm:

Step (a) Generate k_0 covariance matrices $\boldsymbol{\Sigma}_k$, $k = 1, \ldots, k_0$.

Step (b) Construct a p_1-dimensional equilateral simplex whose edges have length 2. The first two vertexes are $\boldsymbol{v}_1 = -\boldsymbol{e}_1$ and $\boldsymbol{v}_2 = \boldsymbol{e}_1$ respectively, where the $p_1 \times 1$ vector $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^T$. Denote the j-th vertex as $\boldsymbol{v}_j$, $j = 1, \ldots, p_1 + 1$.

Step (c) If $k_0 \leq p_1 + 1$, then take the first k_0 vertexes of the simplex as initial cluster centers. If $k_0 > p_1 + 1$, then we start adding vertexes from the following sequence after $\boldsymbol{v}_{p_1+1}$ until all k_0 cluster centers are allocated:

$$\boldsymbol{v}_2 + 2 * \boldsymbol{e}_1, \ldots, \boldsymbol{v}_{p_1+1} + 2 * \boldsymbol{e}_1, \boldsymbol{v}_2 + 4 * \boldsymbol{e}_1, \quad \ldots, \boldsymbol{v}_{p_1+1} + 4 * \boldsymbol{e}_1,$$

$$\boldsymbol{v}_2 + 6 * \boldsymbol{e}_1, \ldots, \boldsymbol{v}_{p_1+1} + 6 * \boldsymbol{e}_1,$$

Essentially this just keeps on adding points on a shifted symmetric simplex.

Step (d) Calculate the separation index matrix $\boldsymbol{J}^*_{k_0 \times k_0}$.

Step (e) Scale the length of the simplex edge by a scalar c_1 so that the minimum separation index $\min_{i \neq j} J^*_{ij}$ among clusters is equal to J_0.

Step (f) Obtain the separation indexes, $J^*_{k\min}$, $k = 1, \ldots, k_0$, between clusters and their nearest neighboring clusters, and obtain $k^* = \arg \max_{k=1,\ldots k_0} J^*_{k\min}$. If $J^*_{k^*\min} > J_0$, then go to Step (g). Otherwise scaling is complete.

Step (g) Scale the covariance matrix $\boldsymbol{\Sigma}_{k^*}$ by a scalar c_2 so that $J^*_{k^*\,\min} = J_0$. Go back to Step (f).

In the above algorithm, we did not mention how to obtain the simplex vertexes except for the first two vertexes. In fact the vertexes of a p-dimensional equilateral simplex, whose edge length is $L = 2$ and first two vertexes are $\boldsymbol{v}_1 = -\boldsymbol{e}_1$ and $\boldsymbol{v}_2 = \boldsymbol{e}_1$, are not unique. In the following, we propose a way to obtain a set of vertexes for a p-dimensional equilateral simplex whose edge length is 2 and first two vertexes are $\boldsymbol{v}_1 = -\boldsymbol{e}_1$ and $\boldsymbol{v}_2 = \boldsymbol{e}_1$.

We first describe how to obtain the third vertex for the p-dimensional simplex. Since the simplex is equilateral, we have

$$(\boldsymbol{v}_3 - \boldsymbol{v}_1)^T (\boldsymbol{v}_3 - \boldsymbol{v}_1) = 4,$$
$$(\boldsymbol{v}_3 - \boldsymbol{v}_2)^T (\boldsymbol{v}_3 - \boldsymbol{v}_2) = 4.$$

By adding the two equations, we can obtain

$$\boldsymbol{v}_3^T \boldsymbol{v}_3 - 2\boldsymbol{v}_3 \bar{\boldsymbol{v}}_2 = 4 - \frac{1}{2}\sum_{i=1}^{2} \boldsymbol{v}_i^T \boldsymbol{v}_i,$$

where

$$\bar{\boldsymbol{v}}_2 = \frac{1}{2}\sum_{i=1}^{2} \boldsymbol{v}_i = (\bar{v}_{21}, \ldots, \bar{v}_{2p})^T.$$

Let $v_{31} = \bar{v}_{21}$ and $v_{33} = \cdots = v_{3p} = 0$. Then we can get

$$v_{32}^2 = 4 - \frac{1}{2}\sum_{i=1}^{2} \boldsymbol{v}_i^T \boldsymbol{v}_i + \bar{v}_{21}^2 + 2v_{32}\bar{v}_{22}.$$

Note that $\bar{v}_{22} = 0$. Hence

$$v_{32} = \left\{ 4 - \left[\frac{1}{2}\sum_{i=1}^{2} \boldsymbol{v}_i^T \boldsymbol{v}_i - \bar{\boldsymbol{v}}_2^T \bar{\boldsymbol{v}}_2 \right] \right\}^{1/2}$$
$$= \left\{ 4 - \frac{1}{2}\sum_{i=1}^{2} (\boldsymbol{v}_i - \bar{\boldsymbol{v}}_2)^T (\boldsymbol{v}_i - \bar{\boldsymbol{v}}_2) \right\}^{1/2}.$$

By using the same technique, we can obtain the coordinates of the k-th vertex $\boldsymbol{v}_k$ given $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_{k-1}$, $2 < k \leq p+1$:

$$v_{ki} = \bar{v}_{ki},\ i = 1, \ldots, k-2,$$
$$v_{kj} = 0,\ j = k, \ldots, p,$$
$$v_{k,k-1} = \left\{ 4 - \frac{1}{k-1}\sum_{i=1}^{k-1} (\boldsymbol{v}_i - \bar{\boldsymbol{v}}_k)^T (\boldsymbol{v}_i - \bar{\boldsymbol{v}}_k) \right\}^{1/2},$$

where

$$\bar{v}_k = \frac{1}{k-1}\sum_{i=1}^{k-1} v_i = (\bar{v}_{k1}, \ldots, \bar{v}_{kp})^T .$$

3.5 Generating a Covariance Matrix

To generate a $p \times p$ covariance matrix $\mathbf{\Sigma}$, we use the following decomposition:

$$\mathbf{\Sigma} = \boldsymbol{Q}\mathbf{\Lambda}\boldsymbol{Q}^T,$$

where $\boldsymbol{Q}$ is a $p \times p$ orthogonal matrix, $\mathbf{\Lambda} = \text{diag}\,(\lambda_1, \ldots, \lambda_p)$, i.e.

$$\mathbf{\Lambda} = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_p \end{pmatrix}.$$

$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p > 0$ are eigenvalues of $\mathbf{\Sigma}$. The j-th column of the orthogonal matrix $\boldsymbol{Q}$ is the normalized eigenvector of $\mathbf{\Sigma}$ corresponding to the eigenvalue λ_j. The term "normalized" means that each column of $\boldsymbol{Q}$ has unit length. In other words $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{I}_p$, where $\boldsymbol{I}_p$ is the p-dimensional identity matrix.

The eigenvalues and eigenvectors determine the diameter, shape and orientation of the matrix $\mathbf{\Sigma}$ respectively. Different eigenvalues and eigenvectors correspond to covariance matrices with different diameters, shapes and orientations. To randomly generate a positive definite matrix $\mathbf{\Sigma}$ is equivalent to randomly generating p positive real numbers $\lambda_1, \ldots, \lambda_p$, and a normalized orthogonal matrix $\boldsymbol{Q}$.

We can generate p eigenvalues with equal probability from an interval whose lower bound is positive. The lower bound λ_{min} and upper bound λ_{max} of the interval determine the size and variation of the eigenvalues. If the difference $\lambda_{\text{max}} - \lambda_{\text{min}}$ is too large, then it is highly possible that clusters generated are too elongated. It is not common in real problems that the shapes of clusters are too elongated. If λ_{min} is too large, then the diameters of clusters are too large. Given the number of data points, the larger the diameter of a cluster is, the more sparse the data points would be. So to avoid too elongated clusters and too sparse clusters, we use $\lambda_{\text{min}} = 1$ and $\lambda_{\text{max}} = 10$ when we generate simulated data sets. Other values can be used.

To generate a $p \times p$ normalized orthogonal matrix $\boldsymbol{Q}$, we can first generate a $p \times p$ lower triangle matrix $\boldsymbol{M}$ whose diagonal elements are all non-zero. Then we use the Gram-Schmidt Orthogonalization (Kotz and Johnson, 1983, Volume 3, page 478) to transform the lower triangle matrix $\boldsymbol{M}$ to a normalized orthogonal matrix. Denote $\boldsymbol{q}_j$ and $\boldsymbol{m}_j$ as the j-th column of the matrices $\boldsymbol{Q}$ and $\boldsymbol{M}$ respectively, $j = 1, \ldots, p$. Then the procedure of the Gram-Schmidt Orthogonalization is as follows:

Step 1 $\boldsymbol{q}_1 = \boldsymbol{m}_1$.

Step 2

$$\boldsymbol{q}_j = \boldsymbol{m}_j - \sum_{i=1}^{j-1} \frac{\boldsymbol{m}_j^T \boldsymbol{q}_i}{\boldsymbol{q}_i^T \boldsymbol{q}_i} \boldsymbol{q}_i, \quad j = 2, \ldots, p.$$

Step 3

$$\boldsymbol{q}_j = \frac{\boldsymbol{q}_j}{\|\boldsymbol{q}_j\|}, \quad \text{where } \|\boldsymbol{q}_j\| = \sqrt{\boldsymbol{q}_j^T \boldsymbol{q}_j},\ j = 1, \ldots, p.$$

3.6 Constructing Noisy Variables

There is no unified definition of noisy variable. Milligan (1985) assumed that noisy variables are uniformly distributed and are independent of each other and of non-noisy variables. We assume that noisy variables are normally distributed and independent of non-noisy variables. However, noisy variables are not necessarily independent of each other.

Like Milligan (1985), we require that the variations of noisy variables in the generated data sets are similar to those of non-noisy variables. If noisy variables have smaller variations than those of non-noisy variables, then we implicitly downweight noisy variables. Hence the data sets would be less challenging.

Denote the $p_1 \times p_1$ matrix $\boldsymbol{\Sigma}^*$ as the covariance matrix of non-noisy variables and the $p_2 \times p_2$ matrix $\boldsymbol{\Sigma}_0$ as the covariance matrix of noisy variables. One possible way to make the variations of noisy variables similar to those of non-noisy variables is to make the ranges of eigenvalues of $\boldsymbol{\Sigma}_0$ similar to those of $\boldsymbol{\Sigma}^*$.

If we assume that data points in non-noisy dimensions are from a mixture of distributions with the density function $f(\boldsymbol{x}) = \sum_{k=1}^{k_0} \pi_k f_k(\boldsymbol{x})$, where $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{k_0} \pi_k = 1$, then the

covariance matrix $\mathbf{\Sigma}^*$ of the mixture of distributions is

$$\mathbf{\Sigma}^* = \sum_{k=1}^{k_0} \pi_k \mathbf{\Sigma}_k + \sum_{k<k'} \pi_k \pi_k' \left(\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k'}\right) \left(\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k'}\right)^T,$$

where $\boldsymbol{\mu}_k$ and $\mathbf{\Sigma}_k$ are the mean vector and covariance matrix of the k-th component of the mixture of distributions. We can randomly generate the eigenvalues of $\mathbf{\Sigma}_0$ from the interval $[\lambda_{p_1}^*, \lambda_1^*]$, where p_1 is the number of non-noisy variables, $\lambda_{p_1}^*$ and λ_1^* are the minimum and maximum eigenvalues of the matrix $\mathbf{\Sigma}^*$. In this way, the variations of noisy variables would be similar to those of non-noisy variables.

The mean vector

$$\boldsymbol{\mu}^* = \sum_{k=1}^{k_0} \pi_k \boldsymbol{\mu}_k$$

of the mixture of distributions can be used to generate the $p_2 \times 1$ mean vector $\boldsymbol{\mu}_0$ of the noisy variables. For example, we can randomly generate the p_2 elements of $\boldsymbol{\mu}_0$ from the interval $[\min_{j=1,\ldots,p_1} \mu_j^*, \max_{j=1,\ldots,p_1} \mu_j^*]$.

Once we generate the mean vectors and covariance matrices of non-noisy and noisy variables, we can randomize the labels of variables to make the generated data sets closer to real data sets.

3.7 Rotating Data Points

In most cases, we could not detect the numbers of clusters of real data sets in high dimensional spaces by lower dimensional scatter plots. However, the simulated data sets produced by the methods mentioned in the cluster analysis literature do not always have this property. For example, we can easily detect the numbers of clusters in data sets generated by Milligan (1985) from the scatter plots of the first variable versus any one of other variables. This is because Milligan (1985) intentionally makes sure that clusters are separated in the first dimension. The data sets generated by our algorithm might have the same problem.

To improve the simulated data sets so that we could not detect the numbers of clusters by scatter plots, we can simply transform these data sets by random rotations. To rotate a data point $\boldsymbol{x}$, we can apply the transformation $\boldsymbol{y} = \boldsymbol{Q}\boldsymbol{x}$, where $\boldsymbol{Q}$ is an orthogonal matrix. We can use the method proposed in Section 3.5 to generate an orthogonal matrix. We only rotate non-noisy

variables and do not rotate noisy variables because otherwise it is possible that noisy variables are no longer noisy after rotation.

The effect of random rotation can be visualized by an example shown in Figure 3.3. In this

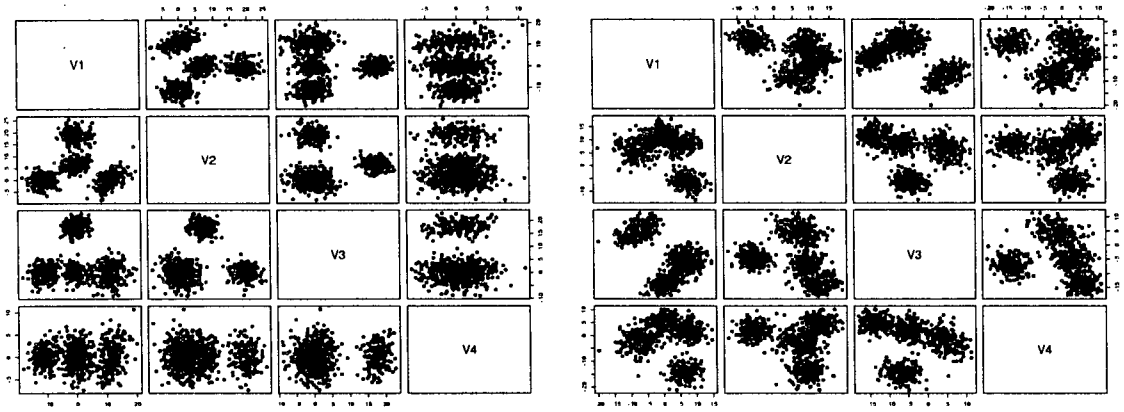


Figure 3.3: Effect of random rotation. The left panel shows the pairwise scatter plot of the original dat set containing 4 well-separated clusters in a 4-dimensional space. The right panel is the pairwise scatter plot after a random rotation.

example, the original data set has four well-separated clusters in a four-dimensional space. The sample separation index matrix is

$$\begin{pmatrix} -1.000 & 0.342 & 0.342 & 0.362 \\ 0.342 & -1.000 & 0.358 & 0.402 \\ 0.342 & 0.358 & -1.000 & 0.341 \\ 0.362 & 0.402 & 0.341 & -1.000 \end{pmatrix}$$

The pairwise scatter plot in the left panel of Figure 3.3 shows obvious four-cluster structure. The right panel of Figure 3.3 shows the pairwise scatter plot of the original data set after a random rotation. We can see that there are two obvious clusters. However, the four-cluster structure is no longer obvious.

3.8 Adding Outliers

Outliers, like noisy variables, are frequently encountered in real data sets. And outliers may affect the recovery of true cluster structures. Therefore any cluster generating algorithm should provide a function to produce outliers for simulated data sets.

Milligan (1985) generated outliers for each cluster from multivariate normal distributions N ($\boldsymbol{\mu}_k$, $9\boldsymbol{\Sigma}_k$), where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$, are the mean vector and covariance matrix of the k-th cluster, $k = 1, \ldots, k_0$. A point generated from N ($\boldsymbol{\mu}_k$, $9\boldsymbol{\Sigma}_k$) is accepted as an outlier of the k-th cluster if this point exceeds the boundary of the k-th cluster on at least one dimension. The number of outliers of the k-th cluster is proportional to the size of the k-th cluster. There are either 20% or 40% additional points which are added to each cluster as outliers.

For simplicity, we generate outliers from a distribution whose marginal distributions are independent uniform distributions. The outliers are generated for the whole data set instead of for each cluster. The range of the j-th marginal uniform distribution depends on the range of non-outliers in the j-th dimension. We set the range as $[\hat{\mu}_j - 4\hat{\boldsymbol{\Sigma}}_j, \hat{\mu}_j + 4\hat{\boldsymbol{\Sigma}}_j]$, where $\hat{\mu}_j$ and $\hat{\boldsymbol{\Sigma}}_j$ are the sample mean and standard deviation of the j-th variable respectively.

3.9 Factorial Design

We can use different combinations of the input parameters in our cluster generating algorithm to generate data sets with different cluster structures. Like Milligan (1985), we can propose a factorial design with input parameters as factors so that the simulated data sets can be used to systematically study the performances of clustering methods.

The factorial design is an effective technique to decompose different factors which might affect the performances of clustering methods. The factors include the type of data (categorical or continuous), the number of data points, the number of variables, the number of clusters, the shapes, diameters and orientations of clusters, the degrees of separation between clusters, sizes of clusters, noisy variables, outliers, missing values, and measurement errors, etc. Not all these factors are considered in this chapter. Those factors which are not considered in this chapter will be studied in future research.

Before we propose our design, we briefly review the factorial design in Milligan (1985), which has 3 factors: (1) the number of clusters; (2) the number of dimensions; and (3) sizes of clusters. For the first factor, there are 4 levels — 2, 3, 4 or 5 clusters. There are 3 levels for the second factor — 4, 6, or 8 dimensions. For the third factor, there are also three levels: all clusters have equal size; one cluster contains 10% of the data points while the other clusters have equal cluster

size; one cluster contains 60% of the data points while other clusters have equal cluster size. This is summarized in Table 3.1. There are three replicates for each combination of the design. So the design produces $3 \times 4 \times 3 \times 3 = 3 \times 36 = 108$ data sets.

Table 3.1: The design proposed by Milligan (1985)

Factors	Levels
Number of clusters	2, 3, 4, 5
Number of dimensions	4, 6, 8
Cluster sizes	All equal; 10% and others equal; 60% and others equal
Totally $4 \times 3 \times 3 = 36$ combinations in the design.	

Real data sets usually are not "clean". There exist outliers, noisy variables, measurement errors, etc.. In Milligan's (1985) design, different kinds of noises (e.g. outliers, noisy variables, measurement errors, etc.) were imposed on the simulated clusters so that the simulated data sets are closer to real situations.

Milligan did not consider the degree of separation as a factor. However the degree of separation is important. If clusters generated are well-separated, then we expect any clustering algorithm would have good performance. If a clustering method works poorly for well-separated cluster structures, then we expect it could not work well for real data sets. If clusters generated are close to each other, then we expect some clustering methods would work poorly. If a clustering method has good performance for close cluster structures, then we expect it would work well for real data sets. Thus, it is desirable to generate well-separated, separated, and close cluster structures respectively.

In our design, there are four factors: (1) the number of clusters; (2) the degree of separation — we regard a cluster structure as `close` if $J^*_{k\,\min} = 0.010$, $k = 1, \ldots, k_0$, as `separated` if $J^*_{k\,\min} = 0.210$, $k = 1, \ldots, k_0$, and as `well-separated` if $J^*_{k\,\min} = 0.342$, $k = 1, \ldots, k_0$; (3) the number p_1 of non-noisy variables; and (4) the number p_2 of noisy variables The levels within each factor are listed in Table 3.2. There are three replicates for each combination of the design. So the design produces $3 \times 3 \times 3 \times 3 \times 3 = 3 \times 81 = 243$ data sets.

In our design, we do not consider cluster size as a factor. Instead, we will randomly generate cluster sizes within a specified range. If the range is large enough, then it is possible to get quite different cluster sizes. If the range is small enough, then we can get almost equal cluster sizes.

It is well-known (see Chapter 5) that noisy variables may mask true cluster structure. So in

Table 3.2: The factors and their levels in our design

Factors	Levels
Number of clusters	3, 6, 9
Degree of Separation	close, separated, well-separated
Number of non-noisy variables p_1	4, 8, 20
Number of noisy variables p_2	1, $0.5p_1$, p_1
Totally $3 \times 3 \times 3 \times 3 = 81$ combinations in the design.	

our design, we explicitly add the number of noisy variables as a factor. For the first level, we add 1 noisy variable. We expect clustering methods will still work well in this case. For the second level, we add $0.5p_1$ noisy variables, where p_1 is the number of non-noisy variables. We expect clustering algorithms may not work well in this case except when clusters are far apart. For the third level, we add p_1 noisy variables. That is, half of variables are noisy variables. We expect clustering methods may not work well.

We expect a high ratio of noisy variables and close clusters to be the most difficult scenario for clustering methods.

3.10 Verification and Discussion

Milligan (1985) used three different verification procedures to the data sets produced by his design:

- A discriminant analysis of each data set. However, Milligan (1985) didn't mention the details.
- Check if clusters are overlapped on the first dimension.
- Apply four common hierarchical agglomerative methods to the data sets. Then count misclassification rates and calculate Rand indexes which measure the agreements between true partitions and the partitions obtained by clustering methods.

For the data sets generated by our design, we only need to check whether the theoretical and sample separation indexes, $J^*_{k\min}$ and $\hat{J}^*_{k\min}$, $k = 1, \ldots, k_0$, are close to the specified degree of separation J_0.

Our design produces 243 data sets. There are 81 data sets each for close, separated and well-separated cluster structures. For data sets with close cluster structures, we pool the $J^*_{k\min}$'s into a set S_{Jc} and pool the $\hat{J}^*_{k\min}$'s into a set $S_{\hat{J}c}$. Similarly, we can get sets S_{Js} and $S_{\hat{J}s}$ for data

sets with separated cluster structures and get sets S_{Jw} and $S_{\hat{J}w}$ for data sets with well-separated cluster structures.

We expect that all elements in S_{Jc} are very close to 0.010 since we require them be equal to 0.010 for the theoretical version based on mixtures of multivariate elliptically contoured distributions. Similarly, we expect that all elements in S_{Js} and S_{Jw} are very close to 0.210 and 0.342 respectively. We also expect that the elements in the sets $S_{\hat{J}c}$, $S_{\hat{J}s}$, and $S_{\hat{J}w}$ are close to 0.010, 0.210, and 0.342 respectively. To check these, we can use the sample means, sample standard deviations, estimates of biases, and estimates of MSE to measure the closeness of the obtained degrees of separation to the specified degree of separation.

Let S be any one of the sets S_{Jc}, S_{Js}, S_{Jw}, $S_{\hat{J}c}$, $S_{\hat{J}s}$, and $S_{\hat{J}w}$. Denote s_i as the i-th element of the set S and m as the number of elements in the set S. The estimates of the bias and MSE for the specified degree of separation J_0 are defined as

$$\widehat{\text{bias}}\,(S) = \bar{S} - J_0,$$
$$\widehat{\text{MSE}}\,(S) = \widehat{Var}(S) + \widehat{\text{bias}}\,(S)^2,$$

where

$$\bar{S} = \frac{1}{m}\sum_{i=1}^{m} s_i,$$
$$\widehat{Var}(S) = \frac{1}{m-1}\sum_{i=1}^{m}(s_i - \bar{S})^2,$$

are the sample mean and variance of the set S respectively. The specified degree of separation J_0 can take values 0.010, 0.210, or 0.342.

When generating these 243 data sets, we set $\alpha = 0.05$, $\lambda_{\text{min}} = 1$, and $\lambda_{\text{max}} = 10$ and we use mixtures of multivariate normal distributions. The cluster sizes are randomly generated from the interval [200, 500].

Table 3.3 lists the results for the sets S_{Jc}, S_{Js}, and S_{Jw}. We can see that the theoretical degrees of separation of the data sets are very close to the specified degrees of separation. In fact there is nothing random here. So we expect there is no variation among elements in each set.

Table 3.4 lists the results for the sets $S_{\hat{J}c}$, $S_{\hat{J}s}$, and $S_{\hat{J}w}$. We can see that the sample degrees of separation of the data sets are close to the specified degree of separation. Overall the sample degrees of separation are slightly larger than the specified degree of separation, although by

Table 3.3: The sample means and standard deviations of the sets S_{Jc}, S_{Js}, and S_{Jw} as well as the corresponding estimates of biases and MSEs of J_0.

J_0	mean (sd)	bias	$\sqrt{\text{MSE}}$
0.010	0.010 (0.000)	-0.000	0.000
0.210	0.210 (0.000)	-0.000	0.000
0.342	0.342 (0.000)	-0.000	0.000

randomness some sample degrees of separation are smaller than the specified degrees of separation by checking the log files produced by our algorithm.

Table 3.4: The sample means and standard deviations of the sets $S_{\hat{J}c}$, $S_{\hat{J}s}$, and $S_{\hat{J}w}$ as well as the corresponding estimates of biases and MSEs of J_0.

J_0	mean (sd)	bias	$\sqrt{\text{MSE}}$
0.010	0.013 (0.016)	0.003	0.016
0.210	0.211 (0.015)	0.001	0.015
0.342	0.344 (0.013)	0.002	0.013

In addition to the separation indexes between clusters and their nearest neighbors, the separation indexes between clusters and their other **direct neighboring clusters** can also provide useful information about the degree of separation of the cluster structure in a data set. In our algorithm, the cluster k_2 is a **direct neighboring cluster** of the cluster k_1 if the distance between the two cluster centers is equal to L, where L is the edge length of the simplexes.

When we generate the 243 data sets, we record the separation indexes between clusters and their farthest direct neighboring clusters. We denote these separation indexes as the **farthest separation indexes**. We also record the median of the separation indexes between clusters and their direct neighboring clusters. We denote these separation indexes as the **median separation indexes**. We pool the theoretical and sample farthest separation indexes of the data sets with close cluster structures into the sets S_{Jcf} and $S_{\hat{J}cf}$ respectively. Similarly, the sets S_{Jcf} and $S_{\hat{J}cf}$ for separated, S_{Jwf} and $S_{\hat{J}wf}$ for well-separated. Similarly, we pool the theoretical median separation indexes of the data sets with close, separated, and well-separated cluster structures into the sets S_{Jcm}, S_{Jsm}, and S_{Jwm} respectively. The sample median separation indexes are pooled into the sets $S_{\hat{J}cm}$, $S_{\hat{J}sm}$, and $S_{\hat{J}wm}$ respectively. The means and standard deviations of these sets are listed in Table 3.5 and Table 3.6.

Table 3.5: Means and standard deviations of the median separation indexes.

J_0	mean (sd) (set)	mean (sd) (set)
0.010	0.043 (0.025) (S_{Jcm})	0.052 (0.026) ($S_{\hat{J}cm}$)
0.210	0.241 (0.026) (S_{Jsm})	0.249 (0.027) ($S_{\hat{J}cm}$)
0.342	0.372 (0.023) (S_{Jwm})	0.379 (0.023) ($S_{\hat{J}cm}$)

Table 3.5 shows that the median separation indexes are still close to the specified degree of separation. This is desirable since we want the separation indexes between clusters and their direct neighboring clusters as close to the specified degree of separation as possible.

Table 3.6: Means and standard deviations of the farthest separation indexes.

J_0	mean (sd) (set)	mean (sd) (set)
0.010	0.098 (0.054) (S_{Jcf})	0.111 (0.053) ($S_{\hat{J}cf}$)
0.210	0.291 (0.051) (S_{Jsf})	0.301 (0.051) ($S_{\hat{J}sf}$)
0.342	0.416 (0.044) (S_{Jwf})	0.424 (0.043) ($S_{\hat{J}wf}$)

However, the farthest separation indexes in Table 3.6 are much larger than the specified separation indexes because if $k_0 > p$, then not all cluster centers can be neighboring vertexes of a simplex.

In our cluster generating algorithm, we calculate the separation indexes after generating the mean vector and covariance matrix of the noisy variables. Table 3.3 shows that noisy variables do not affect the projection directions and separation indexes if the true cluster structures are known. In fact, we can show this theoretically.

Suppose that clusters 1 and 2 are in a p-dimensional space. Without loss of generality, suppose that the first p_1 variables are non-noisy and the remaining p_2 variables are noisy variables ($p = p_1 + p_2$). Then we can partition the mean vectors and covariance matrices of the two clusters as follows:

$$\boldsymbol{\mu}_k = \begin{pmatrix} \boldsymbol{\theta}_k \\ \boldsymbol{\theta} \end{pmatrix}, \quad \boldsymbol{\Sigma}_k = \begin{pmatrix} \boldsymbol{V}_{k11} & 0 \\ 0 & \boldsymbol{V} \end{pmatrix}, \; k = 1, 2,$$

where the $p_1 \times 1$ vector $\boldsymbol{\theta}_k$ is the mean vector of cluster k on non-noisy dimensions, $k = 1, 2$, the $p_2 \times 1$ vector $\boldsymbol{\theta}$ is the mean vector for noisy variables, the $p_1 \times p_1$ matrix $\boldsymbol{V}_{k11}$ is the covariance matrix of cluster k in non-noisy dimensions, $k = 1, 2$, and the $p_2 \times p_2$ matrix $\boldsymbol{V}$ is the covariance matrix for noisy variables. Recall that we assume that noisy variables are independent of the

non-noisy variables. That is why the covariance matrices are block-diagonal.

The optimal projection direction $\boldsymbol{a}^*$ for cluster 1 and 2 satisfies:

$$J^*(\boldsymbol{a}^*) = \max_{\boldsymbol{a}} J^*(\boldsymbol{a}) = \max_{\boldsymbol{a}} \frac{\boldsymbol{a}^T[\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1] - Z_{\alpha/2}\left(\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_1\boldsymbol{a}} + \sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_2\boldsymbol{a}}\right)}{\boldsymbol{a}^T[\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1] + Z_{\alpha/2}\left(\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_1\boldsymbol{a}} + \sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}_2\boldsymbol{a}}\right)}$$

$$= \max_{\boldsymbol{a}} \frac{\boldsymbol{a}_1^T[\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1] - Z_{\alpha/2}\left(\sqrt{\boldsymbol{a}_1^T\boldsymbol{V}_{111}\boldsymbol{a}_1 + \boldsymbol{a}_2^T\boldsymbol{V}\boldsymbol{a}_2} + \sqrt{\boldsymbol{a}_1^T\boldsymbol{V}_{211}\boldsymbol{a}_1 + \boldsymbol{a}_2^T\boldsymbol{V}\boldsymbol{a}_2}\right)}{\boldsymbol{a}_1^T[\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1] + Z_{\alpha/2}\left(\sqrt{\boldsymbol{a}_1^T\boldsymbol{V}_{111}\boldsymbol{a}_1 + \boldsymbol{a}_2^T\boldsymbol{V}\boldsymbol{a}_2} + \sqrt{\boldsymbol{a}_1^T\boldsymbol{V}_{211}\boldsymbol{a}_1 + \boldsymbol{a}_2^T\boldsymbol{V}\boldsymbol{a}_2}\right)}$$

It is not difficult to show that the function

$$h(x) = \frac{b - x}{b + x}$$

is a monotone decreasing function of x if $b > 0$. To maximize $J^*(\boldsymbol{a})$, we only need to minimize the part containing $\boldsymbol{a}_2$. For fixed $\boldsymbol{a}_1$, $J^*(\boldsymbol{a})$ reach its maximum when $\boldsymbol{a}_2 = \boldsymbol{0}$. Therefore, the optimal projection direction does not depend on the noisy variables. Consequently, the optimal separation index does not depend on the noisy variables.

3.11 Summary and Future Research

In this chapter, we quantify the degree of separation among clusters based on the separation index we proposed in Chapter 2 and propose a cluster generating algorithm which can generate clusters with a specified degree of separation. We improve Milligan's (1985) factorial design so that (1) the theoretical degrees of separation for neighboring clusters can be set to the specified degree of separation; (2) no constraint is imposed on the isolation among clusters in each dimension; (3) the covariance matrices of clusters can have different shapes, diameters and orientations; (4) the full cluster structures generally will not be detected simply by pairwise scatter plots.

To generate data sets, we assume that variables are all continuous and that clusters have elliptically contoured structures. However the variables in real data sets usually have mixed-type variables and are not elliptically contoured. We will investigate how to generate data sets with mixed-type variables and non-elliptically contoured structures.

Missing values are common in real data sets and can be caused by different mechanisms. We will investigate how to generate missing values in our future research.

Chapter 4

A Sequential Clustering Method

4.1 Introduction

It is an important issue in cluster analysis to determine the number of clusters or an interval for the plausible number of clusters. The ultimate goal of cluster analysis is to determine if there exist patterns (clusters) in multivariate data sets or not. If clusters exist, then we would like to determine how many there are in the data set. Many clustering methods (e.g. non-hierarchical clustering methods) require specification of the number of clusters as an input parameter. Although hierarchical clustering methods do not need the information about the number of clusters, we have to specify the number of clusters if we want to get a single partition instead of a tree structure.

It is a quite challenging problem to determine the number of clusters. First of all, there is no universal agreement on the definition of the concept "cluster" (Section 3.1, Everitt 1974). Different people may have different points of view on what constitutes a cluster and hence on how many clusters exist. Moreover, the data usually are in high-dimensional spaces so that visualization is not easy, and it is usually hard to detect their cluster structures from lower dimensional spaces. Even though we can view how many clusters appear in a a lower dimensional space, it is still possible that the true number of clusters is more than we can view. To illustrate this, we generate a simulated data set, which consists of 4 well-separated elliptical-shaped clusters in a 4-dimensional space. The pairwise scatter plot is shown in Figure 4.1. We can see that only 3 obvious clusters. Furthermore, factors such as shapes, sizes, and orientations of clusters, the degree of separation among clusters, noisy variables, and outliers may cause difficulties to determine the number of clusters.

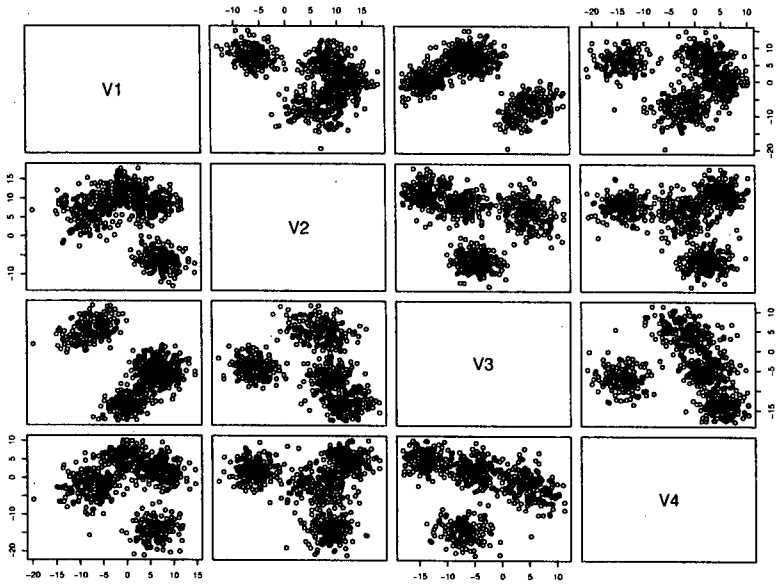


Figure 4.1: The pairwise scatter plot of a data sets containing 4 well-separated clusters in a 4-dimensional space. The 4-cluster structure is not obvious from the plot.

Many methods have been proposed to determine the number of clusters. These methods can be grouped roughly into two categories: subjective and objective methods.

One example of a subjective method is to use the subject matter knowledge of the data set. Sometimes the experts in the subject fields may provide prior information on the number of clusters. This subject knowledge is very useful to give a rough range of the number of clusters.

Another example of a subjective method is based on a plot of the within-cluster sum of squares versus the number of clusters. The number of clusters is determined subjectively by locating the "elbow" point in the plot. The rationale is that although the value of the within-cluster sum of squares is always monotone decreasing as the number of clusters increases, the decreases after the true number of clusters tends to be not as steep as those before the true number of clusters. Recent studies of the elbow phenomenon include Sugar (1998) and Sugar et al. (1999).

Tibshirani et al. (2001) and Sugar and James (2003) modified the above subjective method so that the number of clusters can be determined relatively objectively. Tibshirani et al. (2001) compared the curve based on the original data set with that based on reference data sets and proposed a gap statistic based on these two curves. The maximum point of the gap statistic is regarded as the number of clusters. This method is computationally intensive. Moreover, it may

not be easy to specify an appropriate reference distribution. Sugar and James (2003) assigned an appropriate negative power to a criterion called the minimum achievable distortion. They defined a jump statistic as the difference between the minimum achievable distortion at the current number of clusters and that at the previous number of clusters. The number of clusters is then determined by finding the maximum point of the jump statistic. Sugar and James's (2003) method has nice properties. However these nice properties depend on the assumption that all cluster covariance matrices have the same shape.

Actually, the methods proposed by Tibshirani et al. (2001) and Sugar and James (2003) belong to a sub-category of the objective methods. The common feature of the methods in this sub-category is that a non-monotone function of the number of clusters is first defined. Then the number of clusters is determined by finding the maximum or minimum point of the function. Milligan and Cooper (1985) compared 30 methods in this sub-category by a Monte Carlo study. Dubes (1987) and Kaufman and Rousseeuw (1990) also proposed methods belonging to this sub-category.

Peña and Prieto (2001) proposed another version of the concept of the gap statistic to determine the number of clusters. Data points are first projected into a one-dimensional space. The gap statistics are defined as the distances between the adjacent projected points. The number k_0 of large gap statistics in the interior of the projected points indicates that there exist $k_0 + 1$ clusters.

Mode detection or bump hunting methods (e.g. Cheng and Hall, 1998) has also been used to determine the number of clusters.

Peck et al. (1989) applied bootstrapping techniques to construct an approximate confidence interval for the number of clusters.

Many researchers assume that the data are from a mixture of distributions and determine the number of clusters by estimating the number of components in the mixture of distributions. References include Bozdogan (1993), Zhuang et al. (1996), Richardson and Green (1997), Fraley and Raftery (1998, 2002), Stephens (2000), and Schlattmann (2002). To determine the number of clusters, Bozdogan (1993) proposed the informational complexity criterion; Zhuang et al. (1996) proposed the Gaussian mixture density decomposition (GMDD) method; Richardson and Green (1997) used the reversible jump Markov chain Monte Carlo methods; Fraley and Raftery (1998, 2002) used the BIC criterion; Stephens (2000) used the Markov birth-death process; and Schlattmann (2002)

used bootstrap techniques. Frigui and Krishnapuram (1999) gave a brief review of robust clustering methods based on the assumption of the mixture of distributions when the number of clusters is unknown.

Another sub-category of the objective methods is to use data sharpening techniques. The idea is to iteratively shrink data points toward dense areas. The number of clusters corresponds to the final number of converged points. There are mainly two approaches in this sub-category. The first approach, gravitational clustering methods, is from the physical law point of view (Wright 1977; Kundu 1999; Sato 2000; and Wang and Rau 2001). The second approach, mean shift methods, is from the point of view of non-parametric density estimation (Fukunaga and Hostetler 1975; Cheng 1995;, Comaniciu and Meer 1999, 2000, 2001, 2002; and Wang et al. 2003).

Instead of shrinking data points toward to dense areas to merge data points, competitive cluster merging methods merge data points by changing their fuzzy memberships. These methods first obtain a fuzzy partition with an over-specified number of clusters. Then clusters compete for the data points based on the cluster sizes via a penalized fuzzy c-means method. The larger a cluster is, the more power this cluster has to compete for data points. Clusters whose sizes are smaller than a threshold will be deleted. The final partition gives the number of clusters. References on competitive cluster merging methods include Krishnapuram and Freg (1992), and Frigui and Krishnapuram (1997, 1999).

Lin and Chen (2002) proposed a simpler version of the cluster merging idea based on the cohesion index they defined which measures the joinability (similarity) of two clusters. Like competitive cluster merging methods, Lin and Chen (2002) first get a partition with an over-specified number of clusters. Then two clusters are merged if they have the largest cohesion index among all pairwise cohesion indexes and their cohesion index is larger than a specified threshold.

Cluster merging algorithms are intuitively appealing. However they require that the initial number of clusters be larger than the true number of clusters. It is not a problem to over-specify the number of clusters if subject knowledge is available to provide a rough idea about the true number of clusters. However, if there is no such subject knowledge available, then the initial number of clusters may be too small. It is more reasonable to allow both splitting and merging. In this way, the initial number of clusters can be under-specified.

The idea of using both splitting and merging to simultaneously determine the number of

clusters and do clustering can be dated back to the 1960's. Anderberg (1973) and Gnanadesikan (1977) mentioned a stepwise clustering method called the Iterative Self-Organizing Data Analysis Techniques A (ISODATA), which was originally proposed by Ball and Hall (1965). The ISODATA method has been extensively studied and been applied mainly to the image analysis and remote sensing areas (e.g. Simpson et al. 2001; Huang 2002b). The ISODATA method is simple and intuitively appealing. While it is flexible to have several thresholds to adjust, the user has to choose suitable values for these thresholds by trial and error based on a priori knowledge about the cluster structure (Carman and Merickel, 1990; Huang, 2002a). Moreover, the criteria for merging and splitting usually are different and the splitting criterion usually is based solely on the within-cluster variation without considering if the two subclusters from splitting are really well-separated.

In this chapter, we propose a sequential clustering (SEQCLUST) method that improves the ISODATA method. The main improvements include:

- The SEQCLUST method produces a sequence of estimated number of clusters based on varying input parameters. The most frequently occurring estimates in the sequence lead to a point estimate of the number of clusters with an interval estimate.

- The same criterion is used to determine if two clusters should be merged or a cluster should be split. And the criterion is based directly on the degree of separation among clusters.

Since the SEQCLUST method is closely related to the ISODATA method, we describe the details of the ISODATA method and several improvements in Section 4.2. In Section 4.3, we give an overview of the SEQCLUST method. We discuss the issue of initialization of the input parameters in Section 4.4. Some pre-processes, such as the process to adjust the initial number of clusters and the tuning parameter α, are described in Section 4.5. The merging method and the splitting method are introduced in Section 4.6 and in Section 4.7 respectively. The post-processes of the SEQCLUST method, such as the outlier-cluster detection, are described in Section 4.8. We check the performance of the SEQCLUST method by comparing it with several other easy-to-implement number-of-cluster-estimation methods through simulated and real data sets in Section 4.9. Further discussion is given in Section 4.10.

4.2 ISODATA Method

The ISODATA method can estimate the number of clusters and obtain a partition of a data set simultaneously. Although the user still has to input an initial number of clusters, the ISODATA method in principle does not depend much on the initial number of clusters. If the initial number is less than the true number, then we expect the ISODATA method could obtain the correct number of clusters by splitting iterations. If the initial number is larger than the true number, then we expect the ISODATA method could obtain the correct number by merging iterations. Although some software packages such as `Geomatica`[1] implement the ISODATA method, it seems that the ISODATA method has not being implemented into common statistical software (e.g. `SAS`, `R/Splus`, and `SPSS`) yet.

Since Ball and Hall (1965) proposed ISODATA, many variants have been proposed to improve its performance. We first illustrate ISODATA by the variant described in Anderberg (1973), which consists of the following steps:

ISODATA Method:

Step 1 Initialize the input parameters: the initial number of clusters k_0, `ITERMAX`, `NCLST`, `NWRDSD`, `THETAC`, `THETAE`, and `THETAN`.

ITERMAX is the maximum allowable cycles of the merging and splitting.

NCLST is a upper bound of the number of iterations in the lumping step.

THETAN is a threshold used to decide if a cluster is too small to be discarded.

THETAC is a threshold on between-cluster distance to decide in merging.

NWRDSD is used to decide on thresholds for splitting and merging cluster.

THETAE is a scalar threshold on sample standard deviations to decide in splitting.

Step 2 Obtain an initial k_0-cluster partition of the data set using a method such as `kmeans`.

Step 3 Delete outlier clusters whose sizes are smaller than the threshold `THETAN`. Update the number of clusters k_0.

[1] http://www.pcigeomatics.com/cgi-bin/pcihlp/ISOCLUS

Step 4

- If $k_0 \geq 2 \times$ NWRDSD, then a *lumping* iteration is invoked.
- If $k_0 \leq$ NWRDSD/2, then a *splitting* iteration is invoked.
- If NWRDSD/2 $< k_0 <$ 2 $\times$ NWRDSD and if the current iteration number is odd, then a *splitting* iteration is invoked.
- If NWRDSD/2 $< k_0 <$ 2 $\times$ NWRDSD and if the current iteration number is even, then a *lumping* iteration is invoked.

Update the number of cluster k_0.

Step 5 Obtain a k_0-cluster partition.

Step 6 Repeat steps 3, 4, and 5 until these three steps have been repeated ITERMAX times or the convergence criterion (e.g. a full cycle with no changes in cluster membership) is satisfied.

A *lumping* iteration consists of the following steps:

Lumping Iteration:

Step L1 Calculate the pairwise distances d_{ij}, $i, j = 1, \ldots, k_0$, among the cluster centers.

Step L2 Denote $(i_0, j_0) = \arg\min_{i \neq j} d_{ij}$. If $d_{i_0 j_0} <$ THETAC, then the cluster i_0 and j_0 are merged.

Step L3 If the iteration number is greater than NCLST or there is no merge, then stop the lumping iteration. Otherwise go back to Step L1.

We can see that the merging criterion depends solely on the distances among cluster centers. The variations in shapes/sizes of clusters are not involved. Moreover, without prior information, it is difficult to set the thresholds THETAC and NCLST.

A *splitting* iteration consists of the following steps:

Splitting Iteration:

Step S1 Calculate the sample standard deviation s_j^* for each variable, $j = 1, \ldots, p$, where p is the number of variables.

Step S2 Calculate the sample standard deviations s_{kj} of the k-th cluster in the j-th dimension, $k = 1, \ldots, k_0$, $j = 1, \ldots, p$, where k_0 is the number of clusters.

Step S3 Let $S = \{(k, j) : s_{kj} > \texttt{THETAE} \times s_j^*,\ k = 1, \ldots, k_0, j = 1, \ldots p\}$. Stop the splitting iteration if S is empty.

Step S4 For $(k, j) \in S$,

Step S4.1 Calculate the center m_{kj} of the cluster k in the j-th dimension.

Step S4.2 The data points in the cluster k are split into two subclusters according to whether their j-th coordinates are larger than or smaller than m_{kj}.

Step S5 Go back to Step S2.

We can see that the splitting criterion depends merely on the within-cluster variations without checking if there exists a gap or sparse area between the two subclusters or not.

Gnanadesikan (1977) mentioned another variant in which the splitting method is modified. To determine if a cluster should be split or not, the cluster is first projected along the direction of the first principal component. If the variance of the projected data is larger than a threshold, then the cluster is split into two subclusters along the projected direction.

Simpson et al. (2000) determined if a cluster should be split or not based on its diameter. The diameter is the distance between two carefully selected data points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. If the diameter is larger than a specified threshold, then the cluster is split into two subclusters. Two subclusters are formed around the two carefully selected data points by applying the nearest neighbor rule. That is, the data points which are closer to $\boldsymbol{x}_1$ form the subcluster 1 and the data points which are closer to $\boldsymbol{x}_2$ form the subcluster 2. The merging criterion is still based on the pairwise distances among cluster centers. The initial number of clusters is 1.

The splitting criterion in Huang (2002) is based on cluster sizes and the user-specified upper bound of the number of clusters. If the size of a cluster is greater than $P\%$ of the total size and if the current number of cluster is less than the upper bound, then the cluster is split to two subclusters. To split a cluster into two subclusters, a hyperplane is first obtained. The hyperplane passes through the cluster center and is perpendicular to the line connecting the cluster center and the data point which has the largest distance to the cluster center. The hyperplane splits the

cluster into two parts. Huang (2002) then used the sample means of the two parts as the two initial cluster centers for the k-means algorithm.

The above improvements on the splitting criterion still consider only the within-cluster variations and may split an elongated cluster into two subclusters.

Carman and Merickel (1990) used Consistent AIC (CAIC) to determine if clusters should be merged or split. A cluster is to be split if the CAIC value after the splitting is smaller than that before splitting. Two clusters are to be merged if the CAIC value after the merging is smaller than the value obtained before merging. To split a cluster into two subclusters, Carman and Merickel (1990) directly used the k-means algorithm. The initial number of clusters is 1. (We point out in Appendix B that there is a mistake in the CAIC formula used in Carman and Merickel (1990)).

Carman and Merickel's (1990) improvement is easier to use than other improvements in that no threshold is required for merging and splitting processes. Hence their improvement can do merging and splitting automatically. Also Carman and Merickel's improvement requires only a single criterion instead of two separate criteria to determine if the merging or splitting is attempted.

However, we found that Carman and Merickel's improvement usually overestimates the number of clusters (see Appendix B). One possible reason is that the CAIC does not directly measure the degree of separation among clusters.

4.3 SEQCLUST Method

In this section, we propose a sequential clustering method, SEQCLUST, based on the degree of separation among clusters.

The SEQCLUST method is based on the assumptions that (1) clusters are compact or only slightly overlapped, that is, there exist gaps or sparse areas among clusters; (2) data points in a cluster are concentrated more toward the cluster center; (3) clusters are convex in shape. The main idea of the SEQCLUST method is to directly use the magnitude of the gap or sparse area between the two (sub)clusters to decide if two clusters should be merged into one cluster. The key issues include (a) determining the threshold of the gap magnitude below which the two clusters should be merged and (b) making the SEQCLUST method less sensitive to the choice of the input parameters. It is desirable that the threshold could be automatically determined by the data themselves. The

features of the SEQCLUST method include:

1. Estimation of the number of clusters and clustering can be done simultaneously.
2. Both the merging and splitting criteria are based on the degree of separation between the pair of (sub)clusters.
3. It provides information about the stability of the estimated number of clusters, and an interval estimate.

This section, we give the overall algorithm for the SEQCLUST method. We will describe the details later. The main steps of the SEQCLUST method are given below:

SEQCLUST Method:

Step 1 Initializing the input parameters (Section 4.4)

Step 2 Pre-processing (Section 4.5)

Step 3 Splitting and merging until stable (Sections 4.7 and 4.6)

Step 4 Post-processing (Section 4.8)

The generic version of Step 2 contains the following sub-steps:

Pre-processing:

Step PRE1 (Optional) variable scaling and variable selection/weighting.

Step PRE2 Adjust the initial number k_0 of clusters so that it is neither too large nor too small (Subsection 4.4.1).

Step PRE3 Apply a clustering method to get a partition P with k_0 clusters.

Step PRE4 Invoke the merging process.

Step PRE5 If after merging, there exists only one cluster, then we enlarge the value of the tuning parameter α (See Equation 2.3.4) to allow more overlap, and invoke the merging process for the partition P again. Update P and k_0. Stop if k_0 is 1.

Step PRE6 Output k_0, α and P.

The generic version of the merging process contains the following sub-steps:

Merging:

Step M1 Obtain a matrix of pairwise separation indexes and obtain the merging matrix that indicates which pairs of clusters are eligible to be merged.

Step M2 Determine which eligible pairs of clusters should be merged. Merge these clusters.

Step M3 If the number of clusters does not change or there exists only one cluster, then stop merging. Otherwise go back to Step M1.

In Step M1, we do not need to re-compute the separation indexes among clusters which are not merged in the previous step.

The generic version of the splitting method contains the following sub-steps:

Splitting:

Step S1 Obtain the diameters of the k_0 clusters.

Step S2 Denote S as the set of clusters whose diameters are "close" to the maximum diameter.

Step S3 For each cluster in the set S

(1) Obtain a 2-cluster partition with two subclusters.

(2) If the two subclusters have a small separation index, then we do not split the cluster. Otherwise split the cluster to these two sub-clusters.

Step S4 Determine the number of clusters k_0^*. If $k_0^* = k_0$, then stop splitting. Otherwise $k_0 \leftarrow k_0^*$ and go back to Step S1.

The generic version of Step 4 contains the following sub-steps:

Post-processing:

Step POST1 Check if there are outlier clusters which have small numbers of data points relative to non-outlier clusters

Step POST2 (optional) Outlier detection

Step POST3 Denote k_0^* the final estimated number of clusters and P_1 as the corresponding partition. Obtain another k_0^*-cluster partition P_2 by the clustering algorithm specified in Step 1 of the SEQCLUST method. If the minimum separation index of P_1 is larger than that of P_2, then P_1 is output as the final partition. Otherwise P_2 is output as the final partition.

In the following sections, we outline one implementation of the SEQCLUST method.

4.4 Initializing the Input Parameters

The input parameters include the initial number of clusters, a sequence of the tuning parameter, variable scaling indicator, and the clustering algorithm etc.. Different settings of the input parameters may produce different estimates of the number of clusters and/or different partitions. We try to find a way to make the SEQCLUST method less sensitive to the initial input parameters.

4.4.1 The Initial Number of Clusters

Usually, practitioners have subject knowledge about the data sets and hence have a rough idea about the range of the number of clusters. Therefore it would be not a problem for them to provide an initial estimate of the number of clusters.

In case that one does not have any idea about the number of clusters, it might not be good to randomly choose an initial number of clusters. If k_0 is too far away from the "true" number of clusters, it will increase the computation time. Moreover, it will affect the precision of the estimated number of clusters. For example, if k_0 is too small, then the clusters obtained might not be separable and the SEQCLUST method might end up with a one-cluster partition. If k_0 is too large, then the clusters obtained might be too sparse to be able to be merged together. For example, for two singleton clusters (containing only one data point), the value of the separation index is 1. Hence the two singleton clusters could not be merged.

To automatically get a good initial estimate of the number of clusters, we use Calinski and Harabasz (CH) index (Calinski and Harabasz, 1974):

$$\mathrm{CH}(k) = \frac{\mathrm{tr}\,(\boldsymbol{B}_k)/(k-1)}{\mathrm{tr}\,(\boldsymbol{W}_k)/(n-k)}, \tag{4.4.1}$$

where k is the number of clusters, $\boldsymbol{B}_k$ and $\boldsymbol{W}_k$ are respectively between- and within-groups sum of squares

$$\begin{aligned}
\boldsymbol{B}_k &= \sum_{k=1}^{k_0} n_k(\bar{\boldsymbol{x}}_k - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_k - \bar{\boldsymbol{x}})^T, \\
\boldsymbol{W}_k &= \sum_{k=1}^{k_0}\sum_{i=1}^{n_k}(\boldsymbol{x}_{ki} - \bar{\boldsymbol{x}}_k)(\boldsymbol{x}_{ki} - \bar{\boldsymbol{x}}_k)^T, \\
\bar{\boldsymbol{x}} &= \frac{\sum_{k=1}^{k_0}\sum_{i=1}^{n_k}\boldsymbol{x}_{ki}}{\sum_{k=1}^{k_0} n_k}, \quad \bar{\boldsymbol{x}}_k = \frac{1}{n_k}\sum_{i=1}^{n_k}\boldsymbol{x}_{ki}.
\end{aligned}$$

The CH index measures the compactness of clusters. Clusters formed by the chosen algorithm for each k. One can use

$$k_0 = \arg\max_{k\geq 2} \mathrm{CH}(k)$$

as an estimate of the number of clusters[2]. Milligan and Cooper (1985) reported that the CH index has good performance on deciding the number of clusters. Tibshirani et al. (2001) and Sugar and James (2003) also studied the performance of the CH index. Due to its simplicity, we use the CH index to get an initial number of clusters.

The function $\mathrm{CH}(k)$ does not always have unique maximum points. To save computing time, we use

$$k_{\max 1} + 10$$

as an initial number of clusters, where $k_{\max 1}$ is the smallest local maximum point of $\mathrm{CH}(k)$. The reason that we add 10 to $k_{\max 1}$ is to avoid underestimating the number of clusters. In our experience, an under-specified initial number of clusters is worse than an over-specified initial number of clusters. The choice of the value 10 is somewhat arbitrary. Other values can be used.

4.4.2 The Initial Tuning Parameter

In the implementation of the SEQCLUST method we describe in this chapter, we use the separation index we proposed in Chapter 2 to measure the magnitude of the gap between a pair of clusters. The tuning parameter α in the separation index reflects the percentage in the two tails of the 1-dimensional projection of a cluster that might be outlying. It might affect the estimation of the number of clusters when some clusters are close to each other.

[2] CH(1) is not defined.

If the cluster structure is quite obvious, then we expect that we can get the same estimate of the number of clusters for a range of values of the tuning parameter α. So instead of one value, we input a sequence of values of the tuning parameter to obtain a sequence of estimates of the number of clusters and an interval estimate consisting of the whole range. Then the most frequently occurring estimate in this sequence, i.e. the mode of the estimate sequence, is regarded as the final estimate of the number of clusters. The variation of this estimation sequence provides information about the stability of the final estimate of the number of clusters. The smaller the variation is, the more stable the final estimated number of clusters is.

The sequence of estimates can also provide information about the lower and upper bound on the number of clusters. For example, if the sequence of estimates is $\{2, 2, 3, 3, 3, 3, 5\}$, then the final estimated number of clusters is 3. And we can regard $[2, 5]$ as an interval estimate.

4.4.3 Variable Scaling

In Chapter 2, we show that the separation index J^* is theoretically scale invariant. However, scaling variables may distort the cluster structure and hence affect the performance of clustering algorithms (Milligan and Cooper 1988; Schaffer and Green 1996). The SEQCLUST method requires a clustering algorithm to get an initial partition and to get the two-cluster sub-partition in the splitting process. So scaling variables may affect the performance of the SEQCLUST method.

The SEQCLUST method allows the user to determine if scaling is needed or not. If the user does not make the decision, then the SEQCLUST method will use a very simple way to automatically determine if scaling is required or not. The criterion is that if the ratio of the maximum sample standard deviation to the minimum sample standard deviation is larger than 3, then the variables will be scaled so that the mean and standard deviation of each variable are 0 and 1 respectively.

4.4.4 The Clustering Algorithm

In principle, any clustering algorithm can be used in the SEQCLUST method to obtain the initial partition and two-cluster partitions in the splitting process. However, in practice, we need to consider the availability, performance, and speed of clustering algorithms.

We use the statistical software `R` to implement our ideas. There are several clustering

algorithms implemented in R, such as partitioning algorithms (kmeans, PAM, and CLARA) hierarchical clustering algorithms (Ward), and model-based clustering algorithm (EMclust and Mclust). All these algorithms can get good partitions for spherical-shaped clusters provided that the correct number of clusters is given.

The kmeans algorithm is sensitive to the choice of initial cluster centers. So we modified it by running the kmeans algorithm T times (e.g. $T = 10$). Each time, the kmeans algorithm is provided with a set of randomly selected cluster centers. Then we choose the partition whose average within-cluster sum of squares is the smallest. We denote this modified kmeans algorithm as MKmeans.

The PAM algorithm has a special mechanism to choose initial cluster centers and implements robust techniques so that the resultant partitions are stable and robust. However PAM is quite slow when handling large data sets. The CLARA algorithm improves the speed of the PAM algorithm by a sampling technique. We denote PAM/CLARA as the mixed algorithm that PAM is used if the number of data points n in the data set is less than or equal to 200; otherwise the CLARA method is used.

The Mclust algorithm is a model-based agglomerative hierarchical clustering algorithm which successively merges pairs of clusters corresponding to the greatest increase in the classification likelihood among all possible pairs (Fraley and Raftery, 2002). Different covariance structures correspond to different models. The EMclust algorithm provides more flexibility including the choice of models and also can do clustering based on a sample of the data set. The R functions EMclust and Mclust can allow the user to input a sequence of the number of clusters and select an optimal partition by using BIC criterion. To distinguish different usages of EMclust and Mclust, we introduce the notation:

EMclust0: obtains a k_0-cluster partition by using the R function EMclust. If the total number of clusters is greater than 500, then a sample of 500 data points is used to get the cluster centers. Otherwise, all data points are used.

Mclust0: obtains a k_0-cluster partition by using the R function Mclust. All data points are used.

EMclust1: obtains an optimal estimate of the number of clusters from a sequence of candidate number of clusters by using BIC model selection procedure. The R function EMclust is used. If the total number of clusters is greater than 500, then a sample with size 500 is used to get

the cluster centers. Otherwise, all data points are used.

`Mclust1`: obtains an optimal estimate of the number of clusters from a sequence of candidate number of clusters by using BIC model selection procedure. The R function `Mclust` is used. All data points are used.

The current implementation of the SEQCLUST method allows the user to choose a clustering algorithm from `kmeans`, `MKmeans`, `PAM/CLARA`, `Ward`, `EMclust0`, and `Mclust0`. The clustering methods `EMclust1` and `Mclust1` will be used in Sections 4.9.3 and 4.9.4 to compare the performance with the SEQCLUST method.

To have a rough idea about the speed of these clustering algorithms, we conducted a simple simulation study. In this simulation study, we generate data sets from the bivariate normal distribution $N(\mathbf{0}, I_2)$. For each sample size, we generate 100 data sets. Then we get a 2-cluster partition for each data set. The system times used to get the 2-cluster partition are recorded. Then we obtain the averages of the 100 system times [3] which are summarized in Table 4.1 and Figure 4.2. From Table 4.1 and Figure 4.2, we can see that `kmeans`, `PAM/CLARA`, and `MKmeans` are fast while

Table 4.1: Average system time (seconds)

sample size	kmeans	MKmeans	PAM/CLARA	EMclust0	Ward	Mclust0
100	0.000	0.007	0.009	0.097	0.006	0.0905
500	0.001	0.012	0.006	0.906	0.220	0.9361
1000	0.001	0.018	0.010	1.085	0.716	5.0111
1500	0.003	0.028	0.008	1.242	1.388	14.7580
2000	0.003	0.037	0.008	1.399	2.072	32.4534
2500	0.005	0.047	0.009	1.560	2.968	
3000	0.006	0.056	0.009	1.749	4.091	
3500	0.006	0.064	0.009	1.902	5.359	
4000	0.008	0.078	0.010	2.081	6.992	
4500	0.009	0.089	0.010	2.241	8.827	
5000	0.010	0.100	0.011	2.433	10.866	

`EMclust0`, `Ward`, and `Mclust0` are relatively slow.

[3] I ran my R programs in a computing system with dual CPU (AMD Athlon(TM) MP 2000+) and 1Gb RAM. The operation system is Linux with the Redhat 9.0 distribution.

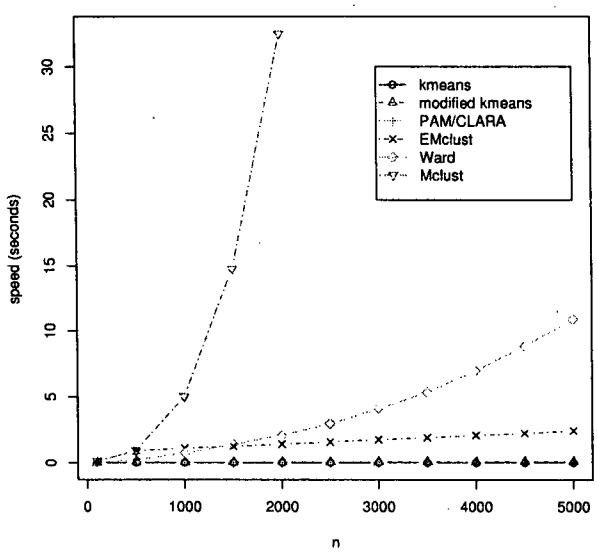


Figure 4.2: Plot of the average system time (seconds) versus the sample size.

4.5 Pre-Processing

The main task in the pre-processing step is to adjust the initial estimate of the number of clusters and the value of a tuning parameter α.

If the initial estimated number $\hat{k}_0$ of clusters is too large, then some cluster sizes will be too small to be merged since the data points will be relative sparse if cluster sizes are small. To avoid small cluster sizes, we require that the minimum cluster size be larger than a threshold like 30. If the minimum cluster size is smaller than `sizethr`= 30, then we reduce the initial estimate of the number of clusters by half ($\hat{k}_0 \leftarrow \hat{k}_0/2$) until the minimum cluster size is larger than 30 or the initial estimate of the number $\hat{k}_0$ of clusters is equal to 1. Instead of threshold on the number of data points, we can use threshold of the percentage of the number of data points.

It is also not desirable that the initial estimated number of clusters is too small since in this case we might end up with only one cluster for multi-cluster-structure data sets. Figure 4.3 illustrates that if the initial estimated number of clusters is 1, then we could not accept the 2-cluster split since the two sub-clusters are not separated. Hence we end up with one-cluster structure instead of 12-cluster structure.To alleviate this problem, we can increase the value of the tuning parameter α to allow more overlap (see Step PRE4 in page 65).

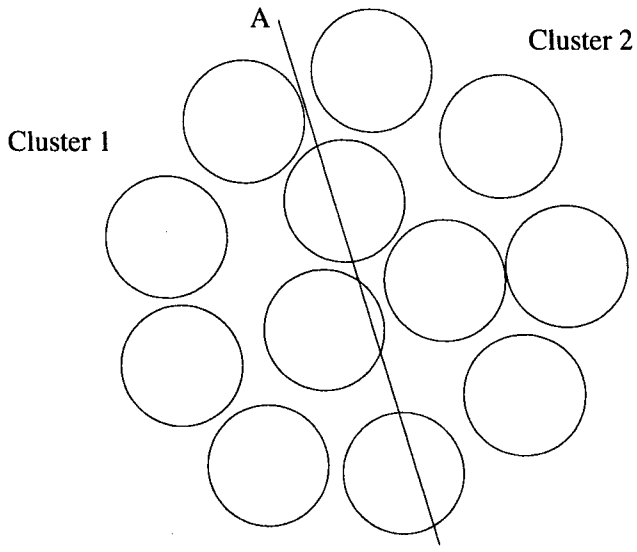


Figure 4.3: An example illustrates the need to avoid too small initial estimate of the number of clusters.

Variable selection/weighting can eliminate or reduce the effect of noisy variables which may mask the true cluster structures (see Chapter 5). So we incorporate the variable selection/weighting procedure to the pre-process stage of the SEQCLUST method. In this chapter, we apply SEQCLUST assuming there are no noisy variables in data sets. We will discuss how to handle noisy variables in Chapter 5.

4.6 Merging

The merging process consists of three parts: (1) checking which pairs of clusters are eligible to be merged — denote S as the set of the eligible pairs; (2) deciding a subset S_1 of eligible pairs in S to be actually merged; (3) developing an algorithm to merge the pairs of clusters in S_1.

4.6.1 Mergable Pairs of Clusters

In the SEQCLUST method, whether a pair of clusters is eligible to be merged depends on the degree of separation between the two clusters. There are many ways to measure the degree of separation between two clusters (see Chapter 2). In the implementation of the SEQCLUST method, we use the separation index we proposed in Chapter 2 to measure the degree of separation between the two clusters.

We may simply allow two clusters to be merged if their separation index is negative. When the value of the separation index is close to zero, 0.01 say, there are two possibilities. One is that two clusters are touching or separated. The other is that the two clusters actually overlap and the positive value is due to the randomness of the data. To handle this situation, we can either construct an asymptotic $100(1-\alpha_0)\%$ confidence lower bound [4] $J_L^*(\alpha_0)$ to be described in Section 4.6.3 for the separation index or set a threshold J_T^*. If $J_L^*(\alpha_0) > 0$ or $J^* > J_T^*$, then we do not merge two clusters. Otherwise, we regard the two clusters as eligible cluster pair to be merged.

It is relatively objective to use an asymptotic confidence lower bound to determine if two clusters are eligible to be merged or not. In addition to the location information, the asymptotic confidence lower bound also uses the variation information about the two clusters. However the precision of the asymptotic confidence lower bound depends on sample sizes, outliers, and distributional assumption of the linear projection. The threshold is less sensitive to sample sizes, outliers, and the distributional assumption. However it does not use the variation information at all. Moreover different data sets may need different thresholds. The current implementation of the SEQCLUST method automatically determines if $J_L^*(\alpha_0)$ or J_T^* should be used by the following criterion:

Merging Criterion:

(1) If the normal version of the separation index $J^* > 0$ (see page 16) and its corresponding asymptotic confidence lower bound $J_L^*(\alpha_0) > 0$, then we do not merge the two clusters.

(2) Otherwise, we calculate the quantile version J_q^* (see page 14) of the separation index. If $J_q^* > J_T^*$, then we do not merge the two clusters. Otherwise, we merge the two clusters.

Given a partition C, we can obtain a **merging indicator** matrix $M_{k_0 \times k_0}$ based on the above merging criterion. $M_{ij} = 1$ if we do not merge clusters i and j. Otherwise $M_{ij} = 0$.

Definition 4.6.1 *Cluster i and j are called* ***directly mergable*** *if $M_{ij} = 0$.*

Because direct mergability is not a transitive operation, we need the following two definitions and a complicated merging algorithm.

[4] We use the notation α_0 to distinguish the confidence level from the tuning parameter α used in the separation index (See Definition 2.3.4).

Definition 4.6.2 *Cluster i and j are called* **indirectly mergable** *if $M_{ij} = 1$ and there exist a sequence $\{k_1, \ldots, k_\ell\}$ of clusters such that $M_{ik_1} = 0$, $M_{k_1k_2} = 0$, $\ldots$, $M_{k_\ell j} = 0$.*

Definition 4.6.3 *Cluster i and j are called* **mergable** *if cluster i and j are either directly mergable or indirectly mergable.*

Once we obtain the set of mergable cluster pairs, we have at least two choices: (1) merge the mergable cluster pair whose separation index is the smallest; (2) merge all **mergable** pairs. We take the second choice to increase the speed of the SEQCLUST method.

4.6.2 Merging Algorithm

In this subsection, we propose an algorithm to merge all **mergable** pairs of clusters. The difficulty is how to get the disjoint sets of **mergable** pairs of clusters from the merging indicator matrix. We use matrix operations to move **mergable** clusters together and make use of the row and column names to record the information of the disjoint sets of **mergable** pairs of clusters.

Let B_k be the $k \times k$ matrix with diagonal elements equal to 0 and off-diagonal elements equal to 1. If the merging indicator matrix M_k has the form B_k, then no merging is needed.

The pseudo-code is given below:

Step 1 Initialize the row names and column names as $L(1)$ ="1", ..., $L(k)$ ="k". Set $t = 1$.

Step 2 If $M_k = B_k$, then go to Step 6. Otherwise go to Step 3.

Step 3 Denote $\boldsymbol{v}_t$ as the t-th row of the matrix M_k. Find the set $S = \{j : j > t, v_{tj} = 0\}$.

Step 4 If S is empty, then $t \leftarrow t + 1$ and go back to Step 3. Otherwise, go to Step 5.

Step 5 Move the $S[1]$-th row and column to the $(t+1)$-th row and column. Update the row names and column names of the $(t+1)$-th row and column $L(t+1) \leftarrow L(t) \cup \{S[1]\}$. Put in this new row/column the product of the $(t+1)$-th row with the t-th row and the $(t+1)$-th column with the t-th column. Delete the t-th row and column. $k \leftarrow k-1$. Go back to Step 2.

Step 6 The sets of mergable clusters are recorded in the row names or column names of the merging indicator matrix. Merge clusters in each set to one cluster.

We give an example to illustrate the merging algorithm. In the example, the initial merging indicator matrix M_5 is:

	"1"	"2"	"3"	"4"	"5"
"1"	0	1	1	0	0
"2"	1	0	0	1	1
"3"	1	0	0	1	1
"4"	0	1	1	0	1
"5"	0	1	1	1	0

At the first iteration, $t = 1$, $S = \{4, 5\}$, and $S[1] = 4$. We move the $S[1]$-th row and column to the $(t + 1)$-th row and column. The updated merging indicator matrix is

	"1"	"4"	"2"	"3"	"5"
"1"	0	0	1	1	0
"4"	0	0	1	1	1
"2"	1	1	0	0	1
"3"	1	1	0	0	1
"5"	0	1	1	1	0

We then update the row and column names of the $(t + 1)$ row and column. After we multiply the $(t + 1)$-th row and column with the t-th row and column respectively, the updated merging indicator matrix is

	"1"	"1 4"	"2"	"3"	"5"
"1"	0	0	1	1	0
"1 4"	0	0	1	1	0
"2"	1	1	0	0	1
"3"	1	1	0	0	1
"5"	0	0	1	1	0

We then delete the t-th row and column and the resultant merging indicator matrix is

	"1 4"	"2"	"3"	"5"
"1 4"	0	1	1	0
"2"	1	0	0	1
"3"	1	0	0	1
"5"	0	1	1	0

In the second iteration, t is still equal to 1. The row and column names are $L(1)$ ="1 4", $L(2)$ ="2", $L(3)$ ="3", and $L(4)$ ="5". The set S is equal to $\{4\}$ and $S[1] = 4$. We move the $S[1]$-th row and column to the t-th row and column. The resultant merging indicator matrix is

	"1 4"	"5"	"2"	"3"
"1 4"	0	0	1	1
"5"	0	0	1	1
"2"	1	1	0	0
"3"	1	1	0	0

We then update the row and column names of the $(t+1)$ row and column. we then multiply the $(t+1)$-th row and column with the t-th row and column respectively. After removing the t-th row and column, we obtain

	"1 4 5"	"2"	"3"
"1 4 5"	0	1	1
"2"	1	0	0
"3"	1	0	0

In the third iteration, $t = 1$ and S is empty. Then we set $t \leftarrow t + 1 = 2$. Now $S = \{3\}$. Since $t + 1 = 3 = S[1]$, we do not need to move the $S[1]$ row and column. We then update the row and column names of the $(t+1)$-th row and column. After deleting the t-th row and column, we obtain

	"1 4 5"	"2 3"
"1 4 5"	0	1
"2 3"	1	0

Now that $M_2 = B_2$, we stop iteration and merge clusters 1, 4, and 5 to new cluster 1 and clusters 2 and 3 to new cluster 2.

4.6.3 Asymptotic Properties of the Estimate of the Separation Index

In Subsection 4.6.1, we mention a normality-based asymptotic $100(1-\alpha_0)\%$ confidence lower bound of the separation index to check if a pair of clusters are mergable. In this subsection, we first study the asymptotic properties of the estimated separation index and then construct a normality-based asymptotic $100(1-\alpha_0)\%$ confidence lower bound $J_L^*(\alpha_0)$ of the separation index.

Given the projection direction $\mathbf{a}$, the definition of the separation index under the normality assumption is (Equation 2.3.4)

$$J^* = \frac{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) - z_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)}{\mathbf{a}^T(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1) + z_{\alpha/2}\left(\sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}} + \sqrt{\mathbf{a}^T\boldsymbol{\Sigma}_2\mathbf{a}}\right)},$$

where the tuning parameter $\alpha \in [0, 0.5]$ is in general different from the confidence level α_0 in the asymptotic confidence lower bound $J_L^*(\alpha_0)$.

Denote $\mu_i = \mathbf{a}^T\boldsymbol{\theta}_i$, $\tau_i^2 = \mathbf{a}^T\boldsymbol{\Sigma}_i\mathbf{a}$, $i = 1, 2$. Then the separation index becomes

$$J^* = \frac{(\mu_2 - \mu_1) - z_{\alpha/2}(\tau_1 + \tau_2)}{(\mu_2 - \mu_1) + z_{\alpha/2}(\tau_1 + \tau_2)}.$$

If we obtain the Maximum Likelihood Estimates (MLEs) of μ_i and τ_i, $i = 1, 2$, then we can obtain the MLE $\hat{J}^*$ of J^* and know the asymptotic distribution of $\hat{J}^*$. Hence we can construct a $100(1-\alpha_0)\%$ confidence lower bound of the separation index.

We would like to emphasize that the asymptotic confidence lower bound derived in Subsection 4.6.3 is for any **fixed** projection direction $\boldsymbol{a}$. In practice, the projection direction $\mathbf{a}$ depends on the data points in the two clusters. Hence $\mathbf{a}$ is random instead of fixed. The sampling distribution of $J^*(\boldsymbol{a})$ for $\boldsymbol{a}$ fixed would be very different from the sampling distribution of $\max_{\boldsymbol{u}} J^*(\boldsymbol{u})$.

The purpose of the asymptotic confidence lower bound in the merging criterion is only as a less-subjective threshold of J^* to determine if two groups of data points should be merged or not. If the projection direction we choose is happen to be exactly equal to that obtained by the data, then it seems that the asymptotic confidence lower bound can be used as a threshold. So for simplicity we just regard that the projection direction $\mathbf{a}$ as fixed once we obtain it from data, although this might not be rigorous.

(1) MLE of $\hat{J}^*$

Suppose X_{1i}, $i = 1, \ldots, n_1$, are data points from cluster 1, where n_1 is the size of cluster 1, and X_{2j}, $j = 1, \ldots, n_2$, are data points from cluster 2, where n_2 is the size of cluster 2. Denote $x_{1i} = \mathbf{a}^T X_{1i}$, $i = 1, \ldots, n_1$, and $x_{2i} = \mathbf{a}^T X_{2i}$, $j = 1, \ldots, n_2$. Then under the normality assumption of the projections and the assumption that $\mathbf{a}$ is fixed, $x_{1i} \overset{i.i.d}{\sim} \mathrm{N}(\mu_1, \tau_1^2)$, $i = 1, \ldots, n_1$, and $x_{2j} \overset{i.i.d}{\sim} \mathrm{N}(\mu_2, \tau_2^2)$, $j = 1, \ldots, n_2$.

The density function of normal distribution $\mathrm{N}(\mu_k, \tau_k^2)$, $k = 1, 2$, are defined as

$$f(x_{ki}|\mu_k, \tau_k^2) = \frac{1}{\sqrt{2\pi}\tau_k} \exp\left\{-\frac{(x_{ki} - \mu_k)^2}{2\tau_k^2}\right\}.$$

and the log-likelihood function is

$$\ell(\mu_1, \tau_1, \mu_2, \tau_2) = \left[\sum_{i=1}^{n_1} \log\left(f(x_{1i}|\mu_1, \tau_1)\right)\right] + \left[\sum_{j=1}^{n_2} \log\left(f(x_{2j}|\mu_2, \tau_2)\right)\right]. \tag{4.6.2}$$

It is well-known that the MLEs of μ_i's and τ_i's are

$$\hat{\mu}_i = \bar{x}_i, \quad \hat{\tau} = s_i, \quad i = 1, 2,$$

where

$$\bar{x}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{ij}, \quad \text{and} \quad s_i = \sqrt{\frac{1}{n_i}\sum_{j=1}^{n_i}(x_{ij} - \bar{x}_i)^2}.$$

Thus, the MLE of J^* is

$$\hat{J}^* = \frac{(\bar{x}_2 - \bar{x}_1) - z_{\alpha/2}(s_1 + s_2)}{(\bar{x}_2 - \bar{x}_1) + z_{\alpha/2}(s_1 + s_2)}.$$

In the next subsection, we will derive the variance for the MLE $\hat{J}^*$.

(2) Asymptotic Variance of $\hat{J}^*$

Denote $\boldsymbol{\eta} = (\mu_1, \tau_1, \mu_2, \tau_2)^T$. Then the separation index J^* is a function of $\boldsymbol{\eta}$, i.e. $J^* = J^*(\boldsymbol{\eta})$. Let $\hat{\boldsymbol{\eta}} = (\hat{\mu}_1, \hat{\tau}_1, \hat{\mu}_2, \hat{\tau}_2)^T$, i.e., $\hat{\boldsymbol{\eta}}$ is the MLE of $\boldsymbol{\eta}$.

By the property of maximum likelihood estimates,

$$\sqrt{n}\left[J^*(\hat{\boldsymbol{\eta}}) - J^*(\boldsymbol{\eta})\right] \xrightarrow{d} \mathrm{N}\left(0, [\partial J^*(\boldsymbol{\eta}/\partial\boldsymbol{\eta}]^T I^{-1}(\boldsymbol{\eta})[\partial J^*(\boldsymbol{\eta}/\partial\boldsymbol{\eta}])\right),$$

where $I(\boldsymbol{\eta})$ is the Fisher Information matrix of the MLE $\hat{\eta}$:

$$I(\boldsymbol{\eta}) = -E\left[\frac{\partial^2 \ell(\boldsymbol{\eta})}{\partial \boldsymbol{\eta} \partial \boldsymbol{\eta}^T}\right],$$

and $\ell(\boldsymbol{\eta})$ is the log-likelihood function (4.6.2).

The second derivatives are

$$\begin{aligned}
\frac{\partial^2 \ell}{\partial \mu_1^2} &= -\frac{n_1}{\tau_1^2} \\
\frac{\partial^2 \ell}{\partial \tau_1^2} &= \frac{n_1}{\tau_1^2} - \frac{3}{\tau_1^4}\sum_{i=1}^{n_1}(x_{1i} - \mu_1)^2 \\
\frac{\partial^2 \ell}{\partial \mu_2^2} &= -\frac{n_2}{\tau_2^2} \\
\frac{\partial^2 \ell}{\partial \tau_2^2} &= \frac{n_2}{\tau_2^2} - \frac{3}{\tau_2^4}\sum_{j=1}^{n_2}(x_{2i} - \mu_2)^2 \\
\frac{\partial^2 \ell}{\partial \mu_1 \partial \tau_1} &= -\frac{2}{\tau_1^3}\sum_{i=1}^{n_1}(x_{1i} - \mu_1) \\
\frac{\partial^2 \ell}{\partial \mu_2 \partial \tau_2} &= -\frac{2}{\tau_2^3}\sum_{j=1}^{n_2}(x_{2j} - \mu_2) \\
\frac{\partial^2 \ell}{\partial \mu_1 \partial \mu_2} &= 0, \ \frac{\partial^2 \ell}{\partial \mu_1 \partial \tau_2} = 0, \ \frac{\partial^2 \ell}{\partial \tau_1 \partial \mu_2} = 0, \ \frac{\partial^2 \ell}{\partial \tau_1 \partial \tau_2} = 0.
\end{aligned}$$

The Fisher information matrix of $\hat{\boldsymbol{\eta}}$ is

$$I(\boldsymbol{\eta}) = \begin{pmatrix} n_1/\tau_1^2 & 0 & 0 & 0 \\ 0 & 2n_1/\tau_1^2 & 0 & 0 \\ 0 & 0 & n_2/\tau_2^2 & 0 \\ 0 & 0 & 0 & 2n_2/\tau_2^2 \end{pmatrix}$$

where $\boldsymbol{\eta} = (\mu_1, \tau_1, \mu_2, \tau_2)^T$.

We can get

$$\begin{aligned}
\frac{\partial J^*}{\partial \mu_1} &= -\frac{2z_{\alpha/2}(\tau_1 + \tau_2)}{\left[(\mu_2 - \mu_1) + z_{\alpha/2}(\tau_1 + \tau_2)\right]^2}, \\
\frac{\partial J^*}{\partial \tau_1} &= -\frac{2z_{\alpha/2}(\mu_2 - \mu_1)}{\left[(\mu_2 - \mu_1) + z_{\alpha/2}(\tau_1 + \tau_2)\right]^2}, \\
\frac{\partial J^*}{\partial \mu_2} &= \frac{2z_{\alpha/2}(\tau_1 + \tau_2)}{\left[(\mu_2 - \mu_1) + z_{\alpha/2}(\tau_1 + \tau_2)\right]^2}, \\
\frac{\partial J^*}{\partial \tau_2} &= -\frac{2z_{\alpha/2}(\mu_2 - \mu_1)}{\left[(\mu_2 - \mu_1) + z_{\alpha/2}(\tau_1 + \tau_2)\right]^2}.
\end{aligned}$$

By using the delta method, we can get the asymptotic variance of $J^*(\hat{\eta})$

$$\begin{aligned}\tau^2 &= [\partial J^*/\partial \eta]^T I^{-1}(\eta)[\partial J^*/\partial \eta] \\ &= \frac{4z^2_{\alpha/2}}{\left[(\mu_2-\mu_1)+z_{\alpha/2}(\tau_1+\tau_2)\right]^4}\left(\frac{\tau_1^2}{n_1}+\frac{\tau_2^2}{n_2}\right)\left[\frac{(\mu_2-\mu_1)^2}{2}+(\tau_1+\tau_2)^2\right]\end{aligned}$$

(3) Confidence Lower Bound of $\hat{J}^*$

To make sure that the $100(1-\alpha_0)\%$ confidence lower bound $J^*_L(\alpha_0) \in [-1,1)$, we first transform J^* to $h(J^*) \in (-\infty, \infty)$, where h is a continuous monotone increasing function such that h^{-1} exists. Then we get a $100(1-\alpha_0)\%$ confidence lower bound h_L for $h(J^*)$. Finally we can get a $100(1-\alpha_0)\%$ confidence lower bound $h^{-1}(h_L)$ for J^*.

The transformation we use is

$$h(J^*) = \tan\left(\frac{\pi}{2}J^*\right).$$

The reverse-transformation is

$$h^{-1}(J^*) = \frac{2}{\pi}\arctan(h).$$

Again using property of MLE

$$\sqrt{n}(h(\hat{J}^*) - h(J^*)) \xrightarrow{d} \mathrm{N}\left(0, \tau^2\left[h'(J^*)\right]^2\right),$$

where

$$h'(J^*) = \frac{\pi}{2}\frac{1}{\left[\cos(\frac{\pi}{2}J^*)\right]^2}.$$

Note that $h'(J^*) \geq 0$ since $\cos(\theta) \geq 0$ when $\theta \in [-\pi/2, \pi/2]$ and $J^* \in [-1,1)$. Hence

$$\frac{\sqrt{n}[h(\hat{J}^*) - h(J^*)]}{\tau h'(\hat{J}^*)} \dot{\sim} \mathrm{N}(0,1),$$

by using the fact that

$$h'(\hat{J}^*) \xrightarrow{p} h'(J^*).$$

From

$$P\left(\frac{\sqrt{n}[h(\hat{J}^*) - h(J^*)]}{\tau h'(\hat{J}^*)} \leq z_{\alpha_0}\right) = 1-\alpha_0$$

we can get approximately

$$P\left(h(\hat{J}^*) - \frac{\tau h'(\hat{J}^*) z_{\alpha_0}}{\sqrt{n}} \leq h(J^*)\right) = 1 - \alpha_0.$$

Therefore, a $100(1-\alpha_0)\%$ confidence lower bound of $h(J^*)$ is

$$h_L^* = h(\hat{J}^*) - \frac{\tau h'(\hat{J}^*) z_{\alpha_0}}{\sqrt{n}}.$$

Finally an approximate $100(1-\alpha_0)\%$ confidence lower bound of J^* is

$$J_L^*(\alpha_0) = \frac{2}{\pi} \arctan\left\{ \tan(\frac{\pi}{2}\hat{J}^*) - z_{\alpha_0} \frac{\pi\hat{\tau}}{2\sqrt{n}\left[\cos(\frac{\pi}{2}\hat{J}^*)\right]^2} \right\}, \tag{4.6.3}$$

We can see that the asymptotic confidence lower bound (4.6.3) is a monotone increasing function of the sample size n. This property makes sense. The larger the sample size is, the more information we can get and hence the closer the confidence lower bound is to the true value of the separation index. The asymptotic confidence lower bound (4.6.3) is also a monotone increasing function of the confidence level α_0. The larger α_0 is, the smaller $1-\alpha_0$ is and hence the smaller the confidence we have to make sure that the true value of the separation index is larger than the lower bound.

This formula for a "lower bound" will be reasonable to use even when $\boldsymbol{a}$ is random. It may not be very good if the projection gives distributions far from normal. We use a small example to illustrate this. In this example, there are 4 well-separated clusters in a two-dimensional space, each containing 50 data points from a multivariate normal distribution $\mathrm{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, \ldots, 4$, where

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_2 = \begin{pmatrix} 10 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_3 = \begin{pmatrix} 20 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_4 = \begin{pmatrix} 30 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma}_i = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, i = 1, \ldots, 4.$$

For the two clusters of a 2-split of the simulated data set obtained by `MKmeans`, the separation index is -0.01 ($\alpha = 0.05$) and the corresponding asymptotic 5% confidence lower bound is -0.015. A one-dimensional projection of the two clusters along the optimal projection direction is shown in Figure 4.4. Kernel densities are also shown in Figure 4.4. We can see that although the two clusters are well-separated, the separation index and its asymptotic confidence lower bound are negative. This is because the distributions of the two clusters are far from normal and hence the estimated variances of the two projected clusters could not capture the dispersion of the points well.

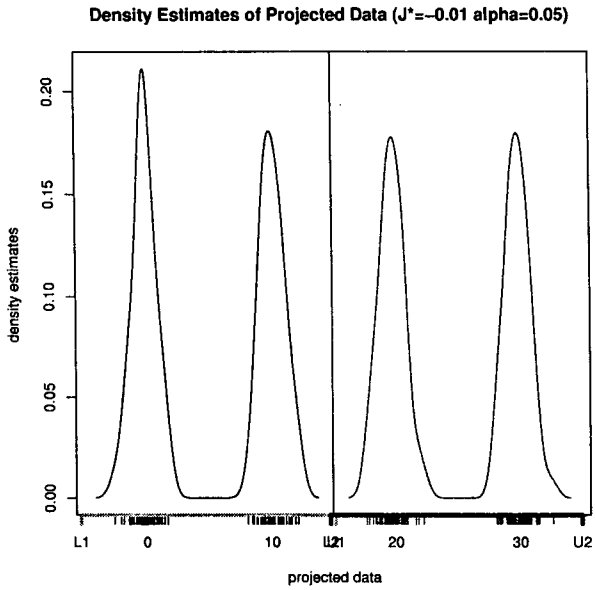


Figure 4.4: If the one-dimensional projections are far from normal, then the separation index and its asymptotic confidence lower bound may not be good.

This small example shows that it would be better to use both the normal-version and the quantile version of the separation index to check if two clusters should be merged or not. A possible merging criterion is the merging criterion in Section 4.6.1.

The choice of the value $J_T^* = 0.15$ is based on the fact that we assume that $J^* = 0.01$ if clusters are close to each other and $J^* = 0.21$ if clusters are separated from each other (see Section 3.3). So we want to choose a value between 0.01 and 0.21. The value we choose is $2 \times (0.01 + 0.21)/3 = 0.15$. As small change of this value will not have big effect.

4.7 Splitting

The purpose of the splitting process is to check if there exists sub-cluster structure in current clusters, that is, we want to know if there exist gaps within each current cluster.

We do not need to check the gaps for each current cluster if the cluster sizes are vary. The reasoning is that if we could not split the cluster with the maximum "diameter", then we likely could not split other clusters either. So we only check if the clusters with the largest diameters need to be split or not. Specifically, we try to split the clusters whose diameters ψ_k satisfy $(\psi_{\max} - \psi_k)/\psi_{\max} <$

0.1, where $\psi_{\max}$ is the largest diameter. The threshold 0.1 can be adjusted. There are many ways to define the diameter of a cluster. We measure the diameter of a cluster by the trace of its covariance matrix.

Once we obtain the clusters which are eligible for splitting, we can check if an eligible cluster can be split by dividing into two sub-clusters and considering the separation index.

In principle, we can use any clustering algorithm to obtain a 2-cluster split. However, not all clustering algorithm could produce 2-cluster split as what we expect. For example, if we use `kmeans` or `PAM/CLARA` to split the data points in Figure 4.5 into two sub-clusters, then the middle cluster may be split into two parts, each part paired with its closest group. If the separation index of the two sub-clusters is about 0 the split does not take place. Hence we will end up with only 1 cluster for this data set. However, there exist three obvious clusters.

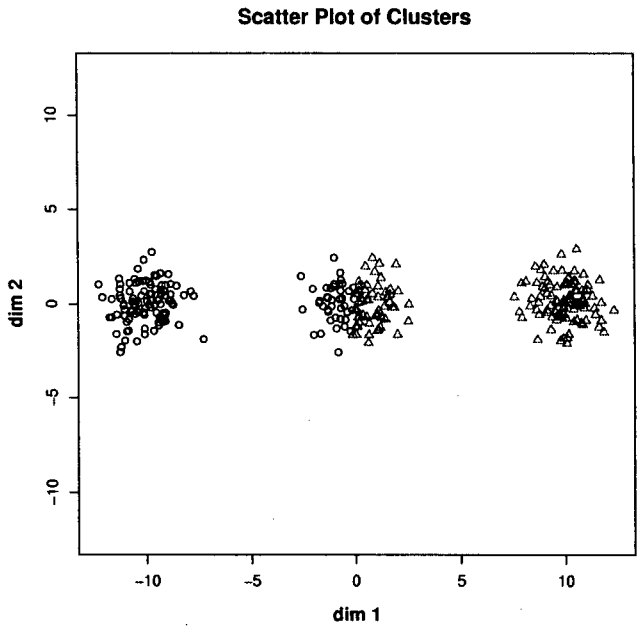


Figure 4.5: An example shows that splitting a 3-cluster data set into two sub-clusters form a split through the center of the middle cluster. The circles are for points from sub-cluster 1 and the triangles are for points from sub-cluster 2.

To handle this problem, we should choose clustering algorithms that will form a good 2-split if there are 3 or more sub-clusters. We conduct a simple simulation study and find that `Ward`, `EMclust0` and `Mclust0` work well in 2-cluster splitting compared to the `kmeans`, `MKmeans`, and `PAM/CLARA` algorithms. In this simulation study, each data set contains three clusters generated

from bivariate normal distributions $\mathrm{N}(\boldsymbol{\theta}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2, 3$, respectively, where

$$\boldsymbol{\theta}_1 = \begin{pmatrix} -10 \\ 0 \end{pmatrix}, \boldsymbol{\theta}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\theta}_3 = \begin{pmatrix} 10 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{4.7.4}$$

Each cluster has 100 data points. There are 1000 replications in total. For each data set, we produce 2-cluster splits by the `kmeans`, `MKmeans`, `PAM/CLARA`, `EMclust0`, `Ward`, and `Mclust0` algorithms respectively. Then we check if the two sub-clusters shall be merged based on the separation index. If we get only one cluster, then it indicates that the split was through the middle cluster rather than through a gap. In the simulation, we set $\alpha = 0.05$, $\alpha_0 = 0.05$, and $J_T^* = 0.15$. We record in Table 4.2 the number of times of getting only one cluster after split/merge for each algorithm. We can see that the `MKmeans`, `Ward`, `EMclust0`, and `Mclust0` algorithms can produce good 2-split

Table 4.2: Times of getting only one cluster after merging 2 sub-clusters produced by 2-split partitions. There are 1000 3-cluster data sets in total ($\alpha = 0.05$, $\alpha_0 = 0.05$, and $J_T^* = 0.15$).

	`kmeans`	`MKmeans`	`PAM/CLARA`	`Ward`	`EMclust0`	`Mclust0`
$\hat{k} = 1$	3	0	878	0	0	0
system time	0.029	0.043	0.039	0.107	0.513	0.470

partition, in this simple simulation study. The performance of `MKmeans` may still be affected by the initial cluster centers. So we prefer to use `EMclust0`, `Mclust0` or `Ward` to produce 2-split partition. Since the speeds of `EMclust0` and `Mclust0` are slower than that of `Ward`, we will use the `Ward` algorithm to do 2-cluster splitting. The `Ward` algorithm is slow when handling large data sets. We will propose a method to improve the speed of the SEQCLUST algorithm in Section 4.10.

In the splitting step, we need to consider if it is possible that a cluster which is formed by merging several clusters in the previous merging step will be split again. That is, is it possible to result in an infinite loop? We think that it is possible to split a cluster formed by previously merging process, but it will not result in an infinite loop. Suppose a cluster C is formed by merging clusters C_1 and C_2 in the previous step. If C_1 or C_2 contains subclusters, then cluster C might be split in the splitting process. The two subclusters C_1' and C_2' after splitting can not be same as C_1 and C_2. Otherwise, C_1 and C_2 would not be merged in the previous step. For the same reason, if C is split into C_1' and C_2', then C_1' and C_2' could not be merged in the next step. For the simulated and real data sets we tried, we did not find infinite loop cases.

4.8 Post-Process

Sometimes clusters with small sizes will be obtained by the splitting process. So after merging and splitting, we need to detect these clusters and regard the data points in these clusters as potential outliers. But how do we know if the sample size of a cluster is small or not? We simply check the ratio of cluster sizes to the maximum cluster size. If the ratio is less than a threshold, 0.1 say, then we regard the cluster is an outlier cluster, i.e. whose data points are regarded as outliers.

Denote $\hat{k}_0$ as the final estimated number of clusters and P_1 as the corresponding partition. We also can obtain a $\hat{k}_0$-cluster partition P_2 by using the clustering algorithm specified at the initializing stage. We can compare the minimum separation index of the two partition. If the minimum separation index of the partition P_1 is larger than that of P_2, then we use P_1 as the final partition. Otherwise, we use P_2 as the final partition.

4.9 Comparative Study

In this section, we check the performance of the SEQCLUST method by using both simulated data sets and real data sets. We also compare the SEQCLUST method with other methods which were proposed in the literature to estimate the number of clusters.

In Subsection 4.9.1, we describe the criteria to measure the performance of the SEQCLUST method. To compare with the SEQCLUST method, we select several number-of-cluster-estimation methods which are used in Tibshirani et al. (2001) and Sugar and James (2003). These methods are described in Subsection 4.9.2. The results for simulated data sets and real data sets are shown in Subsections 4.9.3 and 4.9.4 respectively.

4.9.1 Measure of Performance

For both simulated data sets and real data sets in this chapter, we know the true numbers of clusters and the true partitions. Therefore, to measure the performance of the SEQCLUST method, we can count the number of underestimated and overestimated estimates of the number of clusters. The sizes of underestimates and overestimates can also measure the performance of the SEQCLUST method. Denote k_0^* as the true number of clusters and $\hat{k}_0$ as the estimated number of clusters. Denote δ as the difference $\hat{k}_0 - k_0^*$. If $\hat{k}_0 < k_0^*$, then the size of underestimate is $-\delta$. If $\hat{k}_0 > k_0^*$,

then the size of overestimate is δ.

In addition to the underestimates and overestimates of the number of clusters, we check the agreement between the obtained partition with the true partition. We use the five external indexes studied in Milligan (1986) to measure the agreement between the obtained partitions and the true partitions.

Given two partitions U and V, let a be the number of point pairs which were grouped together by both of partitions U and V, b be the number of point pairs where the partition U placed them in the same cluster and the partition V placed them in different clusters, c be the frequency count for the opposite results and d be the count for the number of pairs where both U and V placed the points in different clusters. Then the formula of the external indexes we use are:

Hubert and Arabie's Adjusted Rand index	$(a+d-n_h)/(a+b+c+d-n_h)$
Morey and Agresti's Adjusted Rand index	$(a+d-n_m)/(a+b+c+d-n_m)$
Rand index	$(a+d)/(a+b+c+d)$
Fowlkes and Mallows index	$a/[(a+b)(a+c)]^{1/2}$
Jaccard index	$a/(a+b+c)$

where

$$n_h = \frac{1}{2(n-1)}\left[n(n^2+1) - (n+1)\sum_{i=1}^{k_U} n_{i\cdot}^2 - (n+1)\sum_{j=1}^{k_V} n_{\cdot j}^2 + \frac{2}{n}\sum_{i=1}^{k_U}\sum_{j=1}^{k_V} n_{i\cdot}^2 n_{\cdot j}^2\right],$$

$$n_m = \frac{1}{2}n(n-1) - \frac{1}{2}\sum_{i=1}^{k_U} n_{i\cdot}^2 - \frac{1}{2}\sum_{j=1}^{k_V} n_{\cdot j}^2 + \frac{1}{n^2}\sum_{i=1}^{k_U}\sum_{j=1}^{k_V} n_{i\cdot}^2 n_{\cdot j}^2,$$

k_U and k_V are the number of clusters in partition U and V respectively, n_{ij} is the number of points in cluster i as produced by partition U which also are in cluster j as produced by partition V, and $n_{i\cdot} = \sum_{j=1}^{k_V} n_{ij}$, $n_{\cdot j} = \sum_{i=1}^{k_U} n_{ij}$ and $n_{\cdot\cdot} = \sum_{i=1}^{k_U}\sum_{j=1}^{k_V} n_{ij}$.

The advantages of these external indexes are (1) the number of clusters in the two partitions can be different; (2) different labellings of the memberships of data points will not affect the index values.

The third way to check the performance of the SEQCLUST method is to compare the values of the five external indexes for the partitions obtained by the SEQCLUST method with those obtained by other clustering algorithms which are provided with specifying true numbers of

clusters. The clustering algorithms we use are `kmeans`, `MKmeans`, `PAM/CLARA`, `Ward`, `EMclust0`, and `Mclust0`.

4.9.2 Other Number-of-Cluster-Estimation Methods

As mentioned in Section 4.1, many methods have been proposed to estimate the number of clusters. In this chapter, we only consider six methods — `CH`, `Hartigan`, `KL`, `Silhouette`, `EMclust1`, and `Mclust1`.

The `CH` method has already been mentioned in Subsection 4.4.1 (see page 68). The estimated number of clusters corresponds to

$$k_0 = \arg \max_{k \geq 2} \mathrm{CH}(k).$$

The rationale is that the optimal number of clusters should correspond to the most compact cluster structure.

The `Hartigan` method (Hartigan 1975) is based on the index:

$$H(k) = (n - k - 1)\left(\frac{\boldsymbol{W}_k}{\boldsymbol{W}_{k+1}} - 1\right), \tag{4.9.5}$$

where $\boldsymbol{W}_k$ is the within-group sum of squares (see page 68). The optimal number of clusters k_0 is the minimum k such that $H(k) \leq 10$. The idea is to test if it is worth adding a $(k+1)$-th cluster to the model.

`KL` method (Krzanowski and Lai 1985) is based on the index

$$KL(k) = \left|\frac{DIFF(k)}{DIFF(k+1)}\right|,$$

where

$$DIFF(k) = (k-1)^{2/p}\boldsymbol{W}_{k-1} - k^{2/p}\boldsymbol{W}_k.$$

The `KL` method uses

$$k_0 = \arg \max_{k \geq 2} KL(k)$$

as the estimated number of clusters.

The `Silhouette` index proposed by Kaufman and Rousseeuw (1990) is defined as:

$$\bar{s} = \frac{1}{n}\sum_{i=1}^{n} s_i \tag{4.9.6}$$

where

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \tag{4.9.7}$$

and a_i is the average distance of the data point $\boldsymbol{y}_i$ to other points in the cluster A where $\boldsymbol{y}_i$ belongs to, i.e.

$$a_i = \frac{1}{(n_A - 1)} \sum_{j \in A, j \neq i} d(\boldsymbol{y}_i, \boldsymbol{y}_j),$$

and b_i is the average distance to points in the nearest neighbor cluster besides its own. Define

$d(i, C)$ = average dissimilarity of the data point $\boldsymbol{y}_i$ to all data points in Cluster C.

Then

$$b_i = \min_{C \neq A} d(i, C).$$

This index s_i can take values from -1 to 1. When the index is zero, then the data point $\boldsymbol{y}_i$ has equal distance to its cluster and its nearest neighbor cluster. If the index is positive, then the data point $\boldsymbol{y}_i$ is closer to its cluster than other clusters. If the index is negative, then the data point $\boldsymbol{y}_i$ is wrongly assigned to the current cluster. Thus, if all data points are correctly assigned, then average of s_i's should be close to 1.

The `Mclust` method (Fraley and Raftery 1998, 2002) is based on model selection theories. To compare two models M_1 and M_2 for the same data D, one can use the `Bayes factor`

$$B_{12} = \frac{f(D|M_1)}{f(D|M_2)},$$

where

$$f(D|M_k) = \int f(D|\boldsymbol{\theta}_k, M_k) f(\boldsymbol{\theta}_k|M_k) d\boldsymbol{\theta}_k,$$

is the `integrated likelihood` of model M_k, $\boldsymbol{\theta}_k$ is the parameter vector under M_k, $f(\boldsymbol{\theta}_k|M_k)$ is its prior density, and $f(D|\boldsymbol{\theta}_k, M_k)$ is the probability density of D given the value of $\boldsymbol{\theta}_k$, or the likelihood function of $\boldsymbol{\theta}_k$. $B_{12} > 1$ indicates that model M_1 is better than model M_2. For more than two-models, we choose the model whose integrated likelihood is the largest. Note that we use the Bayesian abuse of notation with f standing for more than one density.

Since it is difficult to calculate the integrated likelihood, Fraley and Raftery (1998, 2002) proposed to use the Bayesian Information Criterion (BIC)

$$2\log[f(D|M_k)] \simeq 2\log[f(D|\hat{\boldsymbol{\theta}}_k, M_k) - v_k \log(n) = BIC_k$$

to approximate the value of the integrated likelihood, where v_k is the number of parameters to be estimated in model M_k, n is the number of data points, and $\hat{\theta}_k$ is the maximum likelihood estimate for parameter $\boldsymbol{\theta}_k$.

Fraley and Raftery (1998, 2002) assumed that data points are from a mixture of multivariate normal distributions

$$f(\boldsymbol{x}) = \sum_{k=1}^{k_0} \pi_k \phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{4.9.8}$$

where $\phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the density of the multivariate normal distribution with mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. Each component of the mixture corresponds to a cluster. The Expectation-Maximization (EM) algorithm is used to obtain the maximum likelihood estimates of the parameters in the model (4.9.8).

To obtain a partition, a hierarchical cluster structure is first obtained by successively merging pairs of clusters corresponding to the greatest increase in the classification likelihood

$$\prod_{i=1}^{n} \phi_{\ell_i}(\boldsymbol{y}_i|\boldsymbol{\mu}_{\ell_i}, \boldsymbol{\Sigma}_{\ell_i})$$

among all possible pairs where the ℓ_i are labels indicating a unique classification of each observation. $\ell_i = k$ if $\boldsymbol{y}_i$ belongs to the k-th component.

The BIC approximation for the integrated likelihood in the case where data are from finite mixture distributions may not be appropriate since finite mixture distributions do not satisfy the necessary regularity conditions. However, as Fraley and Raftery (2002) argued, it was demonstrated by several results that the BIC approximation is appropriate and has good performance in the model-based clustering context.

4.9.3 Comparison on Simulated Data Sets

To compare the performances of the methods (`SEQCLUST`, `CH`, `Silhouette`, `KL`, `Hartigan`, `EMclust1`, and `Mclust1`), we use the 243 simulated data sets generated by the design proposed in Section 3. The cluster structures of the 243 data sets can be classified into three groups — close, separated,

and well-separated — each having 81 data sets. The total numbers of under- and over-estimates of the numbers of clusters are given in Table 4.3 and the average values of the 5 external indexes and their standard errors are shown in Table 4.4.

The `SEQCLUST`, `CH`, `KL`, and `Hartigan` methods use `MKmeans` clustering algorithm to obtain partitions. The `Silhouette` method uses the `PAM/CLARA` clustering algorithm to obtain partitions.

The input parameters for SEQCLUST are $\alpha = 0.02, 0.03, \ldots, 0.08$, $\alpha_0 = 0.05$, and $J_T^* = 0.15$. The lower and upper bounds of the number of clusters for `EMclust1` and `Mclust` are 1 and 20 respectively. The lower and upper bounds of the number of clusters for `CH` and `Silhouette` are 2 and 20 respectively. The upper bound of the number of clusters for `KL` is 21. Noisy variables and outliers are excluded in the analysis. Original scales of the variables are used.

Table 4.3: The total numbers and sizes of underestimates and overestimates for the 243 data sets. m_- and s_- are total the number and size of underestimates while m_+ and s_+ are the total number and size of overestimates. For SEQCLUST, $\alpha = 0.02, 0.03, \ldots, 0.08$, $\alpha_0 = 0.05$, and $J_T^* = 0.15$.

close cluster structure (81 data sets)		
Method	m_- (s_-)	m_+ (s_+)
SEQCLUST (MKmeans)	9 (30)	0 (0)
EMclust1	2 (2)	16 (27)
Mclust1	1 (1)	2 (4)
CH	41 (184)	0 (0)
Hartigan	0 (0)	81 (2078)
KL	3 (17)	34 (297)
Silhouette	32 (81)	25 (110)
separated cluster structure (81 data sets)		
SEQCLUST (MKmeans)	0 (0)	0 (0)
EMclust1	0 (0)	17 (58)
Mclust1	0 (0)	4 (7)
CH	3 (6)	0 (0)
Hartigan	0 (0)	81 (1174)
KL	1 (4)	21 (135)
Silhouette	8 (9)	8 (14)
well-separated cluster structure (81 data sets)		
SEQCLUST (MKmeans)	0 (0)	0 (0)
EMclust1	0 (0)	13 (13)
Mclust1	0 (0)	1 (1)
CH	1 (1)	0 (0)
Hartigan	0 (0)	81 (1043)
KL	0 (0)	13 (95)
Silhouette	2 (3)	0 (0)

Table 4.3 shows that for simulated data sets with separated and well-separated cluster structures, only the SEQCLUST method correctly estimates the number of clusters in all cases. The CH method slightly underestimated the number of clusters, while the Mclust method slightly overestimated the number of clusters. It seems that the KL and EMclust methods tend to overestimate the number of clusters, while the Silhouette method is equal likely to slightly under- and over-estimate the number of clusters.

For simulated data sets with close cluster structures, the SEQCLUST method tends to underestimate the number of clusters. Since clusters are close to each other, we expect underestimation instead of overestimation. It is not surprising that the Mclust1 method has good performance since the data sets are generated from mixtures of multivariate normal distributions. Now, the CH method tends to underestimate much more while the KL, and EMclust1 methods tend to overestimate the number of clusters. The Silhouette method is equal likely to under- and over-estimate the number of clusters and the performance is much worse than the separated case.

The Hartigan method overestimates the number of clusters for all 243 data sets and the degree of overestimation is quite large.

For Silhouette method, we did the same analysis by using MKmeans clustering method to obtain partitions. The Silhouette method performs better in this case. This indicates that the performances of these number-of-cluster-estimation methods depend on the clustering methods.

Table 4.4 shows that the recovery rates of the SEQCLUST method are relatively high even for the data sets with close cluster structures. We can see that for the simulated data sets with close cluster structures, even given the true number of clusters , the recovery rates of the partitions obtained by EMclust0, Mclust0, kmeans, MKmeans, PAM/CLARA and Ward are not high. This implies that these data sets are challenging. For separated and well-separated cluster structures, all methods except the Hartigan method can produce good results even if the number of clusters is overestimated. It is a little bit surprising that the recovery rates of the EMclust1, KL, and Silhouette methods are high although their sizes of the overestimates are large. This indicates that these methods capture the main structures of data sets and that the overestimates may be due to splitting large clusters into smaller subclusters. We also can observe that the performances of the different clustering methods (EMclust0, Mclust0, kmeans, MKmeans, PAM/CLARA and Ward) are not the same.

We conclude from the analysis on simulated data sets that the `Hartigan` and `KL` methods are not recommended for determining the number of clusters. For data sets with close structures, the `CH` and `Silhouette` methods are also not good to determine the number of clusters. Overall, the SEQCLUST methods has good performance on determining the number of clusters for data sets with close, separated, and well-separated cluster structures. Here the conclusions are rather obvious, otherwise we would have needed a statistical comparison to check if the performances of the different methods are statistically significant.

4.9.4 Comparison on Some Real Data Sets

Data Sets

The data sets we consider in this section are generated from the test data set in Alimoglu and Alpaydin's (1996) study on pen-based recognition of handwritten digits.

In their study, 44 writers are asked to write 250 digits (0, 1, ..., 8, and 9) in random order inside boxes of 500 by 500 tablet pixel resolution. The x and y tablet coordinates and pressure level values of the pen at fixed time intervals of 100 milliseconds are recorded. The x and y coordinates are normalized so that data are invariant to translations and scale distortions and the new coordinates are within the range $[0, 100]$. Each digit sample is represented by 8 pairs of (x, y) coordinates, which are regularly spaced in arc length. That is, each digit sample can be regarded as a point in a 16-dimensional space. The pressure level values are ignored.

The 8 pairs of (x, y) coordinates implicitly contain the information on the orders of strokes that the writer wrote the digits. In fact, it is this temporal signal of pen movements that makes optical recognition different from a static spatial pattern.

The digits samples written by 14 writers consist of the writer-independent testing set. This testing data set is available at UCI Machine Learning Repository (Blake et al. 1998), which is represented by a 3498×16 matrix [5].

We generate three data sets from the testing set to cluster digits samples [6]. The first data set `DAT1` consists of 1065 samples from digits 2, 4, and 6. These three digits are quite different. So we expect clustering algorithms should perform well for the data set `DAT1`. The second data set

[5] There should be $14 \times 250 = 3500$ samples in this testing set. However, there are actually 3498 samples at UCI Machine Learning Repository.

[6] These three subsets were initially created by my supervisor as blinded data sets.

Table 4.4: Average values of the 5 external indexes and their standard errors (For SEQCLUST, α = 0.02, 0.03, ..., 0.08, $\alpha_0 = 0.05$, and $J_T^* = 0.15$)

Method	HA	MA	Rand	FM	Jaccard
close cluster structure (81 data sets)					
SEQCLUST (MKmeans)	0.742 (0.207)	0.743 (0.207)	0.892 (0.165)	0.812 (0.113)	0.689 (0.158)
EMclust1	0.841 (0.069)	0.841 (0.068)	0.953 (0.020)	0.872 (0.063)	0.877 (0.096)
Mclust1	0.853 (0.062)	0.853 (0.062)	0.956 (0.021)	0.882 (0.055)	0.793 (0.086)
CH	0.554 (0.304)	0.554 (0.304)	0.790 (0.174)	0.690 (0.206)	0.538 (0.260)
Hartigan	0.227 (0.078)	0.230 (0.077)	0.814 (0.090)	0.375 (0.057)	0.163 (0.051)
KL	0.635 (0.233)	0.635 (0.232)	0.892 (0.095)	0.712 (0.179)	0.561 (0.233)
Silhouette	0.635 (0.207)	0.636 (0.207)	0.870 (0.118)	0.727 (0.146)	0.576 (0.195)
EMclust0	0.839 (0.092)	0.835 (0.092)	0.951 (0.028)	0.867 (0.080)	0.773 (0.114)
Mclust0	0.859 (0.051)	0.859 (0.051)	0.958 (0.015)	0.886 (0.049)	0.799 (0.078)
kmeans	0.803 (0.065)	0.803 (0.064)	0.941 (0.018)	0.842 (0.064)	0.731 (0.091)
MKmeans	0.811 (0.047)	0.811 (0.047)	0.943 (0.016)	0.849 (0.050)	0.740 (0.075)
PAM/CLARA	0.536 (0.233)	0.537 (0.232)	0.855 (0.084)	0.632 (0.194)	0.490 (0.208)
Ward	0.765 (0.073)	0.766 (0.073)	0.930 (0.019)	0.811 (0.074)	0.689 (0.103)
separated cluster structure (81 data sets)					
SEQCLUST (MKmeans)	0.984 (0.007)	0.984 (0.007)	0.995 (0.002)	0.987 (0.006)	0.975 (0.012)
EMclust1	0.968 (0.055)	0.968 (0.055)	0.992 (0.012)	0.973 (0.046)	0.950 (0.084)
Mclust1	0.985 (0.023)	0.985 (0.023)	0.996 (0.006)	0.988 (0.019)	0.977 (0.035)
CH	0.968 (0.095)	0.968 (0.095)	0.986 (0.054)	0.978 (0.055)	0.960 (0.087)
Hartigan	0.413 (0.169)	0.415 (0.169)	0.843 (0.098)	0.547 (0.123)	0.317 (0.136)
KL	0.876 (0.201)	0.876 (0.200)	0.961 (0.077)	0.907 (0.145)	0.845 (0.230)
Silhouette	0.963 (0.052)	0.963 (0.051)	0.989 (0.017)	0.971 (0.040)	0.945 (0.073)
EMclust0	0.951 (0.108)	0.951 (0.107)	0.986 (0.030)	0.960 (0.086)	0.933 (0.138)
Mclust0	0.990 (0.007)	0.990 (0.007)	0.997 (0.002)	0.992 (0.005)	0.984 (0.011)
kmeans	0.943 (0.098)	0.944 (0.098)	0.982 (0.037)	0.955 (0.073)	0.922 (0.119)
MKmeans	0.984 (0.008)	0.984 (0.008)	0.995 (0.003)	0.987 (0.007)	0.974 (0.013)
PAM/CLARA	0.901 (0.114)	0.902 (0.114)	0.972 (0.032)	0.920 (0.096)	0.864 (0.147)
Ward	0.979 (0.012)	0.979 (0.012)	0.994 (0.003)	0.983 (0.011)	0.967 (0.020)
well-separated cluster structure (81 data sets)					
SEQCLUST (MKmeans)	0.999 (0.001)	0.999 (0.001)	1.000 (0.000)	0.999 (0.001)	0.998 (0.002)
EMclust1	0.986 (0.041)	0.986 (0.041)	0.996 (0.013)	0.989 (0.032)	0.979 (0.059)
Mclust1	0.998 (0.012)	0.998 (0.012)	0.999 (0.003)	0.999 (0.009)	0.998 (0.018)
CH	0.998 (0.006)	0.998 (0.006)	1.000 (0.001)	0.999 (0.005)	0.997 (0.010)
Hartigan	0.457 (0.209)	0.458 (0.208)	0.851 (0.100)	0.583 (0.151)	0.363 (0.185)
KL	0.935 (0.173)	0.935 (0.173)	0.981 (0.056)	0.951 (0.129)	0.920 (0.202)
Silhouette	0.996 (0.020)	0.996 (0.020)	0.999 (0.005)	0.997 (0.016)	0.994 (0.031)
EMclust0	0.974 (0.065)	0.974 (0.065)	0.993 (0.017)	0.978 (0.054)	0.961 (0.093)
Mclust0	0.964 (0.084)	0.965 (0.084)	0.991 (0.022)	0.971 (0.069)	0.950 (0.117)
kmeans	0.924 (0.120)	0.924 (0.120)	0.977 (0.047)	0.940 (0.087)	0.897 (0.145)
MKmeans	0.999 (0.001)	0.999 (0.001)	1.000 (0.000)	0.999 (0.001)	0.998 (0.002)
PAM/CLARA	0.992 (0.012)	0.992 (0.012)	0.998 (0.003)	0.993 (0.011)	0.987 (0.021)
Ward	0.999 (0.001)	0.999 (0.001)	1.000 (0.000)	0.999 (0.001)	0.999 (0.002)

DAT2 consists of 500 samples from digits 4, 5, and 6. Since the digits 5 and 6 are similar to each other, we expect it would be more difficult to detect the patterns in DAT2 than in DAT1. The third data set DAT3 consists of 2436 samples from digits 1, 3, 4, 6, 8, 9 and 0. We expect it would be the most difficult to detect the natural patterns among the three data sets DAT1, DAT2, and DAT3.

The three data sets are quite challenging in that the data classes are far from elliptical in shape (see the pairwise scatter plots Figure 4.6).

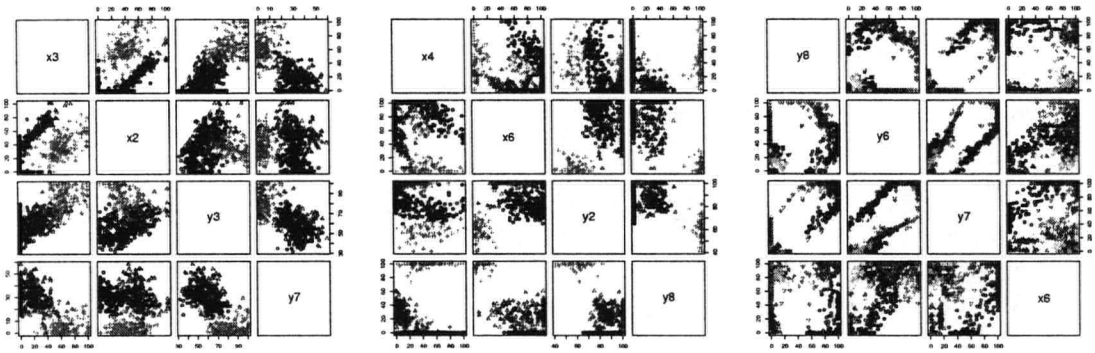


Figure 4.6: Pairwise scatter plots of DAT1, DAT2, and DAT3 show that the clusters are far from elliptical in shape. For each data set, there are 16 variables. Only 4 variables are randomly chosen to draw the scatter plots.

Initial Analysis

Since we know the true membership of the samples, we can calculate the separation index matrix and create the two dimensional projection by the technique described in Appendix A. We set $\alpha = 0.05$ when we calculate the separation index.

The normal and quantile versions of the separation index matrix with $\alpha = 0.05$ for DAT1 are shown in Table 4.5. The minimum separation index value which is close to 0.5 indicates that the three classes are well-separated. The two-dimensional projection for DAT1 is shown in Figure 4.7. We can see that the samples of digits 2, 4, and 6 are separated.

The normal and quantile versions of the separation index matrix with $\alpha = 0.05$ for DAT2 are shown in Table 4.6. the following two tables. The separation index values imply that the three classes of digit samples are well-separated. The two-dimensional projection for DAT2 is shown in Figure 4.8. We can see that the samples of digits 4, 5, and 6 are well-separated. However, there exists obvious two sub-classes in the samples of digit 5. The two sub-classes correspond to two

Table 4.5: The separation index matrix for **DAT1** ($\alpha = 0.05$)

Normal version

	"4"	"6"	"2"
"4"	-1.000	0.459	0.492
"6"	0.459	-1.000	0.618
"2"	0.492	0.618	-1.000

Quantile version

	"4"	"6"	"2"
"4"	1.000	0.472	0.479
"6"	0.472	1.000	0.633
"2"	0.479	0.633	1.000

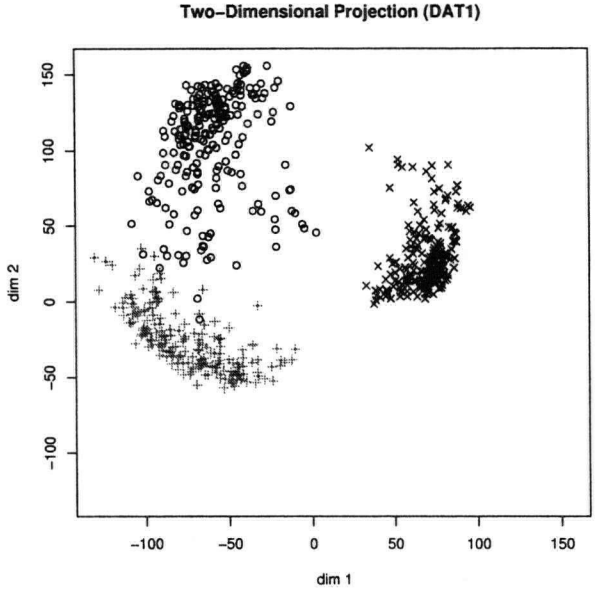


Figure 4.7: 2-d projection for samples of digits 2, 4, and 6. Circles are samples for digit 2. The symbol "+" are samples for digit 4. The symbol "×" are samples for digit 6.

temporal signals of pen movements when writers wrote the digits 5. Some writers first wrote the horizontal stroke "-", while other writers wrote "-" last. Figure 4.9 shows examples of these two different writing order. The numbers and arrows in the figure indicate respectively the 8 pairs of coordinates and temporal pen-movements when the digit was written on the tablet.

The normal and quantile versions of the separation index matrix with $\alpha = 0.05$ for **DAT3** are shown in Tables 4.7 and 4.8 respectively. The minimum separation indexes indicate that the 9 classes of digit samples are separated. The two-dimensional projection for **DAT3** is shown in Figure 4.10. We can see that samples of digits 8 and 0 have the smallest separation index value (normal version 0.156; quantile version 0.140). Also we can see from the Figure 4.10 that there are two subclasses for digit 1. The two sub-classes correspond to two different ways to write the digit

Table 4.6: The separation index matrix for DAT2 ($\alpha = 0.05$)

Normal version

	"4"	"6"	"5"
"4"	-1.000	0.457	0.298
"6"	0.457	-1.000	0.336
"5"	0.298	0.336	-1.000

Quantile version

	"4"	"6"	"5"
"4"	1.000	0.469	0.307
"6"	0.469	1.000	0.343
"5"	0.307	0.343	1.000

Table 4.7: The normal version separation index matrix for DAT2 ($\alpha = 0.05$)

	"8"	"1"	"4"	"9"	"3"	"0"	"6"
"8"	-1.000	0.574	0.635	0.664	0.648	0.156	0.411
"1"	0.574	-1.000	0.358	0.244	0.382	0.455	0.350
"4"	0.635	0.358	-1.000	0.330	0.565	0.473	0.446
"9"	0.664	0.244	0.330	-1.000	0.224	0.610	0.552
"3"	0.648	0.382	0.565	0.224	-1.000	0.674	0.538
"0"	0.156	0.455	0.473	0.610	0.674	-1.000	0.394
"6"	0.411	0.350	0.446	0.552	0.538	0.394	-1.000

Table 4.8: The quantile version separation index matrix for DAT2 ($\alpha = 0.05$)

	"8"	"1"	"4"	"9"	"3"	"0"	"6"
"8"	-1.000	0.569	0.635	0.646	0.654	0.140	0.435
"1"	0.568	-1.000	0.391	0.189	0.382	0.412	0.314
"4"	0.635	0.391	-1.000	0.301	0.584	0.442	0.456
"9"	0.646	0.190	0.301	-1.000	0.229	0.614	0.513
"3"	0.654	0.382	0.584	0.229	-1.000	0.677	0.514
"0"	0.140	0.413	0.442	0.614	0.677	-1.000	0.362
"6"	0.435	0.314	0.456	0.513	0.514	0.362	-1.000

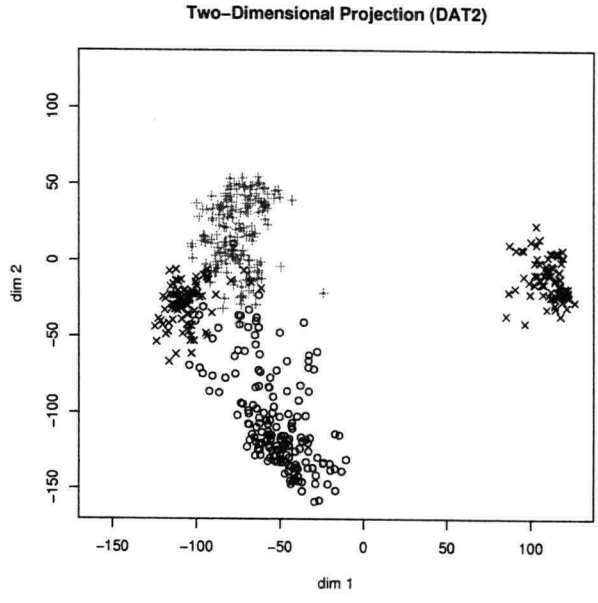


Figure 4.8: 2-d projection for samples of digits 4, 5, and 6. Circles are samples for digit 4. The symbol "+" are samples for digit 6. The symbol "×" are samples for digit 5.

1. Some writers added an additional "-" at bottom of 1, while other writers did not. Figure 4.11 shows these two different writing orders.

The initial analysis in this subsection indicates that samples of digits are separated. However, there might exist sub-classes for some digits.

Comparison Results

Like for simulated data sets, we compare the SEQCLUST method with other methods, such as `CH`, `Silhouette`, `Hartigan`, `KL`, `EMclust1`, and `Mclust1`, which are used to estimate the number of clusters. The `kmeans`, `MKmeans`, `PAM/CLARA`, `Ward`, `EMclust0`, and `Mclust0` methods which are provided with specifying true number of clusters can be used as references. For the SEQCLUST method, different clustering algorithms are used. The input parameters are $\alpha = 0.005, 0.010, 0.015, 0.020, 0.025, 0.030$, $\alpha_0 = 0.05$, and $J_T^* = 0.15$. The α values are smaller than we used for simulated data sets because each variable in pen digit data sets is bounded by $[0, 100]$ with "mass" at the end points and hence projections tend to be short-tailed relative to the normal distribution. That is, there are no "outlying" data points are in the pen digit data sets. Original scales of the variables are used.

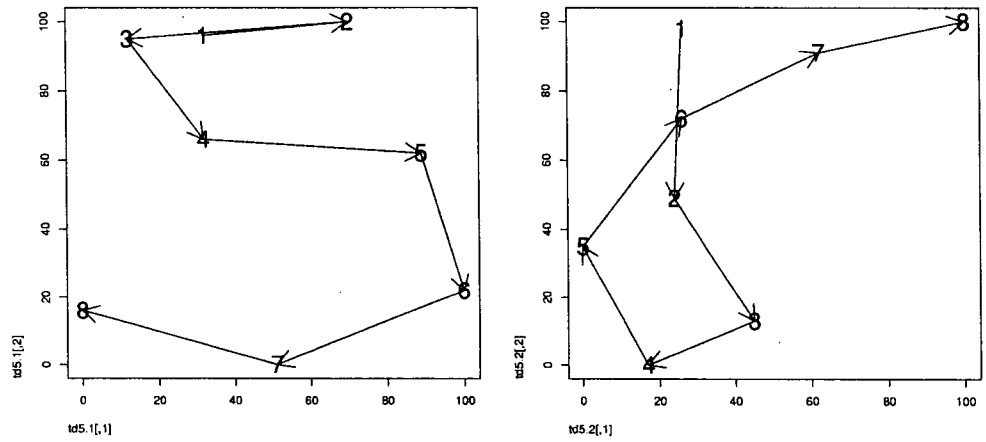


Figure 4.9: Sample "reconstructed" handwritten digit 5. The left-panel shows an example that "-" is the first stroke when writing the digit 5. The right-panel shows an example that "-" is the last stroke.

The estimates of the number of clusters and values of the five external indexes for **DAT1** are shown in Table 4.9. The estimated numbers of clusters obtained by the SEQCLUST method are all equal to the true number of clusters. The recovery rates of the partitions obtained by the SEQCLUST method are high. The interval estimates of the number of clusters obtained by the SEQCLUST method are shown in Table 4.11. We can see that the performance of SEQCLUST method is pretty good and stable for **DAT1**. **CH** and **KL** also have good performance for **DAT1**. The poor performances of **EMclust1** and **Mclust1** suggest that the shape of clusters are far from normal. Given the true number of clusters, all clustering methods (especially **Ward** and **Mclust0**) have good performance probably because the 3 clusters are well-separated. The averages of sample digits in the partition obtained by SEQCLUST with **Ward** are shown in Figure 4.12. The normal and quantile versions of the separation index matrix for the partition obtained by the SEQCLUST method with **Ward** are given in Table 4.10 ($\alpha = 0.01$). [7] These two matrices indicate that the 3 clusters are well-separated. The results for **DAT2** are shown in Tables 4.12 and 4.13. Although the true number of classes are used, the recovery rates of the six clustering algorithms (**kmeans**, **MKmeans**, **PAM/CLARA**, **Ward**, **EMclust0** and **Mclust0**) are low. This indicates that the three clusters are difficult to detect in the 16-dimensional space. Again, the SEQCLUST method produces partitions with

[7] The α value is output with the final partition by the SEQCLUST method with **Ward**.

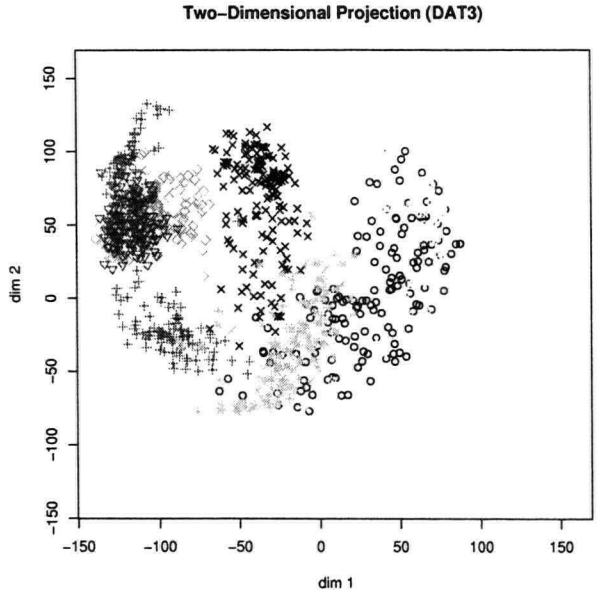


Figure 4.10: 2-d projection for samples of digits 8, 1, 4, 9, 3, 0, and 6. The symbols "$\circ$", "$+$", "$\times$", "$\diamond$", "$\triangledown$", "$\boxtimes$", and "$*$", represent samples of digits 8, 1, 4, 9, 3, 3, 0, 6 respectively.

better recovery rates than those partitions obtained by other methods. Especially the SEQCLUST method has better performance than the clustering methods provided with the true numbers of clusters (`kmeans`, `MKmeans`, `PAM/CLARA`, `Ward`, `EMclust0` and `Mclust0`. The 4-cluster partitions obtained by the SEQCLUST method captures the two subclasses structure of the digit 5. For example, for the partition obtained by the SEQCLUST method with `Mclust` clustering method, all samples for digit 6 have been correctly grouped together. All samples except samples 398 and 482 for digit 4 have been correctly grouped together. All subsamples for digit 5 which "-" is written in last stroke are correctly group together. All subsamples for digit 5 which "-" is the first stroke are also correctly group together. However sample 398 and 482 are mistakenly assigned to the the subsample where "-" is the first stroke. Thus, the SEQCLUST with `Mclust` has only two misclassifications. Figure 4.13 shows the averages of samples from digit 4, 5, and 6 in the partition obtained by SEQCLUST with `Mclust`. The samples 398 and 482 are shown in Figure 4.14 The 5-cluster partition obtained by SEQCLUST with `Ward` finds another subcluster of the digit 5 samples. Figure 4.15 shows the corresponding averages of the samples. All samples are correctly classified except that there are four digit 4 samples are misclassified into two of the three subclusters of the digit 5 samples. The samples 361, 398, and 482 are misclassified into the subclusters of digit

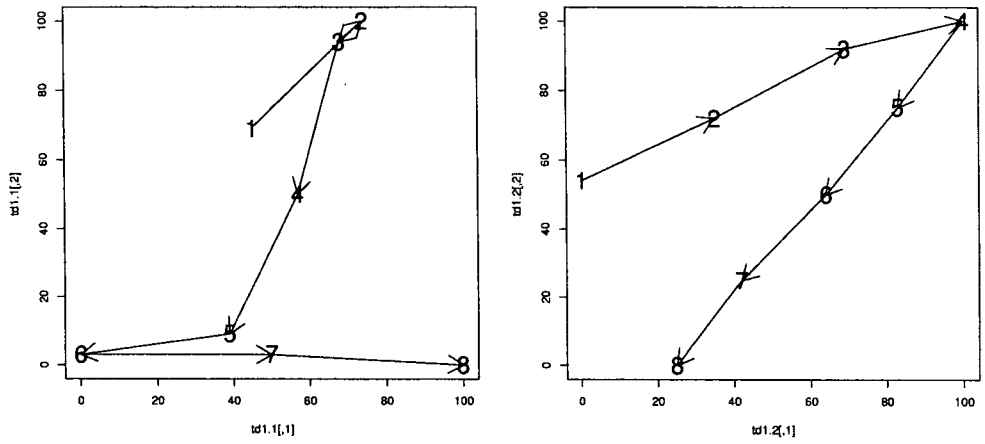


Figure 4.11: Sample "reconstructed" handwritten digit 1. The left-panel shows an example that an additional "-" is added at the bottom of the digit 1. The right-panel shows an example that no additional "-" is added at the bottom of the digit 1.

5 samples shown in the bottom-middle panel of Figure 4.15. The sample 59 is misclassified into the subclusters of digit 5 samples shown in the top-right panel of Figure 4.15. These four samples are shown in Figure 4.16. The CH method still has good performance in estimating the number of clusters, while the KL method has poor performance for DAT2.

The normal and quantile versions of the separation index matrix for the partition of DAT2 obtained by the SEQCLUST method with Ward are given in Table 4.14 ($\alpha = 0.015$). [8] These two matrices indicate that the 5 clusters obtained are well-separated.

The results for DAT3 are shown in Tables 4.15 and 4.16. The averages of the 10 subclusters obtained by using SEQCLUST with Ward are shown in Figure 4.17. We can see that there are two different ways to write the digit 8 and three different ways to write the digit 1. The SEQCLUST method can produce better results in terms of the five external indexes. The estimated number of clusters is reasonable by Figure 4.17. The estimated number of clusters and the recovery rates of other methods are not good.

The normal and quantile versions of the separation index matrix for the partition of DAT3 obtained by the SEQCLUST method with Ward ($\alpha = 0.015$) are given in Tables 4.17 and 4.18 respectively. [9] These two matrices indicate that the 10 clusters obtained are well-separated.

[8] The α value is output with the final partition by the SEQCLUST method with Ward.
[9] The α value is output with the final partition by the SEQCLUST method with Ward.

Table 4.9: Results for unscaled DAT1 ($k_0 = 3$, $n = 600$, $p = 16$), which contains the digits 2, 4, and 6.

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard	system time
SEQCLUST (kmeans)	3	0.951	0.951	0.978	0.967	0.937	7.41
SEQCLUST (MKmeans)	3	0.951	0.951	0.978	0.967	0.937	9.04
SEQCLUST (PAM/CLARA)	3	0.951	0.951	0.978	0.968	0.937	9.43
SEQCLUST (Ward)	3	1.000	1.000	1.000	1.000	1.000	80.53
SEQCLUST (EMclust)	3	1.000	1.000	1.000	1.000	1.000	108.10
SEQCLUST (Mclust)	3	1.000	1.000	1.000	1.000	1.000	94.24
EMclust1	6	0.766	0.767	0.903	0.843	0.711	181.60
Mclust1	24	0.190	0.195	0.716	0.387	0.150	168.91
CH	3	0.899	0.900	0.955	0.933	0.874	2.02
Hartigan	22	0.195	0.200	0.718	0.392	0.154	1.62
KL	3	0.899	0.900	0.955	0.933	0.874	1.34
Silhouette	2	0.572	0.573	0.779	0.774	0.600	0.93
kmeans		0.899	0.900	0.955	0.933	0.874	0.01
MKmeans		0.899	0.900	0.955	0.933	0.874	0.04
PAM/CLARA		0.908	0.908	0.959	0.938	0.884	0.01
Ward		1.000	1.000	1.000	1.000	1.000	0.30
EMclust0		0.875	0.876	0.945	0.917	0.847	1.96
Mclust0		1.000	1.000	1.000	1.000	1.000	2.12

These three pen digits data sets are quite challenging in that clusters probably are far from convex in shape. Moreover, the fact that there are different ways to write a digit causes difficulty to check if the estimated numbers of clusters and the obtained partitions are good or not.

The performances of the `Hartigan`, `KL`, and `Silhouette` methods for these three pen digits data sets are not good. Relatively, the performance of the `CH` method is better and the SEQCLUST methods are among the best.

Overall, the SEQCLUST method is quite robust when we use it to analyze the pen digits

Table 4.10: The separation index matrix for `DAT1` ($\alpha = 0.01$). The partition is obtained by the SEQCLUST method with `Ward`.

Normal version

	1	2	3
1	-1.000	0.618	0.492
2	0.618	-1.000	0.459
3	0.492	0.459	-1.000

Quantile version

	1	2	3
1	-1.000	0.633	0.479
2	0.633	-1.000	0.472
3	0.479	0.472	-1.000

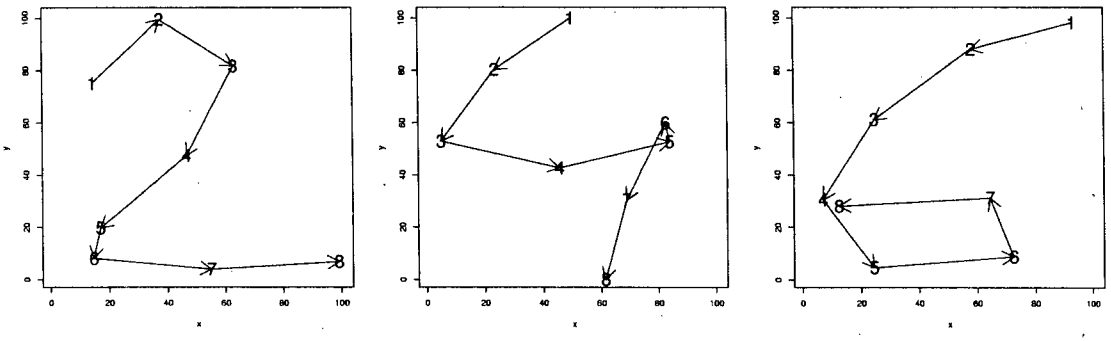


Figure 4.12: Left: Average of the digit 2 samples. Middle: Average of the digit 4 samples. Right: Average of the digit 6 samples.

Table 4.11: Interval estimates of the number of clusters obtained by the SEQCLUST method for DAT1

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[3, 3]	SEQCLUST (Ward)	[3, 4]
SEQCLUST (MKmeans)	[3, 3]	SEQCLUST (EMclust)	[3, 4]
SEQCLUST (PAM/CLARA)	[3, 4]	SEQCLUST (Mclust)	[3, 3]

data sets and shows good performance in estimating the number of clusters and getting partitions. The SEQCLUST method detects that there are probably 3 different ways to write the digits 1 and 5 and two different ways to write the digit 8.

The clustering methods seem to produce well-separated clusters for `DAT1`, `DAT2` and `DAT3`. The reason that different methods get quite different partitions is probably because of the sparseness in high dimensional space.

4.10 Discussion

In this chapter, we propose a sequential clustering algorithm `SEQCLUST` which simultaneously estimates the number of clusters and does clustering. The idea is to obtain a partition of a data set by merging and splitting processes. The merging and splitting criteria are directly based on the degree of separation among clusters. We apply the `SEQCLUST` method to both simulated data sets and real data sets. The results show good performance of the `SEQCLUST` method for data sets with separated or well-separated cluster structures which are elliptical in shape. For data sets with close cluster structures, the `SEQCLUST` method sometimes underestimates the number of clusters, which

Table 4.12: Results for unscaled DAT2 ($k_0 = 3$, $n = 500$, $p = 16$), which contains samples for digits 4, 5, and 6.

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard	system time
SEQCLUST (kmeans)	4	0.861	0.861	0.941	0.906	0.823	11.75
SEQCLUST (MKmeans)	5	0.812	0.813	0.921	0.873	0.766	15.53
SEQCLUST (PAM/CLARA)	5	0.791	0.792	0.912	0.859	0.742	11.06
SEQCLUST (Ward)	5	0.833	0.834	0.930	0.888	0.789	62.38
SEQCLUST (EMclust)	4	0.821	0.822	0.923	0.879	0.781	90.05
SEQCLUST (Mclust)	4	0.872	0.872	0.945	0.914	0.836	99.60
EMclust1	6	0.680	0.682	0.870	0.781	0.619	127.64
Mclust1	25	0.173	0.179	0.712	0.367	0.135	108.80
CH	4	0.763	0.764	0.899	0.838	0.716	1.73
Hartigan	17	0.233	0.238	0.729	0.429	0.186	0.96
KL	12	0.318	0.322	0.753	0.506	0.261	1.22
Silhouette	2	0.125	0.126	0.492	0.569	0.357	0.89
kmeans		0.531	0.533	0.781	0.706	0.542	0.00
MKmeans		0.531	0.533	0.781	0.706	0.542	0.02
PAM/CLARA		0.531	0.532	0.782	0.702	0.538	0.01
Ward		0.625	0.626	0.826	0.763	0.613	0.22
EMclust0		0.624	0.625	0.825	0.763	0.613	1.84
Mclust0		0.606	0.608	0.817	0.751	0.598	1.61

Table 4.13: Interval estimates of the number of clusters obtained by the SEQCLUST method for DAT2

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[4, 5]	SEQCLUST (Ward)	[5, 5]
SEQCLUST (MKmeans)	[5, 5]	SEQCLUST (EMclust)	[4, 4]
SEQCLUST (PAM/CLARA)	[4, 5]	SEQCLUST (Mclust)	[4, 4]

is what we might expect.

The `SEQCLUST` method proposed in Section 4.3 is quite generic. Different implementations can be substituted. For example, we can replace the separation index with other indexes; we can merge a pair of clusters at a time instead of merging all mergable clusters at once; we can use different initialization method to obtain a suitable initial value of the number of clusters, etc..

Like other clustering algorithms, the SEQCLUST method can not be expected to work well for all types of data sets. The SEQCLUST algorithm is designed to deal with continuous-convex-type data. That is, each variable should be continuous. SEQCLUST should do better when the shapes of clusters are approximately elliptical. The densities of data points decreases gradually from cluster centers to cluster edges. This is because that the `SEQCLUST` method depends on the

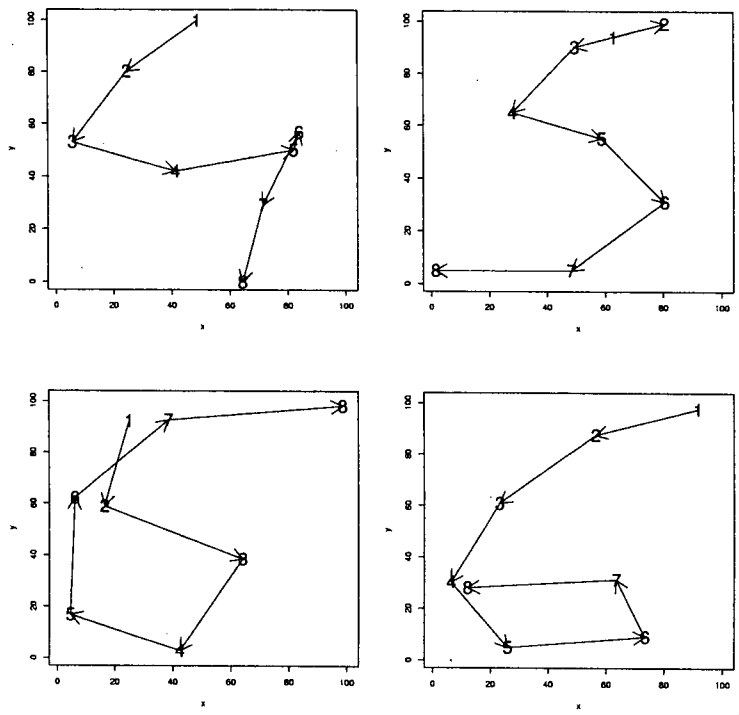


Figure 4.13: Top-left: Average of the digit 4 samples. Top-right: Average of the subcluster 1 of the digit 5 samples. Bottom-left: Average of the subcluster 2 of the digit 5 samples. Bottom-right: Average of the digit 6 samples.

separation index which for faster computation depends on the mean vectors and covariance matrices of clusters. Therefore, clusters are required to be convex-shaped so that the covariance matrix is a good measure of the shape, size, and orientation of a cluster.

Most clustering algorithms assume explicitly or implicitly that they are designed to deal with continuous-convex-type data. It is challenging, but important, to develop clustering algorithms to handle other data types, and we will try to extend the SEQCLUST algorithm in our future research.

It is quite common in practice that the data set contains missing values. Currently, the `SEQCLUST` method could not handle missing values. Missing values can be produced by many mechanisms. So we could not simply delete the data points containing missing values. We need subject knowledge to decide which methods should be used to impute (fill in) the missing values. After imputing the missing values, we can apply the SEQCLUST method. It would be another future research area.

Outliers are also commonly encountered in real data sets. The mean vectors and covariance

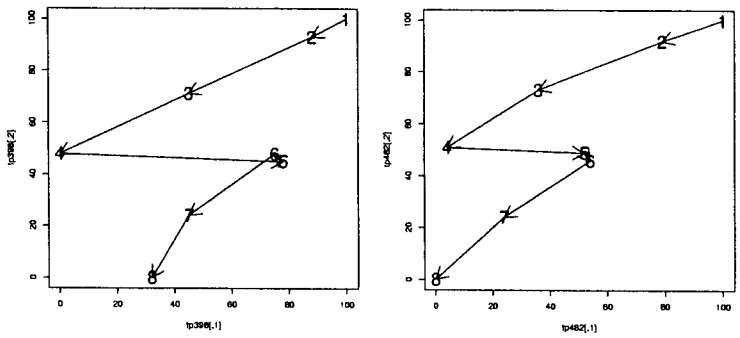


Figure 4.14: Top-left: Sample 398. Top-right: Sample 482.

Table 4.14: The separation index matrix for DAT2 ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with **Ward**.

Normal version

	1	2	3	4	5
1	-1.000	0.488	0.697	0.450	0.462
2	0.488	-1.000	0.815	0.434	0.524
3	0.697	0.815	-1.000	0.818	0.850
4	0.450	0.434	0.818	-1.000	0.322
5	0.462	0.524	0.850	0.322	-1.000

Quantile version

	1	2	3	4	5
1	-1.000	0.467	0.713	0.447	0.470
2	0.467	-1.000	0.795	0.463	0.552
3	0.713	0.795	-1.000	0.815	0.860
4	0.447	0.463	0.815	-1.000	0.297
5	0.470	0.552	0.860	0.297	-1.000

matrices are sensitive to outliers. We will study ways to robustify the **SEQCLUST** method in our future research.

If there is no cluster structure shown in the given dimensions, then either there are really no cluster in the data set, or some variables are non-informative variables and the cluster structure is masked by these variables, or the cluster structure is in a higher dimensional space. For the last case, we might detect the structure in a enlarged space by using a technique similar to that used in support vector machines. A support vector machine is a supervised learning technique to obtain useful structure information from the training set. In cluster analysis, we do not have class information. So we need study how to find a suitable enlarged space. This is a part of our future research. We will propose variable selection/weighting method to downweight the effects of noisy variables.

If cluster sample sizes are large, then the 2-split based on the Ward method will be too slow. To handle this problem, we can sample. We first take 5 (say) samples from the cluster and

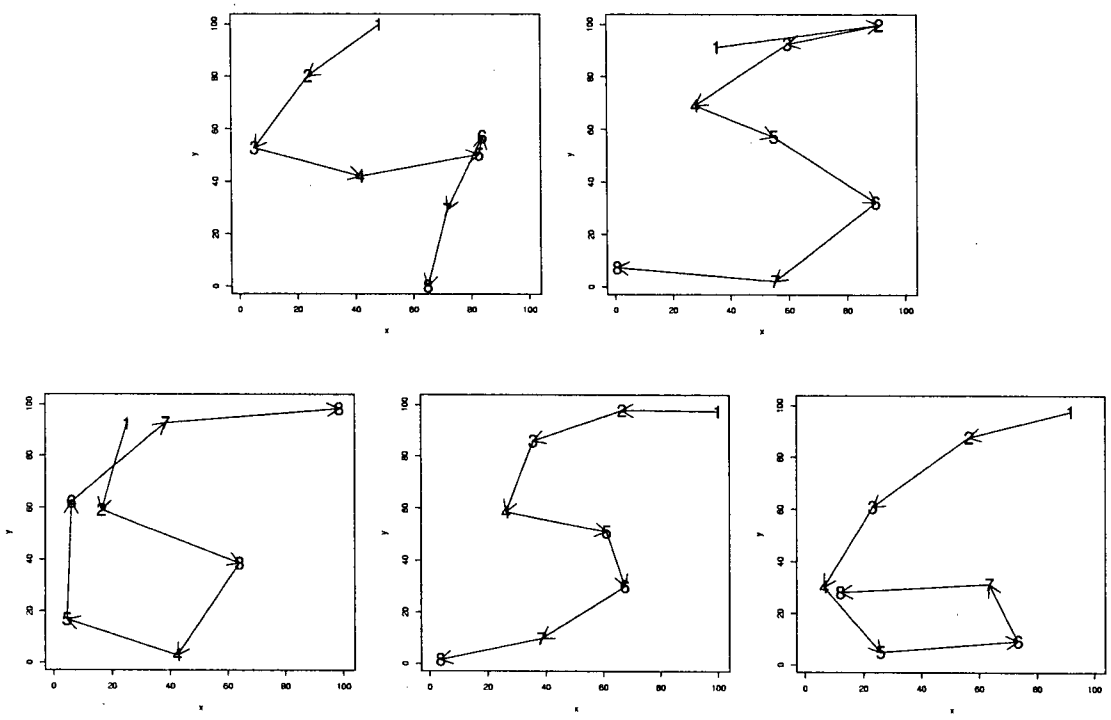


Figure 4.15: Top-left: Average of the digit 4 samples. Top-right: Average of the subcluster 1 of the digit 5 samples. Bottom-left: Average of the subcluster 2 of the digit 5 samples. Bottom-middle: Average of the subcluster 3 of the digit 5 samples. Bottom-right: Average of the digit 6 samples.

then for each sample, we apply the 2-split method. If 3 out of the 5 samples accept the split, then we split the cluster into two subclusters. We take the sample whose 2-split produce the maximum separation index and use nearest neighbor method to assign the rest of the data points to the two subclusters.

Nowadays, data sets become larger and larger. It is important for a new clustering method to handle huge data. The SEQCLUST method has the potential to handle huge data sets since its merging and splitting process can be done based merely on the mean vectors and covariance matrices. If the size of the data set is too large to be handled at once, then we can read a block of data points at a time. For the current block, we apply the SEQCLUST method, then we summarize the block by the number of clusters, the mean vectors and the covariance matrices. Finally, we check if clusters could be merged or not by using the cluster sample sizes, mean vectors, and covariance matrices.

We did not consider the effect of cluster sizes on the performance of the SEQCLUST method.

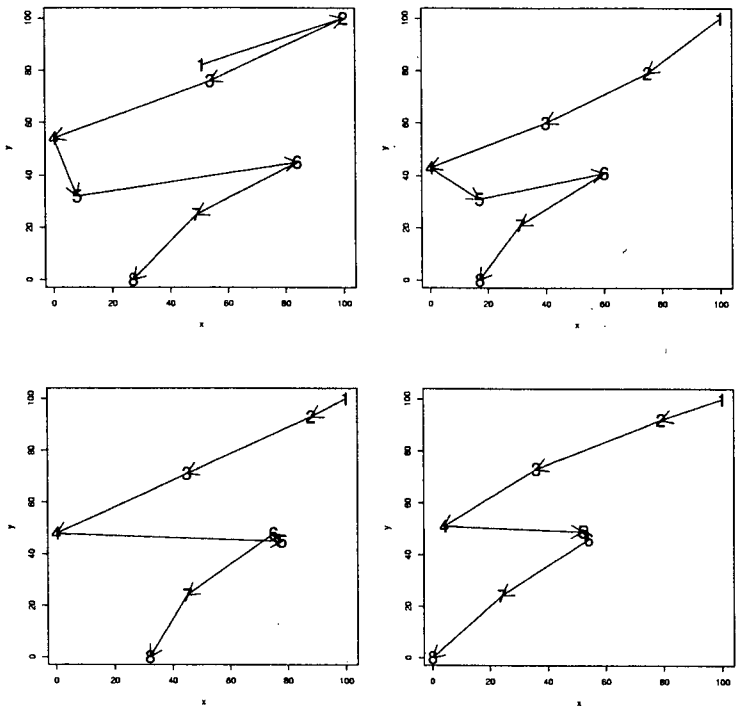


Figure 4.16: Top-left: Sample 59. Top-right: Sample 361; Bottom-left: Sample 398. Bottom-right: Sample 482.

If sample sizes are small relative to the number of dimensions (i.e. data are too sparse), then SEQCLUST tends to overestimate the number of clusters and hence might not perform better than simpler methods. It would be a interesting future research topic to see the effects of cluster sizes.

In this chapter, we also compare the performance of the SEQCLUST method with those of other number-of-cluster-estimation methods. We conclude that the `Hartigan`, `KL` and `silhouette` methods are not recommended for determining the number of clusters, but the `CH` method is acceptable. However, the `CH` method is not as good as the SEQCLUST method.

Table 4.15: Results for unscaled DAT3 ($k_0 = 7$, $n = 1000$, $p = 16$), which contains the digits 1, 3, 4, 6, 8, 9, and 0.

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard	system time
SEQCLUST (kmeans)	10	0.686	0.688	0.927	0.730	0.573	47.95
SEQCLUST (MKmeans)	10	0.758	0.760	0.946	0.794	0.651	55.37
SEQCLUST (PAM/CLARA)	9	0.636	0.638	0.912	0.687	0.523	36.13
SEQCLUST (Ward)	10	0.784	0.786	0.951	0.816	0.684	264.68
SEQCLUST (EMclust)	13	0.770	0.771	0.928	0.709	0.528	230.02
SEQCLUST (Mclust)	10	0.782	0.783	0.914	0.678	0.511	287.48
EMclust1	15	0.463	0.466	0.860	0.547	0.375	347.33
Mclust1	48	0.262	0.269	0.862	0.416	0.176	316.73
CH	3	0.217	0.218	0.640	0.455	0.255	3.06
Hartigan	23	0.459	0.464	0.886	0.567	0.342	2.89
KL	20	0.502	0.506	0.893	0.601	0.381	2.49
Silhouette	19	0.482	0.486	0.889	0.583	0.364	1.43
kmeans		0.557	0.560	0.889	0.622	0.452	0.01
MKmeans		0.517	0.520	0.873	0.594	0.420	0.08
PAM/CLARA		0.579	0.582	0.895	0.641	0.471	0.03
Ward		0.582	0.585	0.894	0.644	0.475	2.04
EMclust0		0.358	0.362	0.816	0.473	0.303	5.23
Mclust0		0.473	0.475	0.843	0.576	0.393	8.21

Table 4.16: Interval estimates of the number of clusters obtained by the SEQCLUST method for DAT3

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[9, 11]	SEQCLUST (Ward)	[10, 12]
SEQCLUST (MKmeans)	[10, 13]	SEQCLUST (EMclust)	[3, 13]
SEQCLUST (PAM/CLARA)	[7, 11]	SEQCLUST (Mclust)	[6, 10]

Table 4.17: The normal version separation index matrix for `DAT3` ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with `Ward`.

	1	2	3	4	5	6	7	8	9	10
1	-1.000	0.651	0.579	0.472	0.306	0.770	0.566	0.337	0.603	0.474
2	0.651	-1.000	0.777	0.684	0.728	0.239	0.577	0.470	0.764	0.431
3	0.578	0.777	-1.000	0.514	0.567	0.834	0.902	0.457	0.530	0.616
4	0.472	0.684	0.514	-1.000	0.290	0.788	0.715	0.497	0.356	0.657
5	0.306	0.728	0.567	0.290	-1.000	0.792	0.677	0.549	0.505	0.664
6	0.770	0.239	0.834	0.788	0.792	-1.000	0.782	0.603	0.833	0.294
7	0.566	0.577	0.902	0.715	0.676	0.782	1.000	0.636	0.545	0.550
8	0.337	0.470	0.457	0.497	0.549	0.603	0.636	-1.000	0.703	0.392
9	0.603	0.764	0.530	0.356	0.505	0.833	0.545	0.703	-1.000	0.635
10	0.474	0.432	0.616	0.657	0.664	0.293	0.550	0.392	0.635	-1.000

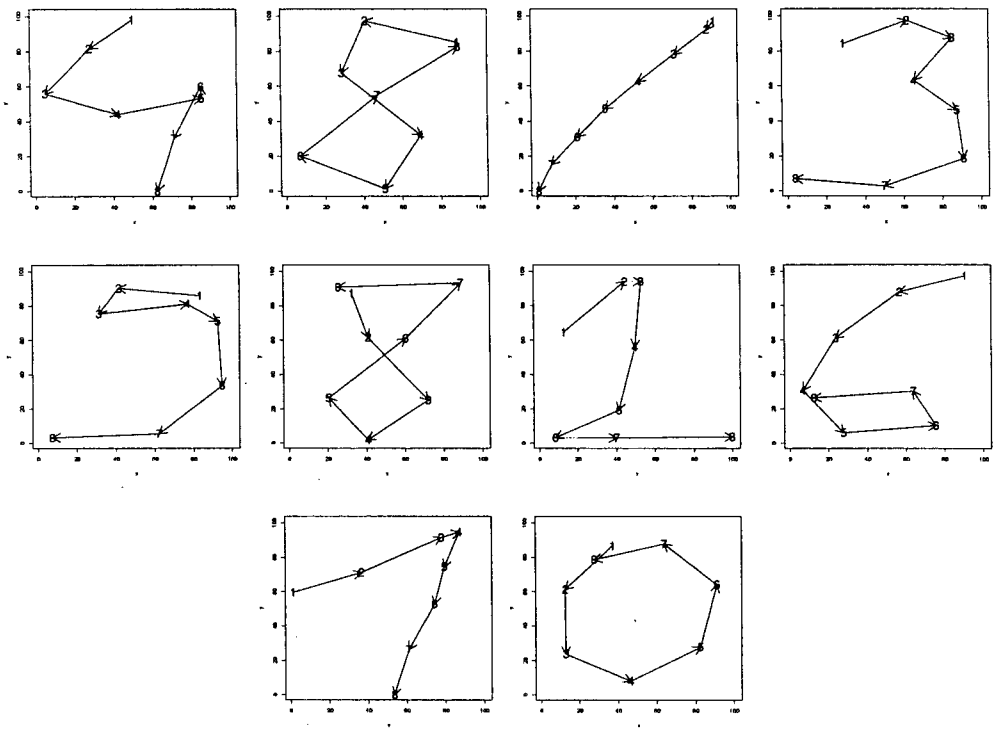


Figure 4.17: From left to right and from top to bottom, the digits are 4, 8, 1, 3, 9, 8, 1, 6, 1, 0

Table 4.18: The quantile version separation index matrix for `DAT3` ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with `Ward`.

	1	2	3	4	5	6	7	8	9	10
1	-1.000	0.658	0.592	0.480	0.316	0.776	0.564	0.285	0.568	0.439
2	0.658	-1.000	0.779	0.686	0.742	0.190	0.580	0.470	0.778	0.432
3	0.591	0.779	-1.000	0.512	0.582	0.839	0.914	0.413	0.479	0.608
4	0.480	0.686	0.512	-1.000	0.270	0.805	0.720	0.474	0.346	0.666
5	0.316	0.742	0.582	0.269	-1.000	0.790	0.658	0.497	0.484	0.697
6	0.776	0.190	0.839	0.805	0.790	-1.000	0.780	0.620	0.848	0.304
7	0.564	0.580	0.914	0.720	0.658	0.780	-1.000	0.615	0.587	0.544
8	0.285	0.470	0.413	0.474	0.497	0.620	0.615	-1.000	0.699	0.369
9	0.569	0.778	0.479	0.346	0.484	0.848	0.587	0.699	-1.000	0.619
10	0.439	0.432	0.608	0.666	0.697	0.304	0.544	0.369	0.619	-1.000

Chapter 5

Variable Weighting and Selection

5.1 Introduction

In cluster analysis, the division into clusters can depend on the variables that are used as well as how they are weighted. Usually researchers want to include all variables which might possibly be relevant in the hope that the dimensions upon which subgroups differ will be represented by one or more of these variables (Donoghue, 1995). However, including `noisy variables` can obscure the genuine differences of interest (Gordon, 1981, section 2.4.5), hence result in low recovery rates (Milligan, 1980). Even if the noisy variables would not obscure the genuine differences of interest, it is better to not include them in order to get a more parsimonious description, hence to increase the ability to produce output that can be visually inspected by a human. For example, in data-mining, there are many variables, and one cannot expect them all be useful in forming clusters. Deleting noisy variables can also save computing time and reduce the cost of future study. Variable weighting is also beneficial. Weights in cluster analysis are like regression coefficients in regression analysis. It says something about the relative importance of variables for prediction (in this case, prediction means predicting cluster membership). Noisy variables are not important to recover the cluster structure and should be assign zero weights.

In this chapter, we will investigate when noisy variables will mask the cluster structure and propose a new noisy-variable-detection method.

To the best of our knowledge, little literature addresses the issue: when noisy variables will mask the cluster structure. If we know when noisy variables will mask the cluster structure, then we

can have a rough idea if the clustering results are reliable or not. So it is worthwhile to investigate when noisy variables will mask the cluster structure.

Many methods have been proposed to eliminate or downweight the effect of noisy variables. Most methods are heuristic and theoretical properties of these methods were not given. Moreover, the existing methods assume that the true number of clusters is known. Furthermore, some methods are not easy to implement. We propose a new variable selection/weighting method, the population version of which has desirable theoretically properties. This new method does not require that the true number of clusters be known, and is easy to implement. The key ideas are (1) to construct a weight vector for a given partition based on the linear combination of the eigenvectors of the product matrix of the between-cluster distance matrix and the inverse of the within-cluster distance matrix; (2) to use a weight-vector-averaging technique so that the weight vector can be obtained without the knowledge of the true number of clusters.

In Sections 5.2 and 5.3, we give a brief review on variable selection and variable weighting methods in the cluster analysis setting. In Section 5.4 we give a mathematically rigorous definition of noisy variable and desired properties that a variable weighting/selection method should have. In Section 5.5, we investigate when noisy variables will mask the cluster structure. In Section 5.6, we propose a new variable weighting and selection method based on compact projection (CP method for short) which can be used both to estimate the variable weights and to delete noisy variables. In Section 5.7, we propose a weight-vector-averaging technique so that a weight vector can be obtained without the knowledge of the true number of clusters. A preliminary theoretical validation of this technique given in Section 5.8 shows that under certain conditions, the population distributions of the noisy variables do not change if we merge k_0 population clusters into $k_0 - 1$ population clusters or if we split k_0 population clusters into $k_0 + 1$ population clusters where k_0 is the true number of clusters. This result does not depend on any particular variable weighting or selection procedure. However, the sample distributions of the noisy variables may change after we split k_0 sample clusters into $k_0 + 1$ sample clusters. To overcome this difficulty, we propose in Section 5.9 to iteratively use the weight-vector-averaging technique. Based on the results obtained in Sections 5.8 and 5.9, we propose a variable weighting/selection algorithm in Section 5.10. In Section 5.11, we study the performance of the variable weighting/selection algorithm implemented by the CP method through simulated data sets and real data sets respectively. In Section 5.12, we implement the CP

method in the SEQCLUST algorithm proposed in Chapter 4. Discussion and future research are given in Section 5.13.

5.2 Literature Review of Variable Selection Methods in Cluster Analysis

In linear regression analysis, the most common method for variable selection is the step-wise method (Miller 2002) which includes forward selection (starting with a single variable and entering one variable at a time), backward elimination (starting with all variables and entering one variable at a time) and guided selection. Guided selection is a combination of the forward selection and backward elimination. After each variable is added, a test is made to see if any of the previously selected variables can be deleted. These approaches can also be used in cluster analysis (Beale et al. 1967; Fowlkes et al. 1988; Milioli 1999).

Jolliffe (1972) proposes a variable selection procedure in which clustering methods are used to classify variables into subgroups. After classifying variables into subgroups, Jolliffe (1972) selects one variable from each subgroup and uses these selected variables to classify objects. One major problem of this procedure is to decide how many subgroups to use and how to select representative variables.

Projection pursuit is a dimension-reducing technique by projecting multivariate data to lower dimension data, meanwhile keeping as much as possible of the structure information in the original data (Huber 1985; Jones and Sibson 1987). Montanari and Lizzani (2001) propose a variable selection procedure based on a one-dimensional projection. They first get a weight vector of variables based on the projection direction. Then they delete variables whose weights are small. The projection direction is the direction such that the projected data are far from normally distributed based on a χ^2 statistic. It is not easy to obtain the projection direction. A special optimization method, the simulated annealing algorithm, has been employed.

Principal component analysis is a special case of projection pursuit. The goal is to find a projection such that the projected data have as much dispersion as possible. The dispersion usually is measured by variance. Jolliffe (1972), Hastie et al. (2000), Ding (2003), and Liu et al. (2003) propose variable selection methods based on principal component analysis.

Many researchers propose univariate screening techniques to delete noisy variables. If a variable fails to pass through the screening, then it is regarded as a noisy variable. Donoghue (1995) proposed an index based on the skewness and the kurtosis to check if the variable is multimodal. The unimodal variables will be regarded as noisy variable and are deleted. Xing and Karp (2001) screen variables by using the knowledge of distinct biological states (e. g. either 'on' or 'off'). Li et al. (2003) use the coefficient of variation and the t-test to screen genes (variables). Bagirov et al. (2003) screen variables by an average pairwise between cluster distance. Ding (2003) use variance as an initial criterion to eliminate irrelevant genes (variables). Correlations among variables might affect the performance of univariate screening.

Carmone et al. (1999) also consider variables one by one. For each variable, Carmone et al. (1999) obtain a partition. Then they get a matrix of the agreement between each pair of the partitions. The agreement is measured by Hubert and Arabie's (1985) adjusted Rand index . Finally Carmone et al. (1999) delete variables which correspond to rows of the index matrix such that the sums of these rows are relatively small compared to other rows. The rationale behind their algorithm is that for a noisy variable, the partition will be random. Therefore, the Hubert and Arabie's adjusted Rand index will be small if one of two variables is noisy. Brusco and Cradit (2001) improve Carmone et al.'s (1999) method to handle possible high degree of correlation among noisy variables and to deal with multiple sets of true cluster structures in the same data set.

Friedman and Meulman (2004) point out that the weight vector for each individual cluster can be different and partially (or completely) overlap with those of other clusters. They obtain the set of weight vectors by minimizing an average negative penalized within cluster distance.

5.3 Literature Review of Variable Weighting Methods in Cluster Analysis

The goals of variable weighting are (1) producing a more compact cluster structure with interpretable rescaled variables; (2) improving the recovery rate of clustering algorithms; and (3) differentiating the importance of variables.

Kruskal (1972) proposes a variable weighting method based on a kind of index of condensation which depends on the collection of inter-point distances. This kind of index does not depend

on a tentative assignment of points to clusters and does not depend on any tentative description of the low-dimensional structures around which the condensation presumably occurs. However, Kruskal (1972) admits that it is difficult to select an appropriate condensation index to optimize (DeSarbo et al. 1984).

By iteratively minimizing the weighted average within cluster distance and clustering, Lumelsky (1982) proposes a variable weighting method which combines the weighting procedure and the clustering procedure. DeSarbo et al. (1984) and Makarenkov and Legendre (2001) proposed the same approach but with three different objective functions. DeSarbo et al. (1984) use a stress-like objective function to optimize weights of variables while Makarenkov and Legendre's (2001) objective function is slightly different from that of Lumelsky (1982). Green et al. (1990) shows that the performance of DeSarbo et al.'s (1984) algorithm depends on the approach used to find seed values for the initial k-means partitioning. De Soete et al. (1985) present an alternating least-squares algorithm to estimate simultaneously variable weights and the associated ultrametric tree in an optimal fashion. De Soete (1986, 1988) propose another variable weighting algorithm for ultrametric and additive tree clustering. Makarenkov and Legendre (2001) reported that the solutions for the optimization problem in De Soete (1986, 1988) may not be unique that degenerate solutions in the optimization for additive tree reconstruction represent a pervasive problem.

Gnanadesikan et al. (1995) propose several variable weighting methods without advance knowledge of the cluster structure. The weights are square roots of the diagonal elements of the weighting matrix M. Different weighting methods correspond to different choice of the matrix M. One promising choice of the matrix M is $M = \text{diag}\,(B^*)\,[\,\text{diag}\,(W^*)]^{-1}$ where B^* and W^* are between and within sum-of-cross-products matrices which are estimated without advance knowledge of the cluster structure (Art et al. 1982).

Carbonetto et al. (2003) propose a variable weighting method based on a Bayesian approach. They assume that data are from a mixture of multivariate normal distributions and assign a multivariate normal prior to the component means. The covariance matrix of this multivariate normal prior is diagonal. And the estimated diagonal elements of the covariance matrix are weights for variables.

One common problem of these variable weighting and selection methods is that there has not been a theoretical or empirical study of the effect of the variable weighting/selection over

misspecified number of clusters.

5.4 Noisy Variables in Mixture of Multivariate Distributions

In this section we introduce some assumptions and definitions needed in this Chapter.

For motivation and analysis, we assume that data come from a mixture of multivariate distributions with density $f(\boldsymbol{x}) = \sum_{k=1}^{k_0} \pi_k f_k(\boldsymbol{x})$ which has exactly k_0 modes, where $f_k(\boldsymbol{x})$ is a unimodal density and $\pi_k, k = 1, \ldots k_0$, are mixture coefficients such that $\pi_k > 0$ and $\sum_{k=1}^{k_0} \pi_k = 1$. We assume that the $f_k(\boldsymbol{x})$'s have distinct mode/mean vectors.

With the probability distributions, we can give somewhat rigorous definitions of cluster and noisy variable.

Definition 5.4.1 *A (population)* **cluster** C_i *is a multivariate distribution with unimodal density* $f_i(\boldsymbol{x})$.

Definition 5.4.2 *A set of variables* $X_1, \ldots, X_{p_2}$ *is called* **unimodal-noisy variables** *if the joint distribution of* $X_1, \ldots, X_{p_2}$, *is unimodal and is the same for all components* $f_k(\boldsymbol{x}), k = 1, \ldots, k_0$, *of the mixture of distributions* $f(\boldsymbol{x}) = \sum_{i=1}^{k_0} \pi_k f_k(\boldsymbol{x})$.

Definition 5.4.3 *A set of variables* $X_1, \ldots, X_{p_2}$, *is called* r**-moment-noisy variables** *if the first* r*-th moments of* $X_1, \ldots, X_{p_2}$, *is the same for all components* $f_k(\boldsymbol{x}), k = 1, \ldots, k_0$, *of the mixture of distributions* $f(\boldsymbol{x}) = \sum_{i=1}^{k_0} \pi_k f_k(\boldsymbol{x})$, *and* X_i *is uncorrelated with non-noisy variables.*

Definition 5.4.4 *A weighting scheme has* **scale equivariance property** *if the weight* w_i *for variable* X_i *and the weight* $w_i^{(c)}$ *for rescaled variable* $X_i^{(c)} = cX_i$ *satisfy* $w_i^{(c)} = w_i/c$, *where* c *is any positive scale.*

The last property is analogous to that in regression analysis.

Definition 5.4.5 *A weighting scheme has the* **noisy-variables-detecting property** *if the weights for noisy variables are zero or near zero.*

Variable weighting and selection methods should have the noisy-variable-detection property.

5.5 Effect of Noisy Variables

From simulation studies, we observed that noisy variables may or may not mask the true cluster structure. To the best of our knowledge, there is little or no research which addresses the question "when do noisy variables affect the recovery of the true cluster structure?" If we know when noisy variables have an effect, then we can have a rough idea if the clustering results are reliable.

Let's consider the simplest case in which there are only two clusters. Suppose that the first p_1 variables are non-noisy and the rest of the variables are noisy. Denote the mean vectors of the two cluster centers as

$$\boldsymbol{\mu}_i = \begin{pmatrix} \boldsymbol{\theta}_i \\ \boldsymbol{\beta} \end{pmatrix}, \quad i = 1, 2,$$

where θ_i is the mean vector of the i-th cluster in the first p_1 non-noisy dimensions and β is the mean vector of the noisy variables.

Then the difference between the distances from the data point $\boldsymbol{X} = (\boldsymbol{X}_1^T, \boldsymbol{X}_2^T)^T$ to the two cluster centers is

$$\begin{aligned} \|\boldsymbol{X} - \boldsymbol{\mu}_1\|^2 - \|\boldsymbol{X} - \boldsymbol{\mu}_2\|^2 &= \|\boldsymbol{X}_1 - \boldsymbol{\theta}_1\|^2 + \|\boldsymbol{X}_2 - \boldsymbol{\beta}\|^2 - \|\boldsymbol{X}_1 - \boldsymbol{\theta}_2\|^2 - \|\boldsymbol{X}_2 - \boldsymbol{\beta}\|^2 \\ &= \|\boldsymbol{X}_1 - \boldsymbol{\theta}_1\|^2 - \|\boldsymbol{X}_1 - \boldsymbol{\theta}_2\|^2 \end{aligned}$$

which is unrelated to the noisy variables. That is, the noisy variables will not affect the partition results if the mean vector of noisy variables is correctly estimated.

If cluster sizes are large enough and clusters are separated, then the estimation of $\boldsymbol{\beta}$ will be precise, i.e. the $\boldsymbol{\beta}_i$ will be very close ($\boldsymbol{\beta}_i$ is the mean vector of noisy component in the i-th cluster). Hence noisy variables have a small effect on the recovery of the true clusters. To illustrate this, we generate 100 data sets each of which contains two clusters from the multivariate normal distributions $\mathrm{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2$, where

$$\boldsymbol{\mu}_1 = \mathbf{0}_{p\times 1},\ \boldsymbol{\mu}_2 = \begin{pmatrix} 6 \\ 0 \\ \vdots \\ 0 \end{pmatrix},\ \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 10I_{p_2} \end{pmatrix}. \tag{5.5.1}$$

That is, the two clusters are separated in the first dimension. The remaining p_2 variables are noisy variables. The scalar 10 ensures that the variations of the noisy variables are similar to that of the non-noisy variable. Denote X_1 as the non-noisy variable and C as cluster membership ($C = 1, 2$ with probability 1/2 each). Then $\mathrm{Var}(X_1) = \mathrm{E}(\mathrm{Var}(X_1|C)) + \mathrm{Var}(\mathrm{E}(X_1|C)) = 1 + (6/2)^2 = 10$. Each cluster has the same size m. The `kmeans` clustering algorithm is used to obtain 2-cluster partitions. We use `kmeans 1` to indicate that the theoretical mean vectors μ_i, $i = 1, 2$, are used as the initial cluster centers for the `kmeans` clustering algorithm. `k-means 2` means that the initial cluster centers are the sample mean vectors of the two clusters in the true partition. `k-means 3` means that the initial cluster centers are randomly generated. The Hubert and Arabie's (1985) adjusted Rand index (HA index) is used to measure the agreement between the true partition and the partition obtained by the `kmeans` clustering algorithm. The value of the HA index for perfect agreement is 1. Table 5.1 shows the values of the HA index for different values of the cluster size m and the the number p_2 of the noisy variables.

Table 5.1: Simulation results for detecting the effect of noisy variables when cluster sizes are large. The entries in the table are the values of the Hubert and Arabie's (1985) adjusted Rand indexes and corresponding standard errors (in the parentheses).

m	algorithm	$p_2 = 0$	$p_2 = 1$	$p_2 = 5$	$p_2 = 10$
10	k-means 1	0.998 (0.020)	0.932 (0.187)	0.044 (0.141)	0.019 (0.084)
	k-means 2	0.998 (0.020)	0.932 (0.187)	0.042 (0.140)	0.020 (0.084)
	k-means 3	0.998 (0.020)	0.741 (0.393)	0.030 (0.104)	0.014 (0.077)
50	k-means 1	0.995 (0.013)	0.989 (0.021)	0.207 (0.397)	0.030 (0.172)
	k-means 2	0.995 (0.013)	0.989 (0.021)	0.207 (0.397)	0.030 (0.172)
	k-means 3	0.995 (0.013)	0.800 (0.366)	0.027 (0.1420	0.000 (0.012)
100	k-means 1	0.993 (0.012)	0.992 (0.014)	0.320 (0.463)	0.090 (0.286)
	k-means 2	0.993 (0.012)	0.992 (0.014)	0.320 (0.463)	0.090 (0.286)
	k-means 3	0.993 (0.012)	0.772 (0.406)	0.003 (0.010)	0.001 (0.009)
500	k-means 1	0.995 (0.004)	0.994 (0.005)	0.565 (0.492)	0.219 (0.413)
	k-means 2	0.995 (0.004)	0.994 (0.005)	0.565 (0.492)	0.219 (0.414)
	k-means 3	0.995 (0.004)	0.879 (0.313)	0.000 (0.002)	0.000 (0.002)
1000	k-means 1	0.994 (0.003)	0.994 (0.004)	0.675 (0.465)	0.239 (0.427)
	k-means 2	0.994 (0.003)	0.994 (0.004)	0.675 (0.465)	0.239 (0.427)
	k-means 3	0.994 (0.003)	0.916 (0.264)	0.030 (0.170)	0.000 (0.000)
10000	k-means 1	0.995 (0.001)	0.995 (0.001)	0.994 (0.001)	0.596 (0.489)
	k-means 2	0.995 (0.001)	0.995 (0.001)	0.994 (0.001)	0.596 (0.489)
	k-means 3	0.995 (0.001)	0.945 (0.215)	0.040 (0.196)	0.020 (0.140)

We see that the value of the adjusted Rand index increases as the number of data points increases, and decreases as the number of noisy variables increases. If cluster sizes are large enough and clusters are separated, then the noisy variables have little effect on clustering. When cluster sizes are equal to 10000 in the above example, the clustering results do not change too much when the number of noisy variables is up to 5. From Table 5.1, we also can see that the initial cluster centers may affect the results of the `kmeans` clustering algorithm.

If cluster sizes are not large enough, then whether the noisy variables matter or not might depend on the following factors:

- degree of separation among clusters
- clustering algorithms
- the number of noisy variables
- the relative variances of noisy variables to non-noisy variables.
- outliers

Since most partitioning algorithms are based on the minimization of the trace of linear combination of covariance matrices, we only need to make sure to get good estimates of covariance matrices. However, for real data sets, cluster sizes usually are not large enough to get good estimates of covariance matrices. So there is a need to develop algorithms to eliminate or at least downweight effects of noisy variables.

5.6 A New Variable Weighting and Selection Method

In this section, we propose a variable weighting and selection method based on a compact projection criterion (`CP` method for short). We first describe the motivation in Subsection 5.6.1 and then introduce the `CP` method I in Subsection 5.6.2. An extension is given in Subsection 5.6.3.

In the following, we use a symbol with $\hat{}$ to represent the sample version of a statistic and use the symbol without $\hat{}$ to represent the population version of the statistic. For example $\boldsymbol{\mu}_k$ is the population version of the mean vector of the k-th cluster and $\hat{\boldsymbol{\mu}}_k$ is the corresponding sample version.

5.6.1 Motivation

One purpose of variable weighting is to obtain a more compact cluster structure. That is, clusters are internally more cohesive and externally more isolated from each other. We first need mathematical definitions for the concepts "internal cohesion" and "external isolation". For one-dimensional data, one can define the **internal cohesion** (within-cluster variation) as the average within-cluster sum of squares:

$$\hat{a} = \frac{1}{n}\sum_{k=1}^{k_0}\sum_{j=1}^{n_k}\left(x_j^{(k)} - \bar{x}^{(k)}\right)^2,$$

where

$$n = \sum_{k=1}^{k_0} n_k, \quad \bar{x}^{(k)} = \frac{1}{n_k}\sum_{j=1}^{n_k} x_j^{(k)}.$$

And one can define the **external isolation** (between-cluster variation) as the average between-cluster sum of squares:

$$\hat{b} = \frac{1}{n}\sum_{k=1}^{k_0} n_k\left(\bar{x}^{(k)} - \bar{x}\right)^2,$$

where

$$\bar{x} = \frac{1}{n}\sum_{k=1}^{k_0}\sum_{j=1}^{n_k}\mathbf{x}_j^{(k)}.$$

Then the ratio of the isolation to the cohesion

$$\hat{r} = \frac{\hat{b}}{\hat{a}}$$

is a measure of the **compactness** of the clusters.

The definitions of cohesion and isolation can be easily extended to high-dimensional space:

$$\begin{aligned}\hat{\boldsymbol{A}} &= \frac{1}{n}\sum_{k=1}^{k_0}\sum_{j=1}^{n_k}\left(\boldsymbol{x}_j^{(k)} - \bar{\boldsymbol{x}}^{(k)}\right)\left(\boldsymbol{x}_j^{(k)} - \bar{\boldsymbol{x}}^{(k)}\right)^T,\\ \hat{\boldsymbol{B}} &= \frac{1}{n}\sum_{k=1}^{k_0} n_i\left(\bar{\boldsymbol{x}}^{(k)} - \bar{\boldsymbol{x}}\right)\left(\bar{\boldsymbol{x}}^{(k)} - \bar{\boldsymbol{x}}\right)^T.\end{aligned}$$

Suppose that $n_k/n \to \pi_k$ as $n \to \infty$. Assuming no misclassification of points to clusters, by the law of large numbers, we can get

$$\hat{\boldsymbol{A}} \xrightarrow{p} \boldsymbol{A}, \quad \hat{\boldsymbol{B}} \xrightarrow{p} \boldsymbol{B}, \quad \text{as } n \to \infty,$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are the population versions of `internal cohesion` and `external isolation` in high dimensional space.

$$\begin{aligned} \boldsymbol{A} &= \sum_{k=1}^{k_0} \pi_k \boldsymbol{\Sigma}_k, \\ \boldsymbol{B} &= \sum_{k=1}^{k_0} \pi_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^T, \end{aligned} \tag{5.6.2}$$

$\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are the mean vector and positive definite covariance matrix of the k-th component of the mixture of distributions $f(\boldsymbol{x}) = \sum_{k=1}^{k_0} f_k(\boldsymbol{x})$, and

$$\boldsymbol{\mu} = \sum_{k=1}^{k_0} \pi_k \boldsymbol{\mu}_k.$$

By simple algebra, we can get

$$\boldsymbol{B} = \sum_{k_1=1}^{k_0-1} \sum_{k_2=k_1+1}^{k_0} \pi_{k_1} \pi_{k_2} (\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2})(\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2})^T$$

We can use $\boldsymbol{A}^{-1}\boldsymbol{B}$ as a measure of compactness. However, there is no scalar measure to compare two matrices. One way to overcome this difficulty is that we first project the data into one dimensional space, then we use the definition of the compactness for one-dimensional space. Denote the projection direction as $\boldsymbol{u}$. Then the projected means and variances are $\boldsymbol{u}^T\boldsymbol{\mu}_k$ and $\boldsymbol{u}^T\boldsymbol{\Sigma}_k\boldsymbol{u}$, $k = 1, \ldots, k_0$. And the population versions of the within- and between-cluster sum of squares are

$$a(\boldsymbol{u}) = \boldsymbol{u}^T\boldsymbol{A}\boldsymbol{u}, \quad b(\boldsymbol{u}) = \boldsymbol{u}^T\boldsymbol{B}\boldsymbol{u}.$$

The compactness for the projection is

$$r(\boldsymbol{u}) = \frac{\boldsymbol{u}^T\boldsymbol{B}\boldsymbol{u}}{\boldsymbol{u}^T\boldsymbol{A}\boldsymbol{u}}.$$

A larger value of r means that the projection direction $\boldsymbol{u}$ has greater between-cluster differences relatively to the within-cluster differences in the projection, i.e. the cluster structure is more compact in the projection.

If $\boldsymbol{u}$ is a projection direction, the $c\boldsymbol{u}$ ($c > 0$) is also a projection direction. We normalize the projection direction $\boldsymbol{u}$ so that $\max_i |u_i| = 1$, where u_i is the i-th element of $\boldsymbol{u}$.

We want to find a projection direction $\boldsymbol{u}^*$ which maximizes $r(\boldsymbol{u})$. That is, $\boldsymbol{u}^*$ is the solution of the maximization problem

$$\max_{\boldsymbol{u}} \frac{\boldsymbol{u}^T \boldsymbol{B} \boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{A} \boldsymbol{u}}, \quad \text{such that } \max_i |u_i| = 1. \tag{5.6.3}$$

It is well-known that the solution for (5.6.3) is $\boldsymbol{u}^* = \boldsymbol{\alpha}_1$ which is the normalized eigenvector ($\max_i |\alpha_{1i}| = 1$, where α_{1i} is the i-th elements of $\boldsymbol{\alpha}_1$), corresponding to the maximum eigenvalue λ_1 of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$ (e. g. Gnanadesikan 1977, Section 4.2). In fact, if we know the class labels of data points, then the optimization problem (5.6.3) is the well-known linear discriminant analysis problem. In clustering problems, we do not know the cluster labels of data points. We obtain the cluster mean vectors and covariance matrices based on the partition of a clustering method.

We can use $\boldsymbol{\alpha}_1$ as a "weight vector" (the elements of $\boldsymbol{\alpha}_1$ may be negative, so $\boldsymbol{\alpha}_1$ may not be a true weight vector). The value of the compactness r increases after we weight variables with $\boldsymbol{\alpha}_1$ because $r(\boldsymbol{\alpha}_1) \geq r(\mathbf{1})$, where $\mathbf{1}$ is the vector whose elements are all equal to one and $r(\mathbf{1})$ is the value of compactness with equal weighting. We give a small numerical example to illustrate this. Suppose there are 2 clusters (each having 150 data points) generated from two bivariate normal distributions $\mathrm{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2$, where

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ \boldsymbol{\mu}_2 = \begin{pmatrix} 6 \\ 2 \end{pmatrix}, \ \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & -0.5 \\ -0.5 & 2 \end{pmatrix}.$$

By plugging in the sample proportions $\hat{\pi}_k$, mean vectors $\hat{\boldsymbol{\mu}}_k$, and covariance matrices $\hat{\boldsymbol{\Sigma}}_k$, $k = 1, 2$, to obtain $\boldsymbol{A}$ and $\boldsymbol{B}$, we can get $\hat{\boldsymbol{\alpha}}_1 = (1.000, 0.370)^T$, $\hat{r}((1.000, 1.000)^T) = 7.420$, and $\hat{r}(\hat{\boldsymbol{\alpha}}_1) = 11.868$. The scatter plot of original and weighted data sets are shown in the left and right panel of Figure 5.1 respectively.

Moreover the vector $\boldsymbol{\alpha}_1$ depends on the orientations of the clusters even if the mean vectors are the same. For example, if we change the covariance in the matrix $\boldsymbol{\Sigma}_2$ in the previous example from -0.5 to 0.5, then $\hat{\boldsymbol{\alpha}}_1 = (1.000, 0.201)^T$. The estimated r values for the original and weighted data are 5.582 and 9.902 respectively. The scatter plot of original and weighted data sets are shown in the left and right panel of Figure 5.2 respectively. Some variable weighting methods, such as Lumelsky's (1982) method, depend only on the diagonal elements of the covariance matrices. However covariance information is also useful to determine the variable weights as these two small examples illustrate.

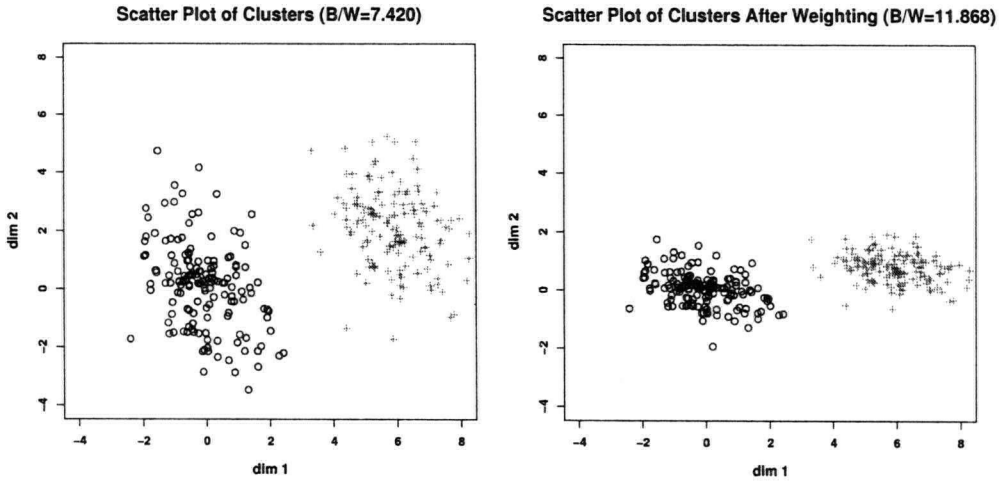


Figure 5.1: The effect of variable weighting. After weighting, the ratio of the between-cluster distance to the within-cluster distance increases from 7.420 to 11.868. The weight vector is $(1.000, 0.370)^T$.

We will show in the next subsection that the elements of $\boldsymbol{\alpha}_1$ corresponding to noisy variables are zero. Thus we can construct a weight vector based on $\boldsymbol{\alpha}_1$ so that the weight vector has noisy-variable-detection property. Since $\boldsymbol{\alpha}_1$ is a projection direction related to the measure of the `compactness` of the clusters $r(\boldsymbol{u})$, we call the variable weighting/selection methods based on $\boldsymbol{\alpha}_1$ as `compact projection` methods (`CP` methods for short).

5.6.2 `CP` Method I

By the `CP` method, we select or weight variables based on the optimal projection direction $\boldsymbol{u}^* = \boldsymbol{\alpha}_1$. We first study some properties of the eigenvectors $\boldsymbol{\alpha}_i$, $i = 1, \ldots, p$, where p is the number of variables.

Theorem 5.6.1 *Suppose that the first p_1 variables $X_1, \ldots, X_{p_1}$ are non-noisy variables and the remaining p_2 variables $X_{p_1+1}, \ldots, X_p$ are noisy variables. Also suppose that the covariance matrices $\boldsymbol{\Sigma}_k$, $k = 1, \ldots, k_0$, are positive definite matrices. Then the eigenvectors $\boldsymbol{\alpha}_i$, $i = 1, \ldots, p_1$ corresponding to the positive eigenvalues, $\lambda_1 \geq \cdots \geq \lambda_{p_1} > 0$, of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$ have the form $\boldsymbol{\alpha}_i = (\boldsymbol{\beta}_i^T, \boldsymbol{0}^T)$, where $\boldsymbol{\beta}_i$ is a $p_1 \times 1$ non-zero vector.*

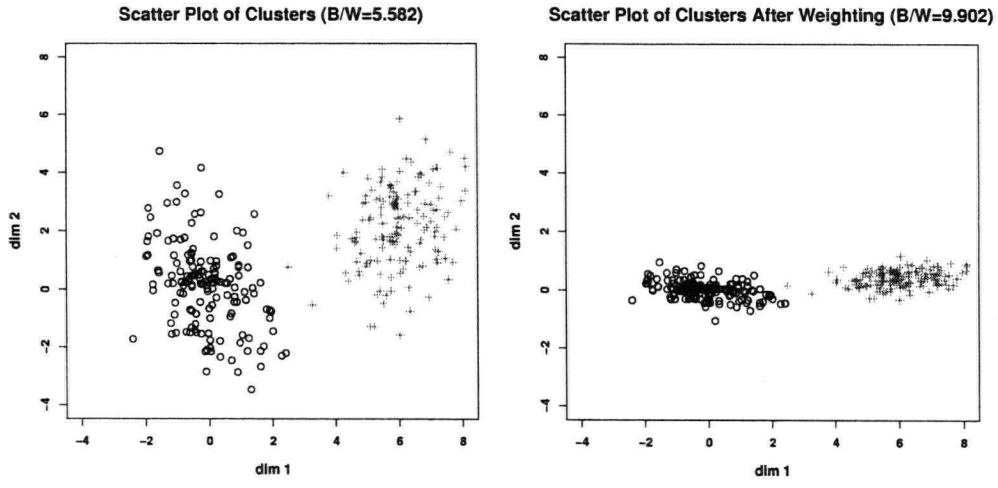


Figure 5.2: The effect of variable weighting. After weighting, the ratio of the between-cluster distance to the within-cluster distance increases from 5.582 to 9.902. The weight vector is $(1.000, 0.201)^T$.

[Proof] By the definition of noisy variables, the mean vectors and covariance matrices of components f_k, $k = 1, \ldots, k_0$, of the mixture of distributions f can be partitioned as

$$\boldsymbol{\mu}_k = \begin{pmatrix} \boldsymbol{\theta}_k \\ \boldsymbol{\theta} \end{pmatrix}, \quad \boldsymbol{\Sigma}_k = \begin{pmatrix} \boldsymbol{V}_{k1} & 0 \\ 0 & \boldsymbol{V} \end{pmatrix}, \quad k = 1, \ldots, k_0,$$

where $\boldsymbol{\theta}_k$'s are $p_1 \times 1$ vectors, $\boldsymbol{\theta}$ is a $p_2 \times 1$ vector, $\boldsymbol{V}_{k1}$'s are $p_1 \times p_1$ matrices, and $\boldsymbol{V}$ is a $p_2 \times p_2$ matrix. Then we can get

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 & 0 \\ 0 & \boldsymbol{A}_2 \end{pmatrix}, \quad \boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{A}^{-1}\boldsymbol{B} = \begin{pmatrix} \boldsymbol{A}_1^{-1}\boldsymbol{B}_1 & 0 \\ 0 & 0 \end{pmatrix}.$$

where $\boldsymbol{A}_1 = \sum_{k=1}^{k_0} \pi_k \boldsymbol{V}_{k1}$ and $\boldsymbol{A}_2 = \boldsymbol{V}$ and

$$\boldsymbol{B}_1 = \sum_{k_1=1}^{k_0-1} \sum_{k_2=k_1+1}^{k_0} \pi_{k_1}\pi_{k_2} \left[(\boldsymbol{\theta}_{k_1} - \boldsymbol{\theta}_{k_2})(\boldsymbol{\theta}_{k_1} - \boldsymbol{\theta}_{k_2})^T\right].$$

Denote $\lambda_1 \geq \cdots \geq \lambda_p$ as eigenvalues of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$.

Decompose the eigenvector $\boldsymbol{\alpha}_i$ of the eigenvalue λ_i as $\boldsymbol{\alpha}_i = (\boldsymbol{\beta}_i^T, \boldsymbol{\xi}_i^T)^T$, $i = 1, \ldots, p$. By the definition of the eigenvalue and eigenvector,

$$\begin{pmatrix} \boldsymbol{A}_1^{-1}\boldsymbol{B}_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_i \\ \boldsymbol{\xi}_i \end{pmatrix} = \lambda_i \begin{pmatrix} \boldsymbol{\beta}_i \\ \boldsymbol{\xi}_i \end{pmatrix}.$$

Therefore $\xi_i = 0$ for positive eigenvalues λ_i, $i = 1, \ldots, p_1$. □

Next we study the relation between the eigenvectors $\boldsymbol{\alpha}_i^*$, $i = 1, \ldots, p$, before and after the transformation $\boldsymbol{Y} = \boldsymbol{HX}$, where the random vector $\boldsymbol{X}$ has the density function $f(\boldsymbol{x}) = \sum_{k=1}^{k_0} \pi_k f_k(\boldsymbol{x})$. We first introduce the following lemmas.

Lemma 5.6.2 *The non-zero eigenvalues of the matrix $\boldsymbol{AB}$ are the same as the non-zero eigenvalues of the matrix $\boldsymbol{BA}$.*

[**Proof**] Suppose λ is a non-zero eigenvalue of the matrix $\boldsymbol{AB}$. Then there exists a non-zero vector $\boldsymbol{\alpha}$ such that $\boldsymbol{AB\alpha} = \lambda\boldsymbol{\alpha}$. Multiplying $\boldsymbol{B}$ on both sides of the equation, we get $\boldsymbol{BA}(\boldsymbol{B\alpha}) = \lambda(\boldsymbol{B\alpha})$, where $\boldsymbol{B\alpha}$ is non-zero. By definition of eigenvalue, λ is a non-zero eigenvalue of the matrix $\boldsymbol{BA}$.

Similarly, we can prove that if λ is a non-zero eigenvalue of the matrix $\boldsymbol{BA}$, then it is also a non-zero eigenvalue of the matrix $\boldsymbol{AB}$. □

Lemma 5.6.3 *Suppose that $\boldsymbol{Y} = \boldsymbol{HX}$, where $\boldsymbol{H}$ is a non-singular matrix. Denote $\boldsymbol{A}^* = \boldsymbol{HAH}^T$ and $\boldsymbol{B}^* = \boldsymbol{HBH}^T$. Then*

(1) *The i-th largest non-zero eigenvalue λ_i^* of the matrix $[\boldsymbol{A}^*]^{-1}\boldsymbol{B}^*$ is equal to the i-th largest non-zero eigenvalue λ_i of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$, $i = 1, \ldots p$;*

(2) *$\boldsymbol{\alpha}_i^* = \boldsymbol{H}^{-T}\boldsymbol{\alpha}_i$, where $\boldsymbol{H}^{-T} = (\boldsymbol{H}^T)^{-1}$, $\boldsymbol{\alpha}_i^*$ is the eigenvector of the matrix $[\boldsymbol{A}^*]^{-1}\boldsymbol{B}^*$ corresponding to λ_i^* and $\boldsymbol{\alpha}_i$ is the eigenvector of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$ corresponding to λ_i, $i = 1, \ldots, p$.*

[**Proof**] By simple algebra,

$$[\boldsymbol{A}^*]^{-1}\boldsymbol{B}^* = \boldsymbol{H}^{-T}\boldsymbol{A}^{-1}\boldsymbol{BH}^T.$$

By Lemma 5.6.2, the non-zero eigenvalues of the matrix $\boldsymbol{H}^{-T}\boldsymbol{A}^{-1}\boldsymbol{BH}^T$ are the same as the non-zero eigenvalues of the matrix $\boldsymbol{A}^{-1}\boldsymbol{BH}^T\boldsymbol{H}^{-T} = \boldsymbol{A}^{-1}\boldsymbol{B}$. Thus, $\lambda_i^* = \lambda_i$, $i = 1, \ldots, p$.

Now we show the second part of the Lemma. By definition of eigenvalue and eigenvector,

$$\boldsymbol{A}^{-1}\boldsymbol{B\alpha}_i = \lambda_i\boldsymbol{\alpha}_i, \ i = 1, \ldots, p.$$

Thus,

$$\boldsymbol{H}^{-T}\boldsymbol{A}^{-1}\boldsymbol{B\alpha}_i = \lambda_i\boldsymbol{H}^{-T}\boldsymbol{\alpha}_i, \ i = 1, \ldots, p.$$

Moreover

$$\boldsymbol{H}^{-T}\boldsymbol{A}^{-1}\boldsymbol{B} = \left(\boldsymbol{H}^{-T}\boldsymbol{A}^{-1}\boldsymbol{H}^{-1}\right)\left(\boldsymbol{H}\boldsymbol{B}\boldsymbol{H}^{T}\right)\boldsymbol{H}^{-T} = \left[\boldsymbol{A}^{*}\right]^{-1}\boldsymbol{B}^{*}\boldsymbol{H}^{-T}.$$

Therefore

$$\left[\boldsymbol{A}^{*}\right]^{-1}\boldsymbol{B}^{*}\left(\boldsymbol{H}^{-T}\boldsymbol{\alpha}_i\right) = \lambda_i^{*}\left(\boldsymbol{H}^{-T}\boldsymbol{\alpha}_i\right),\ i = 1,\ldots,p,$$

i.e. $\boldsymbol{H}^{-T}\boldsymbol{\alpha}_i$ is the eigenvector of the matrix $\left[\boldsymbol{A}^{*}\right]^{-1}\boldsymbol{B}^{*}$, corresponding to the eigenvalue λ_i^{*}. □

From the above analyses, we can see that the elements of the eigenvector $\boldsymbol{\alpha}_1$ corresponding to the noisy variables are zero and that $\boldsymbol{\alpha}_1$ has the scale equivariance property (see Definition 5.4.4). Thus if our purpose is to delete noisy variables or to downweight noisy variables, then we can use $\boldsymbol{\alpha}_1$ as the weight vector $\boldsymbol{w}$. That is

$$\boldsymbol{w} = \boldsymbol{\alpha}_1. \tag{5.6.4}$$

However it is possible that some elements of the normalized eigenvector $\boldsymbol{\alpha}_1$ are negative. And there is no meaningful interpretation for negative weight. So we instead can use the absolute values of the elements of the eigenvector $\boldsymbol{\alpha}_1$ as the weight vector $\boldsymbol{w}^I$. i.e.

$$\boldsymbol{w}^I = \begin{pmatrix} |\alpha_{11}| \\ \ldots \\ |\alpha_{1p}| \end{pmatrix}. \tag{5.6.5}$$

Theorem 5.6.4 *The weight vector $\boldsymbol{w}^I$ has the 1-moment-noisy variable detection property.*

[**Proof**] From Theorem 5.6.1, we know that the elements of the eigenvector $\boldsymbol{\alpha}_1$ which correspond to noisy variables are zero. Thus, the corresponding elements of the weight vector $\boldsymbol{w}$ are zero. □

Theorem 5.6.5 *The weight vector $\boldsymbol{w}^I$ has the scale equivariance property.*

[**Proof**] From Lemma 5.6.3, we know that if we do linear transformation $\boldsymbol{Y} = \boldsymbol{H}\boldsymbol{X}$, where $\boldsymbol{H} = \mathrm{diag}\left(h_1,\ldots,h_p\right)$ and $h_j > 0$, $j = 1,\ldots,p$, then

$$\boldsymbol{\alpha}_1^{*} = \boldsymbol{H}^{-T}\boldsymbol{\alpha}_1 = \begin{pmatrix} h_1^{-1}\alpha_{11} \\ \ldots \\ h_p^{-1}\alpha_{1p} \end{pmatrix}.$$

Thus, $w_j^{I*} = h_j^{-1} w_j^I$ if $Y_j = h_j X_j$, $j = 1, \ldots, p$. □

If only variable selection is required, we can let non-zero elements of $\boldsymbol{w}^I$ be 1.

For a data set, the assumptions of the population version of the **CP** method I could not be exactly hold. However, we expect the two nice properties approximately hold for the data version of the **CP** method I. That is, the weights for noisy variables are close to zero and $w_Y \simeq w_X/c$, where $c > 0$ (provided no variable dominants in a distance metric).

5.6.3 CP Method II

It is possible that some elements of the weight vector $\boldsymbol{w}^I$ corresponding to the non-noisy variables are zero. Thus, if we use $\boldsymbol{w}^I$ as the weight vector, we might delete non-noisy variables.

We illustrate the CP method with the `Ruspini` data set (Ruspini 1970) which consists of 75 observations in a 2-dimensional space (see Figure 5.3). There are 4 obvious groups of data points so

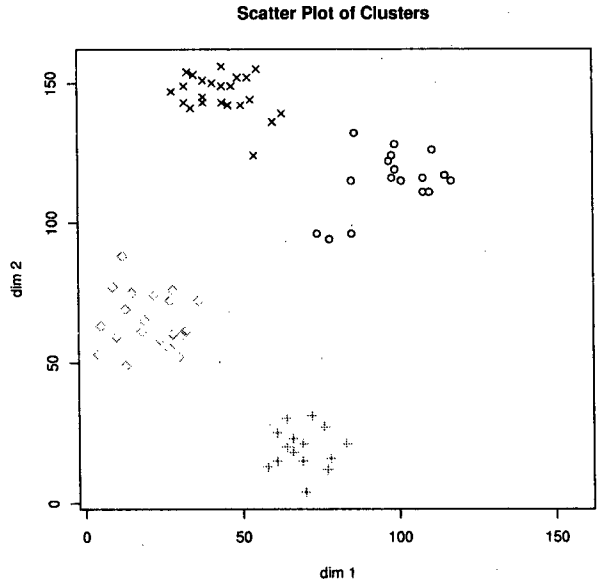


Figure 5.3: Scatter plot of the `Ruspini` data set

we use `MKmeans` to obtain a 4-cluster partition and $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ are replaced by the cluster sample mean vectors and sample covariance matrices. The weight vector is $\hat{\boldsymbol{w}}^I = (0.041, 1.000)^T$. However, the first variable is non-noisy.

To obviate this problem, we can use several eigenvectors instead of only the eigenvector

corresponding to the maximum eigenvalue. More generally, we can use weighted eigenvectors

$$\boldsymbol{w}^{II} = \frac{1}{\tau}\sum_{j=1}^{p} t_j|\boldsymbol{\alpha}_j|,\ t_j > 0, \tag{5.6.6}$$

where τ is the maximum element of the vector $\sum_{j=1}^{p} t_j|\boldsymbol{\alpha}_j|$, $\boldsymbol{\alpha}_j$, $j = 1,\ldots,p$, are eigenvectors of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$. To make sure that the elements of $\boldsymbol{w}^{II}$ corresponding to the noisy variables are still zero, we require that the weights t_j for eigenvectors corresponding to the eigenvalue zero are zero (see Theorem 5.6.1).

For example, we can set $t_j = \lambda_j$, $j = 1,\ldots,p$. λ_j is a measure of the degree of separation among clusters along the direction $\boldsymbol{\alpha}_j$. So it is reasonable to assign more weight to the eigenvector $\boldsymbol{\alpha}_j$ along which the degree of separation among clusters are larger.

For the `Ruspini` data set, the eigenvalues and eigenvectors of the matrix $\hat{\boldsymbol{A}}^{-1}\hat{\boldsymbol{B}}$ are $\hat{\lambda}_1 = 27.972$, $\hat{\lambda}_2 = 8.187$, $\hat{\boldsymbol{\alpha}}_1 = (-0.005, 0.111)^T$ and $\hat{\boldsymbol{\alpha}}_2 = (0.100, -0.002)^T$ respectively. And the new weight vector is $\hat{\boldsymbol{w}}^{II} = (0.302, 1.000)^T$ which improves the weight vector $\hat{\boldsymbol{w}}^{I} = (0.041, 1.000)^T$ in the sense that the element of $\hat{\boldsymbol{w}}^{II}$ is larger and not close to zero.

Other linear combinations of eigenvectors are also possible as long as the weights for eigenvectors corresponding to the eigenvalue zero are zero. We call this class of weight vectors as **CP weight vectors**. Subsequently, the term **CP** weight vector specifically refers to the weight vector (5.6.6) with $t_i = \lambda_i$, $i = 1,\ldots,p$.

5.7 Weight Vector Averaging

In previous sections, we assume that the true number of clusters and cluster membership are known (Ball and Hall 1965; Milligan and Cooper 1985; Zhuang et al. 1996; Frigui and Krishnapuram 1999; Stephens 2000; Comaniciu and Meer 2002; Fraley and Raftery 2002; Sugar and James 2003). However, the true number of clusters is usually unknown in real data sets. Many methods have been proposed to estimate the number of clusters. However their performance will be affected by noisy variables. So we should estimate the number of clusters after we remove or downweight the effects of noisy variables. Therefore, it is desirable that the variable selection/weighting procedures do not depend much on the specification of the number of clusters.

One possible way is to first find weight vectors based on a sequence of specifications for

the number of clusters and then use the average weight vector $\boldsymbol{w}^{III}$ normalized by its maximum element as the final weight vector.

To illustrate the idea, we generate a small data set consisting of 5 clusters in a 3-dimensional space, each having 100 data points. The five clusters are generated from trivariate normal distributions $\mathrm{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, $k = 1, \ldots, 5$, where

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_2 = \begin{pmatrix} 12 \\ 0 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_3 = \begin{pmatrix} 0 \\ 12 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_4 = \begin{pmatrix} 10 \\ 12 \\ 0 \end{pmatrix}, \boldsymbol{\mu}_5 = \begin{pmatrix} 6 \\ 6 \\ 0 \end{pmatrix},$$

$$\boldsymbol{\Sigma}_1 = \cdots = \boldsymbol{\Sigma}_5 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 20 \end{pmatrix}.$$

The third variable is a noisy variable and its variance is 20 so that the variation over the combined clusters of the noisy variable is similar to those of the non-noisy variables. The scatter plot of the five clusters in the first two dimensional space is shown in Figure 5.4. We first obtain the weight vectors

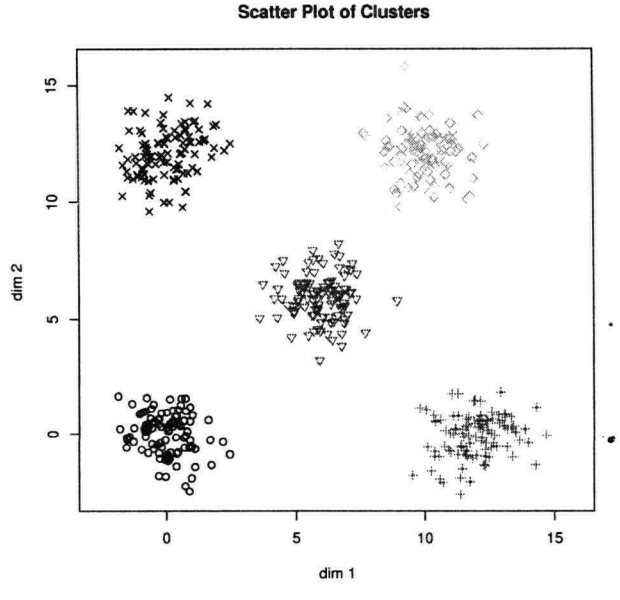


Figure 5.4: Scatter plot of the simulated data set

for $k_0 = 2, \ldots, 10$ and then obtain the normalized average weight vector. When calculating the weight vectors, the sample cluster mean vectors $\hat{\boldsymbol{\mu}}_k^{(k_0)}$ and covariance matrices $\hat{\boldsymbol{\Sigma}}_k^{(k_0)}$, $k = 1, \ldots, 5$,

are used after applying a clustering method specifying k_0 clusters. Note that if k_0 is not the true number of clusters, then $\hat{\boldsymbol{\mu}}_k^{(k_0)}$ and $\hat{\boldsymbol{\Sigma}}_k^{(k_0)}$ may not be estimating population quantities.

The results are listed in Table 5.2; we can see that the noisy variable (the third variable) has weight close to zero for each k_0. After weight vector averaging, the weight is still close to zero. This example motivates us to average the weight vectors for a series of specification of the number of clusters if the true number of clusters is unknown.

Table 5.2: Weight vectors for the simulated data set

k_0	$\hat{w}_1^I$	$\hat{w}_2^I$	$\hat{w}_3^I$	$\hat{w}_1^{II}$	$\hat{w}_2^{II}$	$\hat{w}_3^{II}$
2	0.025	1.000	0.127	0.025	1.000	0.127
3	0.017	1.000	0.051	0.229	1.000	0.057
4	1.000	0.696	0.019	1.000	0.713	0.021
5	1.000	0.705	0.005	1.000	0.726	0.011
6	1.000	0.736	0.023	1.000	0.834	0.029
7	0.601	1.000	0.013	1.000	0.820	0.020
8	0.934	1.000	0.058	0.938	1.000	0.043
9	0.662	1.000	0.020	1.000	0.822	0.047
10	0.047	1.000	0.043	1.000	0.994	0.054
$\hat{\boldsymbol{w}}^{III}$	$(0.650, 1.000, 0.044)^T$			$(0.909, 1.000, 0.052)^T$		

The results for the `Ruspini` data set are listed in Table 5.3. Again weight vector averaging has better performance.

Table 5.3: Weight vectors for the `Ruspini` data set

k_0	$\hat{w}_1^I$	$\hat{w}_2^I$	$\hat{w}_1^{II}$	$\hat{w}_2^{II}$
2	0.673	1.000	0.673	1.000
3	0.783	1.000	0.802	1.000
4	0.041	1.000	0.302	1.000
5	0.286	1.000	0.517	1.000
6	0.270	1.000	0.636	1.000
7	0.190	1.000	0.333	1.000
8	0.130	1.000	0.364	1.000
9	0.174	1.000	0.628	1.000
10	0.070	1.000	0.663	1.000
$\hat{\boldsymbol{w}}^{III}$	$(0.291, 1.000)^T$		$(0.577, 1.000)^T$	

In practice, researchers can use subject matter knowledge to determine a range of the number of clusters that should hopefully include the true number of cluster. In the next subsection, we provide a preliminary theoretical validation for the weight vector averaging technique.

5.8 A Preliminary Theoretical Validation of the Weight Vector Averaging

In this section, we assume that (1) k_0 (population) clusters are from a mixture of multivariate normal distributions with k_0 component $f(\boldsymbol{x}) = \sum_{k=1}^{k_0} \pi_k f_k(\boldsymbol{x})$, $0 < \pi_k < 1$, $\sum_{i=1}^{k_0} \pi_k = 1$; (2) when splitting, only one cluster will be split by a separating hyperplane; (3) when merging, only two clusters will be merged.

We will show that under certain conditions, noisy variables are still noisy variables if we split the k_0 clusters into $k_0 + 1$ clusters or if we merge the k_0 clusters into $k_0 - 1$ clusters.

To do so, we first obtain the mean vectors and covariance matrices of the two parts of a multivariate normal distribution $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ which is truncated by a separating hyperplane in Subsection 5.8.1. Then we show in Subsection 5.8.2 that to minimize the within-cluster distance, the optimal separating hyperplane passes through the mean vector $\boldsymbol{\mu}$ of the multivariate normal distribution and is orthogonal to the eigenvector $\boldsymbol{\alpha}_1$ corresponding to the maximum eigenvalue λ_1 of the covariance matrix $\boldsymbol{\Sigma}$. In Subsection 5.8.3, we show that under certain conditions, the cluster having the largest eigenvalue will be split when we split the k_0 clusters into $k_0 + 1$ clusters. In Subsection 5.8.4, we show that under certain conditions, the two clusters having the "smallest distance" will be merged when we merge the k_0 clusters into $k_0 - 1$ clusters. Finally, we show in Subsection 5.8.5 that under certain conditions, the mean vector and covariance matrix of the noisy variables do not change when we split the k_0 clusters into $k_0 + 1$ clusters or when we merge the k_0 clusters into $k_0 - 1$ clusters.

We expect that these results might hold for non-normal mixtures, but the mathematics for the general case is not tractable.

5.8.1 Mean Vectors and Covariance Matrices of Truncated Multivariate Normal Distributions

In this subsection, we obtain the explicit formula of the mean vectors and covariance matrices of the two parts of a multivariate normal distribution $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ which is truncated by a separating hyperplane

$$\boldsymbol{a}^T (\boldsymbol{x} - \boldsymbol{b}) = 0.$$

We first derive the density functions and then the moment generating functions. Finally we derive the formulas for the mean vectors and covariance matrices from the moment generating functions.

(1) Density Functions

The density functions of the two truncated multivariate normal distributions are

$$f_1(\boldsymbol{x}) = \begin{cases} \frac{1}{c_1}(2\pi)^{-p/2}\left|\boldsymbol{\Sigma}\right|^{-1/2}\exp\left[-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right] & \boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b}) \geq 0, \\ 0 & \boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b}) < 0, \end{cases}$$

and

$$f_2(\boldsymbol{x}) = \begin{cases} \frac{1}{c_2}(2\pi)^{-p/2}\left|\boldsymbol{\Sigma}\right|^{-1/2}\exp\left[-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right] & \boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b}) \leq 0, \\ 0 & \boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b}) > 0, \end{cases}$$

where

$$\begin{aligned} c_1 &= \int_{\boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b})\geq 0}(2\pi)^{-p/2}\left|\boldsymbol{\Sigma}\right|^{-1/2}\exp\left[-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right]d\boldsymbol{x} \\ &= P(\boldsymbol{a}^T(\boldsymbol{Y}-\boldsymbol{b}) \geq 0), \quad Y \sim \mathrm{N}\left(\boldsymbol{\mu},\ \boldsymbol{\Sigma}\right), \\ &= 1-\Phi\left[-\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}}\right] \\ &= \Phi\left[\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}}\right], \end{aligned}$$

and similarly

$$\begin{aligned} c_2 &= 1-\Phi\left[\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}}\right] \\ &= 1-c_1. \end{aligned}$$

(2) Moment Generating Functions

Suppose random vectors $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ have density functions $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ respectively. The moment generating function of $\boldsymbol{X}_1$ is

$$\begin{aligned} g_1(\boldsymbol{t}) &= \mathrm{E}\left(e^{\boldsymbol{t}^T\boldsymbol{X}_1}\right) \\ &= \frac{1}{c_1}\int_{\boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b})\geq 0}\exp\left(\boldsymbol{t}^T\boldsymbol{x}\right)(2\pi)^{-p/2}\left|\boldsymbol{\Sigma}\right|^{-1/2}\exp\left[-\frac{\left(\boldsymbol{x}-\boldsymbol{\mu}\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)}{2}\right]d\boldsymbol{x}. \end{aligned}$$

Note that

$$\begin{aligned}
\delta &= (\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu}) - 2\boldsymbol{t}^T\boldsymbol{x} \\
&= \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}+\boldsymbol{\Sigma t}) + \boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\
&= \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - 2\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}+\boldsymbol{\Sigma t}) + (\boldsymbol{\mu}+\boldsymbol{\Sigma t})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}+\boldsymbol{\Sigma t}) \\
&\quad - (\boldsymbol{\mu}+\boldsymbol{\Sigma t})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}+\boldsymbol{\Sigma t}) + \boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\
&= [\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})]^T\boldsymbol{\Sigma}^{-1}[\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})] - (\boldsymbol{\mu}+\boldsymbol{\Sigma t})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}+\boldsymbol{\Sigma t}) + \boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\
&= [\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})]^T\boldsymbol{\Sigma}^{-1}[\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})] - 2\left[\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right].
\end{aligned}$$

Thus, we can get

$$\begin{aligned}
g_1(\boldsymbol{t}) &= \frac{1}{c_1}\exp\left(\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right)\int_{\boldsymbol{a}^T(\boldsymbol{x}-\boldsymbol{b})\geq 0}(2\pi)^{-p/2}|\boldsymbol{\Sigma}|^{-1/2} \\
&\quad \cdot\exp\left[-\frac{[\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})]^T\boldsymbol{\Sigma}^{-1}[\boldsymbol{x}-(\boldsymbol{\mu}+\boldsymbol{\Sigma t})]}{2}\right]d\boldsymbol{x} \\
&= \frac{1}{c_1}\exp\left(\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right)P(\boldsymbol{a}^T(Z-\boldsymbol{b})\geq 0), \quad Z\sim \mathrm{N}(\boldsymbol{\mu}+\boldsymbol{\Sigma t},\ \boldsymbol{\Sigma}) \\
&= \frac{1}{c_1}\exp\left(\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right)\left[1-\Phi\left(-\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b}+\boldsymbol{\Sigma t})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right)\right] \\
&= \frac{1}{c_1}\exp\left(\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right)\Phi\left(\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b}+\boldsymbol{\Sigma t})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right).
\end{aligned}$$

Similarly, we can get the moment generating function of $\boldsymbol{X}_2$:

$$g_2(\boldsymbol{t}) = \frac{1}{c_2}\exp\left(\boldsymbol{\mu}^T\boldsymbol{t} + \frac{\boldsymbol{t}^T\boldsymbol{\Sigma t}}{2}\right)\left[1-\Phi\left(\frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b}+\boldsymbol{\Sigma t})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right)\right].$$

(3) Mean Vectors and Covariance Matrices

To get the mean vectors $\mathrm{E}(\boldsymbol{X}_i)$ and covariance matrices $\mathrm{Cov}(\boldsymbol{X}_i)$, $i = 1, 2$, we introduce some notation in Table 5.4.

Hence $g_1(\boldsymbol{t})$ can be rewritten as

$$g_1(\boldsymbol{t}) = \frac{1}{k_2(0)}k_1(\boldsymbol{t})k_2(\boldsymbol{t}).$$

Table 5.4: Notations I

$k_1(t) = \exp\left(\mu^T t + \frac{t^T \Sigma t}{2}\right)$	$k_1(0) = 1$
$k_2(t) = \Phi\left(\frac{a^T(\mu - b + \Sigma t)}{\sqrt{a^T \Sigma a}}\right)$	$k_2(0) = \Phi\left(\frac{a^T(\mu - b)}{\sqrt{a^T \Sigma a}}\right)$
$k_3(t) = \phi\left(\frac{a^T(\mu - b + \Sigma t)}{\sqrt{a^T \Sigma a}}\right)$	$k_3(0) = \phi\left(\frac{a^T(\mu - b)}{\sqrt{a^T \Sigma a}}\right)$
$k_4(t) = \frac{a^T(\mu - b + \Sigma t)}{\sqrt{a^T \Sigma a}}$	$k_4(0) = \frac{a^T(\mu - b)}{\sqrt{a^T \Sigma a}}$
$k_1'(t) = k_1(t)(\mu + \Sigma t)$	$k_1'(0) = \mu$
$k_2'(t) = k_3(t)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}$	$k_2'(0) = k_3(0)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}$
$k_3'(t) = -k_4(t)k_3(t)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}$	$k_3'(0) = -k_4(0)k_3(0)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}$
$k_4'(t) = \frac{\Sigma a}{\sqrt{a^T \Sigma a}}$	$k_4'(0) = \frac{\Sigma a}{\sqrt{a^T \Sigma a}}$

The first derivative vector of $g_1(t)$ is

$$\begin{aligned} g_1'(t) &= \frac{1}{k_2(0)}\left[k_1'(t)k_2(t) + k_1(t)k_2'(t)\right] \\ &= \frac{1}{k_2(0)}\left[k_1(t)k_2(t)\left(\mu + \Sigma t\right) + k_1(t)k_3(t)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}\right] \\ &= \frac{k_1(t)}{k_2(0)}\left[k_2(t)\left(\mu + \Sigma t\right) + k_3(t)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}\right]. \end{aligned}$$

and the second derivative vector of $g_1(t)$ is

$$\begin{aligned} g_1''(t) &= \frac{k_1'(t)}{k_2(0)}\left[k_2(t)\left(\mu + \Sigma t\right) + k_3(t)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}\right]^T \\ &\quad + \frac{k_1(t)}{k_2(0)}\left[k_2'(t)\left(\mu + \Sigma t\right)^T + k_2(t)\Sigma + k_3'(t)\frac{a^T \Sigma}{\sqrt{a^T \Sigma a}}\right]. \end{aligned}$$

Thus

$$\begin{aligned} \mathrm{E}\left(\boldsymbol{X}_1\right) &= g_1'(0) = \frac{k_1(0)}{k_2(0)}\left[k_2(0)\left(\mu + \Sigma 0\right) + k_3(0)\frac{\Sigma a}{\sqrt{a^T \Sigma a}}\right] \\ &= \mu + \frac{k_3(0)}{k_2(0)}\frac{\Sigma a}{\sqrt{a^T \Sigma a}}. \end{aligned}$$

and

$$\begin{aligned}
\mathrm{E}\left(\boldsymbol{X}_1\boldsymbol{X}_1^T\right) &= g_1''(\mathbf{0}) \\
&= \frac{k_1'(\mathbf{0})}{k_2(\mathbf{0})}\left[k_2(\mathbf{0})\left(\mu+\boldsymbol{\Sigma}\mathbf{0}\right)+k_3(\mathbf{0})\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right]^T \\
&\quad +\frac{k_1(\mathbf{0})}{k_2(\mathbf{0})}\left[k_2'(\mathbf{0})\left(\mu+\boldsymbol{\Sigma}\mathbf{0}\right)^T+k_2(\mathbf{0})\boldsymbol{\Sigma}+k_3'(\mathbf{0})\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right] \\
&= \boldsymbol{\mu\mu}^T+\boldsymbol{\Sigma}+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\mu a}^T\boldsymbol{\Sigma}+\boldsymbol{\Sigma a\mu}^T}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}-\frac{k_4(\mathbf{0})k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}} \\
&= \boldsymbol{\mu\mu}^T+\boldsymbol{\Sigma}+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\mu a}^T\boldsymbol{\Sigma}+\boldsymbol{\Sigma a\mu}^T}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}-\frac{k_4(\mathbf{0})k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}}.
\end{aligned}$$

Note that

$$\begin{aligned}
\left(\mathrm{E}\left(\boldsymbol{X}_1\right)\right)\left(\mathrm{E}\left(\boldsymbol{X}_1\right)\right)^T &= \left[\mu+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right]\left[\mu+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right]^T \\
&= \boldsymbol{\mu\mu}^T+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\mu a}^T\boldsymbol{\Sigma}+\boldsymbol{\Sigma a\mu}^T}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}+\frac{k_3^2(\mathbf{0})}{k_2^2(\mathbf{0})}\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}}.
\end{aligned}$$

We can get

$$\begin{aligned}
\mathrm{Cov}\left(\boldsymbol{X}_1\right) &= \mathrm{E}\left(\boldsymbol{X}_1\boldsymbol{X}_1^T\right)-\left(\mathrm{E}\left(\boldsymbol{X}_1\right)\right)\left(\mathrm{E}\left(\boldsymbol{X}_1\right)\right)^T \\
&= \boldsymbol{\mu\mu}^T+\boldsymbol{\Sigma}+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\mu a}^T\boldsymbol{\Sigma}+\boldsymbol{\Sigma a\mu}^T}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}-\frac{k_4(\mathbf{0})k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}} \\
&\quad -\boldsymbol{\mu\mu}^T-\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\mu a}^T\boldsymbol{\Sigma}+\boldsymbol{\Sigma a\mu}^T}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}-\frac{k_3^2(\mathbf{0})}{k_2^2(\mathbf{0})}\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}} \\
&= \boldsymbol{\Sigma}-\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\left[k_4(\mathbf{0})+\frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\right]\frac{\boldsymbol{\Sigma aa}^T\boldsymbol{\Sigma}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}.
\end{aligned}$$

Similarly, we can rewrite the moment generating function of X_2 as

$$g_2(t)=\frac{1}{1-k_2(0)}k_1(t)\left[1-k_2(t)\right].$$

Then

$$\begin{aligned}
g_2'(t) &= \frac{1}{1-k_2(\mathbf{0})}\left\{k_1'(t)\left[1-k_2(t)\right]+k_1(t)(-1)k_2'(t)\right\} \\
&= \frac{1}{1-k_2(\mathbf{0})}\left\{k_1(t)\left[1-k_2(t)\right]\left(\mu+\boldsymbol{\Sigma}t\right)-k_1(t)k_3(t)\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right\} \\
&= \frac{k_1(t)}{1-k_2(\mathbf{0})}\left\{\left[1-k_2(t)\right]\left(\mu+\boldsymbol{\Sigma}t\right)-k_3(t)\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}\right\}.
\end{aligned}$$

and

$$\begin{aligned} g_2''(t) &= \frac{k_1'(t)}{1-k_2(0)}\left\{[1-k_2(t)]\,(\mu+\Sigma t)-k_3(t)\frac{\Sigma a}{\sqrt{a^T\Sigma a}}\right\}^T \\ &\quad +\frac{k_1(t)}{1-k_2(0)}\left\{-k_2'(t)\,(\mu+\Sigma t)^T+[1-k_2(t)]\,\Sigma-k_3'(t)\frac{a^T\Sigma}{\sqrt{a^T\Sigma a}}\right\}. \end{aligned}$$

Thus

$$\begin{aligned} \mathrm{E}\,(\boldsymbol{X}_2) &= g_2'(0) \\ &= \mu-\frac{k_3(0)}{1-k_2(0)}\frac{\Sigma a}{\sqrt{a^T\Sigma a}}. \end{aligned}$$

and

$$\begin{aligned} \mathrm{E}\left(\boldsymbol{X}_2\boldsymbol{X}_2^T\right) &= g_2''(0) \\ &= \frac{k_1'(0)}{1-k_2(0)}\left\{[1-k_2(0)]\,(\mu+\Sigma 0)-k_3(0)\frac{\Sigma a}{\sqrt{a^T\Sigma a}}\right\}^T \\ &\quad +\frac{k_1(0)}{1-k_2(0)}\left\{-k_2'(0)\,(\mu+\Sigma 0)^T+[1-k_2(0)]\,\Sigma-k_3'(0)\frac{a^T\Sigma}{\sqrt{a^T\Sigma a}}\right\} \\ &= \mu\mu^T+\Sigma-\frac{k_3(0)}{1-k_2(0)}\frac{\mu a^T\Sigma+\Sigma a\mu^T}{\sqrt{a^T\Sigma a}}+\frac{k_4(0)k_3(0)}{1-k_2(0)}\frac{\Sigma a a^T\Sigma}{\sqrt{a^T\Sigma a}}. \end{aligned}$$

Note that

$$\begin{aligned} (\mathrm{E}\,(\boldsymbol{X}_2))\,(\mathrm{E}\,(\boldsymbol{X}_2))^T &= \left[\frac{k_1(0)}{1-k_2(0)}\left\{[1-k_2(0)]\,\mu-k_3(0)\frac{\Sigma a}{\sqrt{a^T\Sigma a}}\right\}\right] \\ &\quad \left[\frac{k_1(0)}{1-k_2(0)}\left\{[1-k_2(0)]\,\mu-k_3(0)\frac{\Sigma a}{\sqrt{a^T\Sigma a}}\right\}\right]^T \\ &= \mu\mu^T-\frac{k_3(0)}{1-k_2(0)}\frac{\mu a^T\Sigma\Sigma a\mu^T}{\sqrt{a^T\Sigma a}}+\frac{k_3^2(0)}{[1-k_2(0)]^2}\frac{\Sigma a a^T\Sigma}{\sqrt{a^T\Sigma a}}. \end{aligned}$$

We can get

$$\begin{aligned} \mathrm{Cov}\,(\boldsymbol{X}_2) &= \mathrm{E}\left(\boldsymbol{X}_2\boldsymbol{X}_2^T\right)-(\mathrm{E}\,(\boldsymbol{X}_2))\,(\mathrm{E}\,(\boldsymbol{X}_2))^T \\ &= \mu\mu^T+\Sigma-\frac{k_3(0)}{1-k_2(0)}\frac{\mu a^T\Sigma+\Sigma a\mu^T}{\sqrt{a^T\Sigma a}}+\frac{k_4(0)k_3(0)}{1-k_2(0)}\frac{\Sigma a a^T\Sigma}{\sqrt{a^T\Sigma a}} \\ &\quad -\mu\mu^T+\frac{k_3(0)}{1-k_2(0)}\frac{\mu a^T\Sigma+\Sigma a\mu^T}{\sqrt{a^T\Sigma a}}-\frac{k_3^2(0)}{[1-k_2(0)]^2}\frac{\Sigma a a^T\Sigma}{\sqrt{a^T\Sigma a}} \\ &= \Sigma+\frac{k_3(0)}{1-k_2(0)}\left[k_4(0-\frac{k_3(0)}{1-k_2(0)}\right]\frac{\Sigma a a^T\Sigma}{\sqrt{a^T\Sigma a}}. \end{aligned}$$

In summary,

$$\begin{aligned}
\mathrm{E}(\boldsymbol{X}_1) &= \boldsymbol{\mu} + \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}, \\
\mathrm{Cov}(\boldsymbol{X}_1) &= \boldsymbol{\Sigma} - \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\left[k_4(\mathbf{0}) + \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\right]\frac{\boldsymbol{\Sigma a a}^T\boldsymbol{\Sigma}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}, \\
\mathrm{E}(\boldsymbol{X}_2) &= \boldsymbol{\mu} - \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\frac{\boldsymbol{\Sigma a}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}}, \\
\mathrm{Cov}(\boldsymbol{X}_2) &= \boldsymbol{\Sigma} + \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\left[k_4(\mathbf{0} - \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\right]\frac{\boldsymbol{\Sigma a a}^T\boldsymbol{\Sigma}}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma a}}},
\end{aligned} \tag{5.8.1}$$

The above results can also be derived from the results of Tallis (1965).

5.8.2 Optimal Separating Hyperplane

There are infinitely many separating hyperplanes to truncate a multivariate normal distribution $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We want to find a separating hyperplane to minimize the mean square distance of a random point $\boldsymbol{X}$ to the mean vector

$$\begin{aligned}
\mathrm{MSD}(\boldsymbol{a}, \boldsymbol{b}) &= E\left[(\boldsymbol{X} - \boldsymbol{\mu}(C))^T(\boldsymbol{X} - \boldsymbol{\mu}(C))\right] \\
&= E\left[(\boldsymbol{X} - \boldsymbol{\mu}_1)^T(\boldsymbol{X} - \boldsymbol{\mu}_1)|\boldsymbol{a}^T(\boldsymbol{X} - \boldsymbol{b}) \geq 0\right] P(\boldsymbol{a}^T(\boldsymbol{X} - \boldsymbol{b}) \geq 0) \\
&+ E\left[(\boldsymbol{X} - \boldsymbol{\mu}_2)^T(\boldsymbol{X} - \boldsymbol{\mu}_2)|\boldsymbol{a}^T(\boldsymbol{X} - \boldsymbol{b}) < 0\right] P(\boldsymbol{a}^T(\boldsymbol{X} - \boldsymbol{b}) < 0) \\
&= c_1 \,\mathrm{tr}\,[\mathrm{Cov}(X_1)] + c_2 \,\mathrm{tr}\,[\mathrm{Cov}(X_2)] \\
&= k_2(\mathbf{0})\left\{\mathrm{tr}(\boldsymbol{\Sigma}) - \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\left[k_4(\mathbf{0}) + \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})}\right]\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}}\right\} \\
&+ [1 - k_2(\mathbf{0})]\left\{\mathrm{tr}(\boldsymbol{\Sigma}) + \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\left[k_4(\mathbf{0} - \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\right]\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}}\right\} \\
&= \mathrm{tr}(\boldsymbol{\Sigma}) - k_3(\mathbf{0})\left\{k_4(\mathbf{0} + \frac{k_3(\mathbf{0})}{k_2(\mathbf{0})} - k_4(\mathbf{0}) + \frac{k_3(\mathbf{0})}{1-k_2(\mathbf{0})}\right\}\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}} \\
&= \mathrm{tr}(\boldsymbol{\Sigma}) - \frac{k_3^2(\mathbf{0})}{k_2(\mathbf{0})[1-k_2(\mathbf{0})]}\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma a}}.
\end{aligned}$$

where C is a random variable indicating which side of the separating hyperplane $\boldsymbol{a}^T(\boldsymbol{x} - \boldsymbol{b}) = 0$ the random point $\boldsymbol{X}$ belongs to.

Theorem 5.8.1

$$\min_{\boldsymbol{a},\boldsymbol{b}} \mathrm{MSD}(\boldsymbol{a}, \boldsymbol{b}) = \mathrm{MSD}(\boldsymbol{\alpha}_1, \boldsymbol{\mu}) = \mathrm{tr}(\boldsymbol{\Sigma}) - 4\lambda_1\phi^2(0),$$

where λ_1 *and* $\boldsymbol{\alpha}_1$ *are the largest eigenvalue and corresponding eigenvector of the covariance matrix* $\boldsymbol{\Sigma}$.

Table 5.5: Notation II

$m_1(\theta) = \frac{1}{\Phi(\theta)[1-\Phi(\theta)]}$	$m_1(0) = 4$
$m_2(\theta) = \frac{1}{1-\Phi(\theta)} - \frac{1}{\Phi(\theta)}$	$m_2(0) = 0$
$m_3(\theta) = \frac{1}{[1-\Phi(\theta)]^2} + \frac{1}{[\Phi(\theta)]^2}$	$m_3(0) = 8$
$m_1'(\theta) = \phi(\theta)m_1(\theta)m_2(\theta)$	$m_1'(0) = 0$
$m_2'(\theta) = \phi(\theta)m_3(\theta)$	$m_2'(0) = 8\phi(0)$

[**Proof**] We first show that given the direction $\boldsymbol{a}$, the separation hyperplane should pass through the mean vector $\boldsymbol{\mu}$ and then show that the optimal direction is $\boldsymbol{\alpha}_1$. If the multiplicity of λ_1 is greater than 1, then any eigen direction of λ_1 will minimize MSD($\boldsymbol{a}$, $\boldsymbol{\mu}$).

Denote

$$h(\theta) = \mathrm{tr}\,(\boldsymbol{\Sigma}) - \eta \frac{\phi^2(\theta)}{\Phi(\theta)\,[1-\Phi(\theta)]},$$

where

$$\theta = \frac{\boldsymbol{a}^T(\boldsymbol{\mu}-\boldsymbol{b})}{\sqrt{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}},\ \eta = \frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}.$$

We introduce another set of notation in Table 5.5. Then to find the minimum point $\boldsymbol{b}^*$ of the function MSD $(\boldsymbol{a}, \boldsymbol{b})$ given $\boldsymbol{a}$ is equivalent to find the minimum point θ^* of the function $h(\theta)$.

The first derivative of the function $h(\theta)$ is

$$\begin{aligned} h'(\theta) &= -2\eta\phi(\theta)\phi'(\theta)m_1(\theta) - \eta\phi^2(\theta)m_1'(\theta) \\ &= 2\eta\theta\phi^2(\theta)m_1(\theta) - \eta\phi^2(\theta)\phi(\theta)m_1(\theta)m_2(\theta) \\ &= \eta\phi^2(\theta)m_1(\theta)\,[2\theta - \phi(\theta)m_2(\theta)] \end{aligned}$$

Let $h'(\theta) = 0$, we can get

$$\theta = \frac{1}{2}\phi(\theta)m_2(\theta).$$

It is not difficult to see that $\theta = 0$ is a solution of $h'(0) = 0$.

To show that $\theta = 0$ is the unique global minimum point of the function $h(\theta)$, we can show that $h'(\theta) > 0$ for $\theta > 0$ and that $h'(\theta) < 0$ for $\theta < 0$. By the fact that $m_1(-\theta) = m_1(\theta)$ and $m_2(-\theta) = -m_2(\theta)$, we can get that $h'(-\theta) = -h'(\theta)$. Hence we only need to show that $h'(\theta) > 0$

for $\theta > 0$.

$$\begin{aligned} h'(\theta) > 0 \, \forall \theta > 0 &\Leftrightarrow 2\theta > \phi(\theta) m_2(\theta) \quad \forall \theta > 0 \\ &\Leftrightarrow 2\theta > \phi(\theta) \frac{2\Phi(\theta) - 1}{\Phi(\theta)[1 - \Phi(\theta)]} \quad \forall \theta > 0 \\ &\Leftrightarrow 2\theta\Phi(\theta)[1 - \Phi(\theta)] > \phi(\theta)[2\Phi(\theta) - 1] \quad \forall \theta > 0 \end{aligned}$$

By the facts

$$1 - \Phi(\theta) = \int_{\theta}^{\infty} \phi(x) \mathrm{d}x > \phi(\theta),$$

we can get

$$2\theta\Phi(\theta)[1 - \Phi(\theta)] > 2\theta\Phi(\theta)\phi(\theta).$$

We want to show that

$$2\theta\Phi(\theta)\phi(\theta) > \phi(\theta)[2\Phi(\theta) - 1]$$

That is

$$1 - 2\Phi(\theta)[1 - \theta] > 0.$$

Let

$$g(\theta) = 1 - 2\Phi(\theta)[1 - \theta].$$

The first derivative of $g(\theta)$ is

$$\begin{aligned} g'(\theta) &= -2[\phi(\theta) - \Phi(\theta)] + 2\theta\phi(\theta) \\ &> 2\theta\phi(\theta) > 0, \ \forall \theta > 0. \end{aligned}$$

The second last step is due to the fact that $\Phi(\theta) > \phi(\theta)$ for $\theta > 0$. So $g(\theta)$ is a monotone increasing function of θ for all $\theta > 0$. $g(0) = 0$ implies that $g(\theta) > 0$ for all $\theta > 0$. Hence $h'(\theta) > 0$ for all $\theta > 0$ and $h'(\theta) < 0$ for all $\theta < 0$. Therefore

$$h(0) = \arg \min_{\theta} h(\theta).$$

$\theta = 0$ is equivalent to $\boldsymbol{a}^T(\boldsymbol{\mu} - \boldsymbol{b}) = 0$ for any given $\boldsymbol{a}$. That is,

$$\boldsymbol{b}^* = \boldsymbol{\mu}.$$

So given $\boldsymbol{a}$, the separating hyperplane passes through the mean vector $\boldsymbol{\mu}$. We can get

$$\text{MSD}\,(\boldsymbol{a}, \boldsymbol{\mu}) = \text{tr}\,(\boldsymbol{\Sigma}) - 4\phi^2(0)\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}. \tag{5.8.2}$$

Now given $\boldsymbol{b} = \boldsymbol{\mu}$, we want to find a normalized direction $\boldsymbol{a}^*$ such that $[\boldsymbol{a}^*]^T\boldsymbol{a}^* = 1$ and

$$\text{MSD}\,(\boldsymbol{a}^*, \boldsymbol{\mu}) = \min_{\boldsymbol{a}} \text{MSD}\,(\boldsymbol{a}, \boldsymbol{\mu}).$$

From formula (5.8.2), we know that to minimize $\text{MSD}\,(\boldsymbol{a}, \boldsymbol{\mu})$ is equivalent to maximize

$$\frac{\boldsymbol{a}^T\boldsymbol{\Sigma}^2\boldsymbol{a}}{\boldsymbol{a}^T\boldsymbol{\Sigma}\boldsymbol{a}}.$$

The solutions are $\boldsymbol{\alpha}_1$ and $-\boldsymbol{\alpha}_1$, where $\boldsymbol{\alpha}_1$ is the normalized eigenvector of the matrix $\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}^2 = \boldsymbol{\Sigma}$ corresponding to the maximum eigenvalue λ_1 (see Gnanadesikan 1977, Section 4.2). If we require the first element of the maximum point be positive and if the eigenspace of λ_1 has rank 1 (multiplicity of λ_1 is 1), then the solution is unique.

Now we show that $(\boldsymbol{\alpha}_1, \boldsymbol{\mu})$ is the global minimum point of the function $\text{MSD}\,(\boldsymbol{a}, \boldsymbol{b})$. Suppose that $(\boldsymbol{a}^*, \boldsymbol{b}^*)$ is the global minimum point. Then $\text{MSD}\,(\boldsymbol{a}^*, \boldsymbol{b}^*) < \text{MSD}\,(\boldsymbol{\alpha}_1, \boldsymbol{\mu})$. From the previous subsection and this subsection, we know that $\text{MSD}\,(\boldsymbol{a}^*, \boldsymbol{b}^*) \geq \text{MSD}\,(\boldsymbol{a}^*, \boldsymbol{\mu}) \geq \text{MSD}\,(\boldsymbol{\alpha}_1, \boldsymbol{\mu})$ leading to a contradiction. By the uniqueness of the solution, $(\boldsymbol{a}^*, \boldsymbol{b}^*) = (\boldsymbol{\alpha}_1, \boldsymbol{\mu})$, and $\min_{\boldsymbol{a},\boldsymbol{b}} \text{MSD}\,(\boldsymbol{a}, \boldsymbol{b}) = \text{tr}\,(\boldsymbol{\Sigma}) - 4\lambda_1\phi^2(0)$. □

Thus, if we split a multivariate normal distribution to two parts via a separating hyperplane, then the optimal value of MSD will decrease and the amount of decrease is $4\lambda_1\phi^2(0)$.

Corollary 5.8.2 *The mean vectors and covariance matrices of the two truncated distributions f_1 and f_2, which are truncated by the optimal separating hyperplane $\boldsymbol{\alpha}_1^T(\boldsymbol{x} - \boldsymbol{\mu}) = 0$, are*

$$\begin{aligned} \text{E}\,(\boldsymbol{X}_1) &= \boldsymbol{\mu} + 2\sqrt{\lambda_1}\phi(0)\boldsymbol{\alpha}_1, \quad \text{Cov}\,(\boldsymbol{X}_1) = \boldsymbol{\Sigma} - 4\lambda_1\phi^2(0)\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^T, \\ \text{E}\,(\boldsymbol{X}_2) &= \boldsymbol{\mu} - 2\sqrt{\lambda_1}\phi(0)\boldsymbol{\alpha}_1, \quad \text{Cov}\,(\boldsymbol{X}_2) = \boldsymbol{\Sigma} - 4\lambda_1\phi^2(0)\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^T. \end{aligned} \tag{5.8.3}$$

5.8.3 Which Cluster is Chosen to Split?

If we assume that when splitting k_0 clusters into $k_0 + 1$ clusters, one of k_0 cluster will be split, then by intuition the cluster which has the largest size will be split if we want to make sure the within-cluster sum of squares reach minimum after splitting. We can prove this theoretically.

Theorem 5.8.3 *Suppose that when splitting k_0 clusters into k_0+1 clusters, one of k_0 clusters will be split by a hyperplane. Without loss of generality, suppose that the k_0-th cluster will be split. Suppose that the separating hyperplane is the optimal separating hyperplane which minimizes the mean squared distance (MSD) of the k_0-th component density. Then the internal cohesion and external isolation (c.f. Formula (5.6.2)) before and after the splitting have the following relations:*

$$\boldsymbol{A}^* = \boldsymbol{A} - \boldsymbol{B}_2, \quad \boldsymbol{B}^* = \boldsymbol{B} + \boldsymbol{B}_2, \quad \boldsymbol{A}^* + \boldsymbol{B}^* = \boldsymbol{A} + \boldsymbol{B}, \tag{5.8.4}$$

where $\boldsymbol{B}_2 = 4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T$.

[**Proof**]:

After splitting, the new internal cohesion is

$$\boldsymbol{A}^* = \sum_{i=1}^{k_0-1} \pi_i\boldsymbol{\Sigma}_i + \pi_{k_0,1}\boldsymbol{\Sigma}_{k_0,1} + \pi_{k_0,2}\boldsymbol{\Sigma}_{k_0,2},$$

where

$$\begin{aligned}\pi_{k_0,1} &= \pi_{k_0,2} = \frac{\pi_{k_0}}{2}\\ \boldsymbol{\Sigma}_{k_0,1} &= \boldsymbol{\Sigma}_{k_0,2} = \boldsymbol{\Sigma}_{k_0} - 4\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T,\end{aligned}$$

and λ_{k_0} is the maximum eigenvalue of the covariance matrix $\boldsymbol{\Sigma}_{k_0}$ and $\boldsymbol{\alpha}_{k_0}$ is the eigenvector corresponding to λ_{k_0}. $\pi_{k_0,1} = \pi_{k_0,2} = \pi_{k_0}/2$ is because that the optimal separating hyperplane passes through the center and that the multivariate normal distribution is symmetric about the center.

We can get

$$\begin{aligned}\boldsymbol{A}^* &= \sum_{i=1}^{k_0} \pi_i\boldsymbol{\Sigma}_i - 4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T\\ &= \boldsymbol{A} - 4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.\end{aligned}$$

By definition, the new external isolation is

$$\boldsymbol{B}^* = \sum_{i=1}^{k_0+1} \pi_i^* \left(\boldsymbol{\mu}_i^* - \boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_i^* - \boldsymbol{\mu}^*\right)^T.$$

Since the first $k_0 - 1$ clusters have not been changed,

$$\pi_i^* = \pi_i,\ \boldsymbol{\Sigma}_i^* = \boldsymbol{\Sigma}_i,\ \mu_i^* = \mu_i,\ i = 1, \ldots, k_0 - 1.$$

The optimal mean vectors and covariance matrices of the two parts of the k_0-th cluster are

$$\begin{aligned}
\boldsymbol{\mu}_{k_0,1} &= \mu_{k_0} + 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0}, \\
\boldsymbol{\mu}_{k_0,2} &= \mu_{k_0} - 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0}, \\
\boldsymbol{\Sigma}_{k_0,1} &= \boldsymbol{\Sigma}_{k_0,2} = \boldsymbol{\Sigma}_{k_0} - 4\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.
\end{aligned} \tag{5.8.5}$$

We can get

$$\begin{aligned}
\boldsymbol{\mu}^* &= \sum_{i=1}^{k_0+1} \pi_i^* \mu_i^* \\
&= \sum_{i=1}^{k_0-1} \pi_i \boldsymbol{\mu}_i + \pi_{k_0,1}^* \boldsymbol{\mu}_{k_0,1} + \pi_{k_0,2}^* \boldsymbol{\mu}_{k_0,2} \\
&= \sum_{i=1}^{k_0-1} \pi_i \boldsymbol{\mu}_i + \pi_{k_0} \boldsymbol{\mu}_{k_0} \\
&= \boldsymbol{\mu}.
\end{aligned} \tag{5.8.6}$$

Thus,

$$\boldsymbol{\mu}_i^* - \boldsymbol{\mu}^* = \boldsymbol{\mu}_i - \boldsymbol{\mu}, \quad i = 1, \ldots, k_0 - 1.$$

Also

$$\left(\boldsymbol{\mu}_i^* - \boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_i^* - \boldsymbol{\mu}^*\right)^T = \left(\boldsymbol{\mu}_i - \boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_i - \boldsymbol{\mu}\right)^T, \quad i = 1, \ldots, k_0 - 1,$$

and

$$\begin{aligned}
\boldsymbol{\mu}_{k_0,1} - \boldsymbol{\mu}^* &= \boldsymbol{\mu}_{k_0} + 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0} - \boldsymbol{\mu} \\
&= \left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right) + 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0}.
\end{aligned}$$

Thus

$$\begin{aligned}
\left(\boldsymbol{\mu}_{k_0,1} - \boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_{k_0,1} - \boldsymbol{\mu}^*\right)^T &= \left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right)^T \\
&\quad + 2\phi(0)\sqrt{\lambda_{k_0}}\left[\left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right)\boldsymbol{\alpha}_{k_0}^T + \boldsymbol{\alpha}_{k_0}\left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right)^T\right] \\
&\quad + 4\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.
\end{aligned}$$

Similarly,

$$\begin{aligned}
\boldsymbol{\mu}_{k_0,2} - \boldsymbol{\mu}^* &= \boldsymbol{\mu}_{k_0} - 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0} - \boldsymbol{\mu} \\
&= \left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}\right) - 2\phi(0)\sqrt{\lambda_{k_0}}\boldsymbol{\alpha}_{k_0},
\end{aligned}$$

and

$$\begin{aligned}\left(\boldsymbol{\mu}_{k_0,2}-\boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_{k_0,2}-\boldsymbol{\mu}^*\right)^T &= \left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)^T \\ &\quad -2\phi(0)\sqrt{\lambda_{k_0}}\left[\left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)\boldsymbol{\alpha}_{k_0}^T+\boldsymbol{\alpha}_{k_0}\left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)^T\right] \\ &\quad +4\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.\end{aligned}$$

Therefore

$$\begin{aligned}&\frac{\pi_{k_0}}{2}\left(\boldsymbol{\mu}_{k_0,1}-\boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_{k_0,1}-\boldsymbol{\mu}^*\right)^T+\frac{\pi_{k_0}}{2}\left(\boldsymbol{\mu}_{k_0,2}-\boldsymbol{\mu}^*\right)\left(\boldsymbol{\mu}_{k_0,2}-\boldsymbol{\mu}^*\right)^T \\ &\quad=\pi_{k_0}\left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_{k_0}-\boldsymbol{\mu}\right)^T+4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.\end{aligned}$$

Thus, the new external isolation is

$$\boldsymbol{B}^* = \sum_{i=1}^{k_0}\pi_i\left(\boldsymbol{\mu}_i-\boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_i-\boldsymbol{\mu}\right)^T+4\pi_{k_0}\phi^2(0)\lambda_\alpha\alpha_{k_0}\alpha_{k_0}^T,$$

and

$$\boldsymbol{B}^*=\boldsymbol{B}+4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T.$$

Denote

$$\boldsymbol{B}_2 = 4\pi_{k_0}\phi^2(0)\lambda_{k_0}\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T. \tag{5.8.7}$$

Then (5.8.4) obtains. □

Corollary 5.8.4 *Suppose that when splitting k_0 clusters into k_0+1 clusters, one of k_0 clusters will be split by a hyperplane. Suppose that the splitting (separating) hyperplane is the optimal separating hyperplane which minimizes the mean squared distance MSD of the k_0-th component density. Then to minimize the trace of the new internal cohesion A^*, the cluster whose $\pi_j\lambda_j$ is the largest should be chosen to be split, where λ_j is the largest eigenvalue of the covariance matrix of the j-th cluster and π_j is the proportion of the j-th cluster.*

5.8.4 Merging k_0 Clusters into k_0-1 Clusters

If we reduce by one cluster, we expect that the two clusters which are closest will be merged.

Theorem 5.8.5 *Suppose that when merging k_0 clusters into k_0-1 clusters, only two of k_0 clusters will be split. Without loss of generality, we suppose Cluster k_0-1 and Cluster k_0 will be merged. Then the internal cohesion and external isolation before and after merging have the following relations:*

$$\boldsymbol{A}^{**} = \boldsymbol{A} + \boldsymbol{B}_3, \quad \boldsymbol{B}^{**} = \boldsymbol{B} - \boldsymbol{B}_3, \quad \boldsymbol{A}^{**} + \boldsymbol{B}^{**} = \boldsymbol{A} + \boldsymbol{B}, \tag{5.8.8}$$

where $\boldsymbol{B}_3 = \pi_{k_0-1}\pi_{k_0}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T/(\pi_{k_0-1} + \pi_{k_0})$.

[**Proof**]:

The new internal cohesion and external isolation are:

$$\begin{aligned}
\boldsymbol{A}^{**} &= \sum_{i=1}^{k_0-1} \pi_i^{**}\boldsymbol{\Sigma}_i^{**}, \\
\boldsymbol{B}^{**} &= \sum_{i=1}^{k_0-1} \pi_i^{**}\left(\boldsymbol{\mu}_i^{**} - \boldsymbol{\mu}^{**}\right)\left(\boldsymbol{\mu}_i^{**} - \boldsymbol{\mu}^{**}\right)^T,
\end{aligned}$$

where

$$\pi_i^{**} = \pi_i,\ \boldsymbol{\mu}_i^{**} = \boldsymbol{\mu}_i,\ \boldsymbol{\Sigma}_i^{**} = \boldsymbol{\Sigma}_i,\ i = 1, \ldots, k_0 - 2;$$

$$\pi_{k_0-1}^{**} = \pi_{k_0-1} + \pi_{k_0};$$

$$\begin{aligned}
\boldsymbol{\mu}_{k_0-1}^{**} &= \mathrm{E}\left(X|X \in C_{k_0-1}\right)P\left(X \in C_{k_0-1}\right) + \mathrm{E}\left(X|X \in C_{k_0}\right)P\left(X \in C_{k_0}\right) \\
&= \mathrm{E}\left(X|X \in C_{k_0-1}\right)\frac{\pi_{k_0-1}}{\pi_{k_0-1} + \pi_{k_0}} + \mathrm{E}\left(X|X \in C_{k_0}\right)\frac{\pi_{k_0}}{\pi_{k_0-1} + \pi_{k_0}} \\
&= \frac{\pi_{k_0-1}}{\pi_{k_0-1} + \pi_{k_0}}\boldsymbol{\mu}_{k_0-1} + \frac{\pi_{k_0}}{\pi_{k_0-1} + \pi_{k_0}}\boldsymbol{\mu}_{k_0},
\end{aligned} \tag{5.8.9}$$

where C_k stands for the k-th component of the mixture. And

$$\begin{aligned}
\boldsymbol{\Sigma}_{k_0-1}^{**} &= \mathrm{E}\left[\left(X - \boldsymbol{\mu}_{k_0-1}^{**}\right)\left(X - \boldsymbol{\mu}_{k_0-1}^{**}\right)^T|X \in C_{k_0-1}\right]P\left(X \in C_{k_0-1}\right) \\
&\quad + \mathrm{E}\left[\left(X - \boldsymbol{\mu}_{k_0-1}^{**}\right)\left(X - \boldsymbol{\mu}_{k_0-1}^{**}\right)^T|X \in C_{k_0-1}\right]P\left(X \in C_{k_0}\right) \\
&= \frac{\pi_{k_0-1}}{\pi_{k_0-1} + \pi_{k_0}}\left[\boldsymbol{\Sigma}_{k_0-1} + \left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0-1}^{**}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0-1}^{**}\right)^T\right] \\
&\quad + \frac{\pi_{k_0}}{\pi_{k_0-1} + \pi_{k_0}}\left[\boldsymbol{\Sigma}_{k_0} + \left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}_{k_0-1}^{**}\right)\left(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}_{k_0-1}^{**}\right)^T\right].
\end{aligned}$$

We can get

$$\begin{aligned}\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}^{**}_{k_0-1} &= \frac{\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}),\\ \boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}^{**}_{k_0-1} &= \frac{\pi_{k_0-1}}{\pi_{k_0-1}+\pi_{k_0}}(\boldsymbol{\mu}_{k_0} - \boldsymbol{\mu}_{k_0-1}).\end{aligned}$$

Thus

$$\begin{aligned}\boldsymbol{\mu}^{**} &= \sum_{i=1}^{k_0-1} \pi_i^{**}\boldsymbol{\mu}_i^{**}\\ &= \sum_{i=1}^{k_0-2} \boldsymbol{\mu}_i + \pi^{**}_{k_0-1}\boldsymbol{\mu}^{**}_{k_0-1}\\ &= \boldsymbol{\mu}.\end{aligned} \tag{5.8.10}$$

and

$$\begin{aligned}\boldsymbol{\Sigma}^{**}_{k_0-1} = &\frac{\pi_{k_0-1}}{\pi_{k_0-1}+\pi_{k_0}}\boldsymbol{\Sigma}_{k_0-1} + \frac{\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\boldsymbol{\Sigma}_{k_0}\\ &+ \frac{\pi_{k_0-1}\pi_{k_0}}{(\pi_{k_0-1}+\pi_{k_0})^2}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T\end{aligned} \tag{5.8.11}$$

Therefore, we can get

$$\begin{aligned}\boldsymbol{A}^{**} &= \sum_{i=1}^{k_0} \pi_i\boldsymbol{\Sigma}_i + \frac{\pi_{k_0-1}\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T\\ &= \boldsymbol{A} + \frac{\pi_{k_0-1}\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T.\end{aligned}$$

Hence

$$\mathrm{tr}\left(\boldsymbol{A}^{**}\right) = \mathrm{tr}\left(\boldsymbol{A}\right) + \frac{\pi_{k_0-1}\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right).$$

By algebra, we can get

$$\boldsymbol{B}^{**} = \boldsymbol{B} - \frac{\pi_{k_0-1}\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T.$$

Denote

$$\boldsymbol{B}_3 = \frac{\pi_{k_0-1}\pi_{k_0}}{\pi_{k_0-1}+\pi_{k_0}}\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)\left(\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}\right)^T. \tag{5.8.12}$$

Then (5.8.8) obtains. □

Corollary 5.8.6 *Suppose that when merging k_0 clusters into $k_0 - 1$ clusters, only two of k_0 clusters will be split. Then, to minimize the trace of the new internal cohesion* $\mathrm{tr}\,(A^{**})$, *we have to merge the cluster pair k and k' such that*

$$\frac{\pi_k \pi_{k'}}{\pi_k + \pi_{k'}} (\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k'})^T (\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k'})$$

is the smallest. If $\pi_1 = \cdots = \pi_{k_0}$, then two clusters which have the smallest distance between means will be considered for merging.

5.8.5 Effects of the Specification of the Number of Clusters

In this subsection, we will show that under certain conditions, noisy variables are still noisy variables after splitting a cluster or merging two clusters. This provides a partial validation of our weight vector averaging technique.

Theorem 5.8.7 *Assume that the k_0 component densities of the mixture of distributions are normally distributed. Suppose that when merging k_0 clusters into $k_0 - 1$ clusters, only two of k_0 clusters will be merged. Also suppose that the criterion to choose the two clusters to be merged is to minimize the internal cohesion. Then after merging, the **CP** weights corresponding to the noisy variables are zero.*

[**Proof**]: Suppose that the two nearest clusters are clusters $k_0 - 1$ and k_0. From formulas (5.8.9) and (5.8.11), the mean vector and the covariance matrix of the new merged cluster $k_0 - 1$ respectively are

$$\boldsymbol{\mu}^{**}_{k_0-1} = \frac{\pi_{k_0-1}}{\pi_{k_0-1} + \pi_{k_0}} \boldsymbol{\mu}_{k_0-1} + \frac{\pi_{k_0}}{\pi_{k_0-1} + \pi_{k_0}} \boldsymbol{\mu}_{k_0},$$

and

$$\boldsymbol{\Sigma}^{**}_{k_0-1} = \frac{\pi_{k_0-1}}{\pi_{k_0-1} + \pi_{k_0}} \boldsymbol{\Sigma}_{k_0-1} + \frac{\pi_{k_0}}{\pi_{k_0-1} + \pi_{k_0}} \boldsymbol{\Sigma}_{k_0} + \frac{\pi_{k_0-1}\pi_{k_0}}{(\pi_{k_0-1} + \pi_{k_0})^2} (\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0}) (\boldsymbol{\mu}_{k_0-1} - \boldsymbol{\mu}_{k_0})^T,$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$, $i = k_0 - 1, k_0$, have the forms

$$\boldsymbol{\mu}_i = \begin{pmatrix} \eta_i \\ \boldsymbol{\beta} \end{pmatrix}, \quad \boldsymbol{\Sigma}_i = \begin{pmatrix} \boldsymbol{V}_i & 0 \\ 0 & \boldsymbol{V} \end{pmatrix}.$$

It is straightforward to check that for the new merged cluster $k_0 - 1$, the mean vector and the covariance matrix for noisy variables are still β and V respectively. If we assume noisy variables are normally distributed, no change of the mean vector and covariance matrix indicates no change of the distribution. Thus, the noisy variables are still noisy after the merging. □

Lemma 5.8.8 *Suppose that a positive definite matrix Σ has the form*

$$\Sigma = \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix},$$

where V_1 is a $p_1 \times p_1$ matrix and V_2 is a $p_2 \times p_2$ matrix.

1. *Suppose that λ is an eigenvalue of the matrix Σ. Then λ is an eigenvalue of either V_1 or V_2 or both.*

2. *Suppose that λ is an eigenvalue of V_1 but not V_2. Then its normalized orthogonal eigenvectors $\alpha_1, \ldots, \alpha_\ell$ for the matrix Σ have the form $\alpha_i = \left(\xi_i^T, 0^T\right)^T$, $i = 1, \ldots, \ell$, where $\ell \leq p_1$ is the multiplicity of the eigenvalue λ.*

3. *Suppose that λ^* is an eigenvalue of V_2 but not V_1. Then its normalized orthogonal eigenvectors $\beta_1, \ldots, \beta_t$ for the matrix Σ have the form $\beta_i = \left(0^T, \eta_i^T\right)$, $i = 1, \ldots, t$, where $t \leq p_2$ is the multiplicity of the eigenvalue λ^*.*

[**Proof**:] By definition, an eigenvalue λ of the matrix Σ satisfies the equation $\det(\lambda I_p - \Sigma) = 0$, where the function $\det(A)$ calculates the determinant of the square matrix A. Hence we get $\det(\lambda I_{p_1} - V_1) = 0$ or $\det(\lambda I_{p_2} - V_2) = 0$. That is, λ is an eigenvalue of either V_1 or V_2 or both. The first part of the proof is completed.

Now we consider the second part of the lemma. Suppose the $p_1 \times 1$ vectors ξ_i^*, $i = 1, \ldots, \ell$, are p_1 normalized orthogonal eigenvectors of λ for the matrix V_1. Then the $p \times 1$ vectors $\xi_i = \left((\xi_i^*)^T, 0^T\right)^T$ are normalized orthogonal eigenvectors of λ for the matrix Σ. And any eigenvector ξ of λ for the matrix Σ can be expressed by a linear combination of ξ_i, $i = 1, \ldots, \ell$. Decompose ξ as $\xi = (\xi_u^T, \xi_b^T)^T$, where ξ_u and ξ_b are $p_1 \times 1$ and $p_2 \times 2$ matrices respectively. Then there exist scalars $c_1, \ldots, c_\ell$ such that

$$\begin{pmatrix} \xi_u \\ \xi_b \end{pmatrix} = c_1 \begin{pmatrix} \xi_1^* \\ 0 \end{pmatrix} + \cdots + c_\ell \begin{pmatrix} \xi_\ell^* \\ 0 \end{pmatrix}.$$

This implies that $\boldsymbol{\xi}_b = \mathbf{0}$. The second part of the proof is completed. By symmetry, the third part of the lemma is proved. □

Theorem 5.8.9 *Assume that the k_0 component densities of the mixture of distributions are normally distributed. Suppose that when splitting k_0 clusters into $k_0 + 1$ clusters, one of k_0 clusters will be split by a hyperplane. Without loss of generality, suppose that the k_0-th cluster will be split. Suppose that the criterion to choose the cluster to be split is to minimize the internal cohesion and that the criterion to choose the separating hyperplane is to minimize MSD of the k_0-th component density. Also suppose that the eigenvalues of the covariance matrix corresponding to non-noisy variables are larger than those of the covariance matrix corresponding to noisy variables. Then after splitting, the **CP** weights corresponding to the noisy variables are zero.*

[**Proof**]: Without loss of generality, suppose that the cluster whose covariance matrix has the largest maximum-eigenvalue is cluster k_0. From formula (5.8.5), the mean vectors and covariance matrices for the two new split clusters are

$$\begin{aligned} \boldsymbol{\mu}_{k_0,1} &= \boldsymbol{\mu}_{k_0} + 2\sqrt{\lambda_{k_0}}\phi(0)\boldsymbol{\alpha}_{k_0}, \quad \boldsymbol{\Sigma}_{k_0,1} = \boldsymbol{\Sigma}_{k_0} - 4\lambda_{k_0}\phi^2(0)\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T, \\ \boldsymbol{\mu}_{k_0,2} &= \boldsymbol{\mu}_{k_0} - 2\sqrt{\lambda_{k_0}}\phi(0)\boldsymbol{\alpha}_{k_0}, \quad \boldsymbol{\Sigma}_{k_0,2} = \boldsymbol{\Sigma}_{k_0} - 4\lambda_{k_0}\phi^2(0)\boldsymbol{\alpha}_{k_0}\boldsymbol{\alpha}_{k_0}^T, \end{aligned} \tag{5.8.13}$$

where $\boldsymbol{\mu}_{k_0}$ and $\boldsymbol{\Sigma}_{k_0}$ are the mean vector and the covariance matrix of cluster k_0, λ_{k_0} and $\boldsymbol{\alpha}_{k_0}$ are the maximum eigenvalue and corresponding eigenvector of the matrix $\boldsymbol{\Sigma}_{k_0}$, $\boldsymbol{\mu}_{k_0,i}$ and $\boldsymbol{\Sigma}_{k_0,i}$, $i = 1, 2$ are mean vectors and covariance matrices of the new split clusters.

We want to show that under certain conditions, $\boldsymbol{\alpha}_{k_0}$ has the form $(\boldsymbol{\xi}_{k_0}^T, \mathbf{0}^T)$, in which case the mean vector and covariance matrix for noisy variables are unchanged.

Suppose that the first p_1 variables are non-noisy variables and the remaining p_2 variables are noisy variables. If the largest eigenvalue comes from the covariance matrix $\boldsymbol{V}_1$ of the non-noisy variables, then from Lemma 5.8.8, $\boldsymbol{\alpha}_{k_0}$ has the form $\boldsymbol{\alpha}_{k_0} = (\boldsymbol{\xi}_{k_0}^T, \mathbf{0}^T)$, where $\boldsymbol{\xi}_{k_0}$ is a $p_1 \times 1$ vector and $\mathbf{0}$ is $p_2 \times 1$ vector whose elements are all zero. Thus, we can see from (5.8.13) that the mean vectors and covariance matrices for noisy variables do not change after splitting. □

The assumption in Theorem 5.8.9 that the eigenvalues of the covariance matrix corresponding to non-noisy variables are larger than those of the covariance matrix corresponding to noisy variables may not be true.

To illustrate this, we generate a small data set consisting of 2 clusters in a 2-dimensional space from bivariate normal distributions $\mathrm{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2,$, each having 100 data points, where

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ \boldsymbol{\mu}_2 = \begin{pmatrix} 6 \\ 0 \end{pmatrix}, \ \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}.$$

The second variable is noisy. The scatter plot and a 3-cluster partition obtained by `MKmeans` clustering algorithm are shown in Figure 5.5. The right panel of Figure 5.5 shows that the distribution

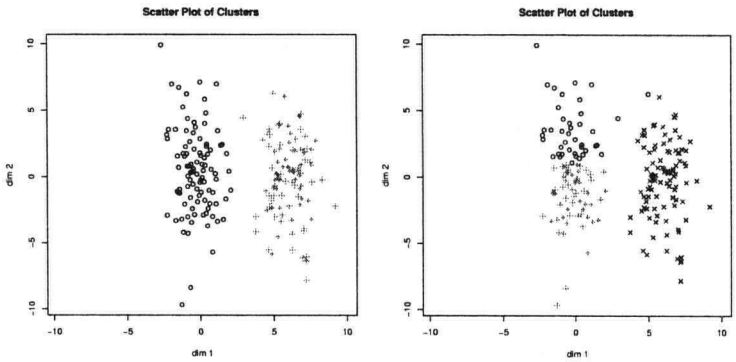


Figure 5.5: The left panel shows the scatter plot of the data set in Example 2. The right panel shows a 3-cluster partition obtained by `MKmeans` clustering algorithm.

of the noisy variable is no longer the same across clusters. The results for the simulated data set are listed in Table 5.6. Table 5.6 shows that the weights (based on the sample mean vectors $\hat{\boldsymbol{\mu}}_k$ and

Table 5.6: Weight vectors for the simulated data set

k_0	$\hat{w}_1^I$	$\hat{w}_2^I$	$\hat{w}_1^{II}$	$\hat{w}_2^{II}$
2	1.000	0.004	1.000	0.004
3	1.000	0.035	1.000	0.064
4	1.000	0.012	1.000	0.110
5	1.000	0.022	1.000	0.194
6	1.000	0.107	1.000	0.349
7	1.000	0.005	1.000	0.480
8	1.000	0.300	1.000	0.735
9	0.887	1.000	0.923	1.000
10	1.000	0.591	0.842	1.000
$\hat{\boldsymbol{w}}^{III}$	$(1.000, 0.234)^T$		$(1.000, 0.449)^T$	

covariance matrices $\hat{\boldsymbol{\Sigma}}$, $k = 1, \ldots, k_0$, of the partition obtained by `MKmeans`) for the noisy variable is no longer close to zero if the number of clusters is well over-specified.

We propose a possible improvement in the next section.

5.9 Improvement by Iteration

In previous section, we shows that the weights of noisy variables computed from method in Section 5.6 might be large if the largest eigenvalue of covariance matrices corresponds to the noisy variables and if the number of clusters is over-specified. The weight vector averaging technique could downweight the noisy variables in this case. However the weights for noisy variables are still not close to zero. To overcome this problem, we propose to iteratively use the weight vector averaging technique. More specifically, we propose the following algorithm:

Step 1 Set the initial weight vector as $\boldsymbol{w}_0 = \mathbf{1}_p$, where p is the number of variables and $\mathbf{1}_p$ is the $p \times 1$ vector whose elements are all equal to one. Set the iteration number ITMAX.

Step 2 Obtain a weight vector $\boldsymbol{w}$ by applying the weight vector averaging technique for the data set $\boldsymbol{Y}$ weighted by $\boldsymbol{w}_0$.

Step 3 $\boldsymbol{w}_0 \leftarrow \boldsymbol{w} * \boldsymbol{w}_0 / \max(\boldsymbol{w} * \boldsymbol{w}_0)$, where the symbol $*$ means point-wise multiplication. If the current iteration number is greater than ITMAX, then output $\boldsymbol{w}_0$ and stop. Otherwise go back to Step 2.

For the example in the previous subsection (Subsection 5.8.5), the weight vectors are

$$\hat{\boldsymbol{w}}^{I,III,2} = \begin{pmatrix} 1.000 \\ 0.018 \end{pmatrix}, \quad \text{and} \quad \hat{\boldsymbol{w}}^{II,III,2} = \begin{pmatrix} 1.000 \\ 0.089 \end{pmatrix}$$

respectively after 2 iterations. $\hat{\boldsymbol{w}}^{I,III,2}$ means that we iterate twice the Method I with weight vector averaging technique and $\hat{\boldsymbol{w}}^{I,III,2}$ means that we iterate twice the Method II with weight vector averaging technique. Now the weight for noisy variable is close to zero.

The weights of the noisy variables decrease with iteration. After some down-weighting, the covariance matrix of the noisy variables will not dominate that of the non-noisy variables.

5.10 Overall Algorithm

Combining the results in the previous sections, we propose the following compact projection weight vector averaging algorithm:

CP-WVA Algorithm

Step 1 Initialize the values of k_0^{low}, k_0^{upp}, k_0^{step}, $\epsilon_{\boldsymbol{w}}$, and *maxiter*, where k_0^{low}, k_0^{upp}, and k_0^{step} are the lower, upper bound and increment of the number of clusters, $\epsilon_{\boldsymbol{w}}$ is the threshold to decide if a variable is a noisy variable, and *maxiter* is the maximum allowable number of iterations. Input the $n \times p$ data matrix $\boldsymbol{Y}$ where n is the number of objects and p is the number of variables. Set *iter*$\leftarrow$0 and $\boldsymbol{w}_{\text{old}} \leftarrow \mathbf{1}_p$. Set the value of flag which indicates variable weighting or selection.

Step 2 Adjust the value of k_0^{low} and k_0^{upp} so that the minimum cluster size of any partition is greater than $p+1$.

Step 3 Weight the columns of the data matrix $\boldsymbol{Y}$ by the weight vector $\boldsymbol{w}_{\text{old}}$. For each $k_0 \in [k_0^{\text{low}}, k_0^{\text{upp}}]$ with step k_0^{step}, obtain a partition using a clustering method, and get the **CP** weight vector $\boldsymbol{w}^{(k_0)}$. Let the average **CP** weight vector be

$$\boldsymbol{w} = \frac{1}{g} \sum_{k_0 \in [k_0^{\text{low}}, k_0^{\text{upp}}] \text{ with step } k_0^{\text{step}}} \boldsymbol{w}^{(k_0)},$$

where g is the number of items in the summation. Set $\boldsymbol{w} \leftarrow \boldsymbol{w} * \boldsymbol{w}_{\text{old}} / \max(\boldsymbol{w} * \boldsymbol{w}_{\text{old}})$.

Step 4 *iter*$\leftarrow$*iter* $+ 1$. If *iter* $\leq$ *maxiter*, let $\boldsymbol{w}_{\text{old}} \leftarrow \boldsymbol{w}$ and go to Step 3. Otherwise go to Step 5.

Step 5 For variable selection, set $w_i = 0$ if $w_i \leq \epsilon_{\boldsymbol{w}}$ and $w_i = 1$ if $w_i > \epsilon_{\boldsymbol{w}}$.

Step 6 Output the final weight vector $\boldsymbol{w}$.

Note that different clustering methods and variable weighting methods can be substituted in Step 3. Variables might be standardized in some way (e.g. the standard deviations of all variables are equal to 1) before starting this algorithm.

In practice, researchers can use subject matter knowledge to determine a range of the number of clusters that should hopefully include the "true" number of clusters.

5.11 Examples

In this section, we use simulated data sets and real data sets to study the performances of the Algorithm 5.10 implemented with the `CP` method II.

To measure the performance of a variable selection method, we introduce type I error (ϵ_I) and type II error (ϵ_{II}). Denote

$$S_{0t} = \{i : \text{ variable } i \text{ is a true noisy variable}\},$$
$$S_{0o} = \{i : \text{ variable } i \text{ is an observed noisy variable}\},$$
$$S_{1t} = \{i : \text{ variable } i \text{ is a true non-noisy variable}\},$$
$$S_{1o} = \{i : \text{ variable } i \text{ is an observed non-noisy variable}\}.$$

The type I error and type II error are defined as

$$\epsilon_I = \frac{|S_{0t} \cap S_{1o}|}{|S_{0t}|},$$
$$\epsilon_{II} = \frac{|S_{1t} \cap S_{0o}|}{|S_{1t}|},$$

where $|S|$ is the cardinality of the set S. Type I error measures the rate of missing noisy variables and Type II error measures the rate of deleting non-noisy variables. We hope that both ϵ_I and ϵ_{II} are small. Especially, we want ϵ_{II} small. That is, deleting non-noisy variables is worse than keeping some noisy variables. If $S_{0t} = \emptyset$, then we define $\epsilon_I = 0$. If $S_{1t} = \emptyset$, then we define $\epsilon_{II} = 0$.

To measure the agreement between the true partition and the partition obtained by clustering algorithms, we can use the five external indexes studied in Milligan (1986) (see Section 4.9.1). For real data sets with known classes, we directly count the number of misclassifications.

We denote `SCP` (variable Selection method based on Compact Projection) and `WCP` (variable Weighting method based on Compact Projection) respectively as the variable selection and weighting method based on the Algorithm 5.10 implemented with the `CP` method II.

For each data set, we obtain three k_0-cluster partitions where k_0 is the true number of clusters of the data set. The first partition is obtained with all the original variables. The second partition is obtained with the resultant non-noisy variables by `SCP`. The third partition is obtained with weighted variables where weights are obtained by `WCP`. When doing clustering, we provide the clustering algorithm with the true number of clusters. However, when calculating the weight vectors we only assume that the true number of clusters is between 2 and 10. The input parameters are set to $k_0^{\text{low}} = 2$, $k_0^{\text{upp}} = 10$, $k_0^{\text{step}} = 1$, $\epsilon_w = 0.1$, and $maxiter = 2$.

Table 5.7: Average Type I and II errors for simulated data sets.

data	estimated Type I error	estimated Type II error
close	0.028 (0.158)	0.052 (0.123)
separated	0.014 (0.112)	0.006 (0.030)
well-separated	0 (0)	0.012 (0.032)

5.11.1 Simulated Data Sets

In this subsection, we use 243 simulated data sets generated by the design in Section 3.9 to study the performance of the Algorithm 5.10 implemented with the CP method II. There are 81 data sets each for close, separated, and well-separated cluster structures. We use MKmeans clustering algorithm to get partitions.

The average Type I and II errors and corresponding standard errors for the results obtained by the SCP method are shown in Table 5.7. We can see from Table 5.7 that the averages of both Type I and II errors are small. This indicates that the CP method is effective in detecting noisy variables with a small probability of misclassifying non-noisy variable as noisy. When data sets have close structures, the variation of the estimated Type I errors are relatively large. This means the distributions of Type I and Type II errors are heavily skewed, i.e. the performance of a few cases are very bad. Overall, the performance is good. Table 5.8 shows that SCP and WCP can improve the

Table 5.8: The average values of the five external indexes for the 243 simulated data sets (the true number of clusters is used for clustering, but not for SCP/WCP).

data (method)	HA	MA	Rand	FM	Jaccard
close	0.728 (0.191)	0.729 (0.190)	0.923 (0.042)	0.778 (0.172)	0.662 (0.189)
close (SCP)	0.792 (0.095)	0.792 (0.095)	0.937 (0.027)	0.833 (0.087)	0.722 (0.110)
close (WCP)	0.519 (0.167)	0.520 (0.167)	0.855 (0.061)	0.615 (0.153)	0.462 (0.166)
separated	0.886 (0.231)	0.886 (0.231)	0.972 (0.054)	0.903 (0.200)	0.866 (0.243)
separated (SCP)	0.983 (0.010)	0.983 (0.010)	0.995 (0.003)	0.986 (0.009)	0.973 (0.016)
separated (WCP)	0.908 (0.084)	0.908 (0.084)	0.973 (0.026)	0.926 (0.070)	0.869 (0.108)
well-separated	0.903 (0.247)	0.903 (0.247)	0.977 (0.060)	0.917 (0.213)	0.896 (0.255)
well-separated (SCP)	0.999 (0.001)	0.999 (0.001)	1.000 (0.000)	0.999 (0.001)	0.998 (0.002)
well-separated (SCP)	0.975 (0.059)	0.976 (0.058)	0.992 (0.021)	0.981 (0.045)	0.966 (0.075)

recovery rate of clustering algorithm. In most cases, the average recovery rates for SCP and WCP are relatively higher than those for equal weighting, while the standard deviations for SCP and WCP are relatively smaller than those for equal weighting. The standard deviations for equal weighting are

quite large. This indicates that sometimes, the recovery rates are quite low if noisy variables are not downweighted or eliminated. In these cases, noisy variables mask the true cluster structures.

The performance of the WCP method is not as good as that of the SCP, especially for the data sets with close cluster structures. It indicates that some non-noisy variables might be eliminated or downweighted.

The table also shows that as the degree of separation among clusters increases, the effect of noisy variables decreases, and better partitions and better weight vectors can be obtained. For these 243 simulated data sets, overall SCP works better than WCP does.

5.11.2 Real Data Sets

In this subsection, we use the three pen digits data sets that we used in Section 4.9.4. These three data sets are extracted from the testing set of hand-written digit samples which is available at UCI Machine Learning Repository (Blake et al. 1998). Each row of these data is a sample of a hand-written digit. The 16 columns correspond to the 8 pairs of (x, y) tablet coordinate values of the sample digits which are recorded according to the order of strokes that the writer wrote the digits on the tablet. The data set DAT1 contains 1065 samples from digits 2, 4, and 6 and the data set DAT2 contains 500 samples from digits 4, 5, 6 and the data set DAT3 contains 2436 samples from digits 1, 3, 4, 6, 8, 9 and 0.

The digits 2, 4, and 6 are quite different and the stroke orders of each digit are almost unique, i.e. there is no subclasses contained in each digit class. So it is relatively easy to detect the class structures in DAT1. However, the stroke orders of some digits in DAT2 and DAT3 are not unique so that several subclasses may be contained in these digit classes.

These three data sets are quite challenging in that the data classes are far from elliptical in shape and some classes contain distinct subclasses. We compare 6 clustering algorithms — kmeans, MKmeans, PAM/CLARA, Ward, EMclust0, and Mclust — to get partitions. After variable weighting/selection, we provide the true number of classes to the 6 clustering algorithms to obtain the final partitions of these three data sets. In Section 5.12, we combine the SEQCLUST method we proposed in Chapter 4 with the CP method so that we do not need to provide the true number of classes. In either case, we do not need the true numbers of classes to obtain weight vectors.

The values of the five external indexes for the 3 data sets are listed in Table 5.9, 5.10,

Table 5.9: Values of the external indexes for DAT1 (the true number of clusters is used for clustering, but not for SCP/WCP).

Method	HA	MA	Rand	FM	Jaccard
kmeans	0.899	0.900	0.955	0.933	0.874
kmeans (SCP)	0.899	0.900	0.955	0.933	0.874
kmeans (WCP)	0.757	0.757	0.892	0.838	0.721
MKmeans	0.899	0.900	0.955	0.933	0.874
MKmeans (SCP)	0.899	0.900	0.955	0.933	0.874
MKmeans (WCP)	0.764	0.765	0.895	0.843	0.729
PAM/CLARA	0.908	0.908	0.959	0.938	0.884
PAM/CLARA (SCP)	0.908	0.908	0.959	0.938	0.884
PAM/CLARA (WCP)	0.752	0.753	0.889	0.835	0.717
Ward	1.000	1.000	1.000	1.000	1.000
Ward (SCP)	1.000	1.000	1.000	1.000	1.000
Ward (WCP)	1.000	1.000	1.000	1.000	1.000
EMclust0	0.896	0.896	0.954	0.931	0.871
EMclust0 (SCP)	0.875	0.876	0.945	0.917	0.847
EMclust0 (WCP)	0.875	0.876	0.945	0.917	0.847
Mclust0	1.000	1.000	1.000	1.000	1.000
Mclust0 (SCP)	1.000	1.000	1.000	1.000	1.000
Mclust0 (WCP)	1.000	1.000	1.000	1.000	1.000

and 5.11 respectively. It seems that all variables in DAT1 are equally important to recover the true cluster structure (the second line is the same as the first line for each triplet in Table 5.9). For DAT2, SCP and WCP can improve the recover of the true cluster structures. For DAT3, the performance of the CP method depends on the clustering algorithm used. Among the three data sets, DAT1 is the easiest to find the cluster structure since the 3 classes are well-separated and there is no obvious subcluster in each class.

For the data set DAT1, the normal and quantile versions of the separation index matrix with $\alpha = 0.05$ for the partition obtained by Ward (SCP) are shown in Table 5.12.

The corresponding two tables for the partition obtained by Ward (WCP) are shown in Table 5.13 ($\alpha = 0.05$).

The separation indexes show that the 3 clusters obtained by Ward (SCP) and Ward (WCP) for the data set DAT1 are well-separated.

The normal and quantile versions of the separation index matrix for the partition of DAT2 obtained by the Ward method with SCP are given in Table 5.14 ($\alpha = 0.05$).

The corresponding two separation index matrices for the partition obtained by the Ward

Table 5.10: Values of the external indexes for DAT2 (the true number of clusters is used for clustering, but not for SCP/WCP).

Method	HA	MA	Rand	FM	Jaccard
kmeans	0.531	0.533	0.781	0.706	0.542
kmeans (SCP)	0.604	0.605	0.816	0.750	0.596
kmeans (WCP)	0.627	0.628	0.827	0.764	0.615
MKmeans	0.531	0.533	0.781	0.706	0.542
MKmeans (SCP)	0.604	0.605	0.816	0.750	0.596
MKmeans (WCP)	0.647	0.648	0.836	0.777	0.632
PAM/CLARA	0.531	0.532	0.782	0.702	0.538
PAM/CLARA (SCP)	0.565	0.567	0.800	0.721	0.562
PAM/CLARA (WCP)	0.647	0.648	0.836	0.777	0.632
Ward	0.625	0.626	0.826	0.763	0.613
Ward (SCP)	0.632	0.633	0.829	0.767	0.619
Ward (WCP)	0.647	0.648	0.836	0.777	0.632
EMclust0	0.624	0.625	0.825	0.763	0.613
EMclust0 (SCP)	0.627	0.628	0.826	0.766	0.616
EMclust0 (WCP)	0.589	0.591	0.808	0.742	0.586
Mclust0	0.606	0.608	0.817	0.751	0.598
Mclust0 (SCP)	0.606	0.608	0.817	0.751	0.598
Mclust0 (WCP)	0.574	0.575	0.801	0.733	0.574

method with WCP are given in Table 5.15 ($\alpha = 0.05$).

The separation indexes show that the 3 clusters obtained by Ward (SCP) and Ward (WCP) for the data set DAT2 are well-separated.

The normal and quantile versions of the separation index matrix for the partition of DAT3 obtained by the Ward method with SCP are given in Tables 5.16 and 5.17 ($\alpha = 0.05$).

The corresponding two separation index matrices for the partition obtained by the Ward method with WCP are given in Tables 5.18 and 5.19 ($\alpha = 0.05$).

The separation indexes show that the 7 clusters obtained by Ward (SCP) and Ward (WCP) for the data set DAT3 are well-separated.

5.12 Integration of CP to the SEQCLUST Method

In this section, we study the performance of the the SEQCLUST clustering method that we proposed in Chapter 4 combined with the CP method. More specifically, we first use the CP method to downweight or eliminate noisy variables, then apply the SEQCLUST clustering method to simul-

Table 5.11: Values of the external indexes for DAT3 (the true number of clusters is used for clustering, but not for SCP/WCP).

Method	HA	MA	Rand	FM	Jaccard
kmeans	0.488	0.491	0.864	0.570	0.396
kmeans (SCP)	0.433	0.435	0.840	0.532	0.357
kmeans (WCP)	0.498	0.501	0.873	0.573	0.401
MKmeans	0.530	0.533	0.877	0.604	0.431
MKmeans (SCP)	0.502	0.505	0.870	0.580	0.407
MKmeans (WCP)	0.396	0.399	0.840	0.491	0.324
PAM/CLARA	0.579	0.582	0.895	0.641	0.471
PAM/CLARA (SCP)	0.532	0.535	0.874	0.610	0.435
PAM/CLARA (WCP)	0.501	0.504	0.872	0.577	0.404
Ward	0.582	0.585	0.894	0.644	0.475
Ward (SCP)	0.642	0.644	0.910	0.695	0.532
Ward (WCP)	0.553	0.555	0.879	0.628	0.454
EMclust0	0.461	0.464	0.854	0.550	0.376
EMclust0 (SCP)	0.409	0.411	0.835	0.511	0.338
EMclust0 (WCP)	0.579	0.581	0.892	0.643	0.473
Mclust0	0.473	0.475	0.843	0.576	0.393
Mclust0 (SCP)	0.508	0.510	0.859	0.598	0.418
Mclust0 (WCP)	0.543	0.546	0.879	0.616	0.443

taneously estimate the number of clusters and obtain a partition.

For the 243 simulate data sets, we use the MKmeans clustering method to obtain partitions. The average Type I and II errors for the simulated data sets are shown in Table 5.20. It is not surprising that the values in Table 5.20 are quite similar to those in Table 5.7 since the CP method does not directly use the information about the true number of clusters. For each data set, we obtain three partitions. The first partition is obtained by directly applying the SEQCLUST method to all variables (i.e. all variables get equal weights). The second partition is obtained by applying

Table 5.12: The separation index matrix for DAT1 ($\alpha = 0.05$). The partition is obtained by Ward (SCP).

Normal version

	1	2	3
1	-1.000	0.459	0.492
2	0.459	-1.000	0.618
3	0.492	0.618	-1.000

Quantile version

	1	2	3
1	1.000	0.472	0.479
2	0.472	1.000	0.633
3	0.479	0.633	1.000

Table 5.13: The separation index matrix for DAT1 ($\alpha = 0.05$). The partition is obtained by Ward (WCP).

Normal version

	1	2	3
1	-1.000	0.459	0.492
2	0.459	-1.000	0.618
3	0.492	0.618	-1.000

Quantile version

	1	2	3
1	1.000	0.472	0.479
2	0.472	1.000	0.633
3	0.479	0.633	1.000

Table 5.14: The separation index matrix for DAT2 ($\alpha = 0.05$). The partition is obtained by Ward (SCP).

Normal version

	1	2	3
1	-1.000	0.267	0.850
2	0.267	-1.000	0.675
3	0.850	0.675	-1.000

Quantile version

	1	2	3
1	-1.000	0.219	0.860
2	0.219	-1.000	0.689
3	0.860	0.689	-1.000

Table 5.15: The separation index matrix for DAT2 ($\alpha = 0.05$). The partition is obtained by Ward (WCP).

Normal version

	1	2	3
1	-1.000	0.386	0.774
2	0.386	-1.000	0.697
3	0.774	0.697	-1.000

Quantile version

	1	2	3
1	-1.000	0.342	0.787
2	0.342	-1.000	0.713
3	0.787	0.713	-1.000

Table 5.16: The normal version separation index matrix for DAT3 ($\alpha = 0.05$). The partition is obtained by Ward (SCP).

	1	2	3	4	5	6	7
1	-1.000	0.597	0.457	0.510	0.384	0.308	0.435
2	0.597	-1.000	0.445	0.132	0.646	0.778	0.393
3	0.457	0.445	-1.000	0.240	0.491	0.748	0.380
4	0.510	0.132	0.240	-1.000	0.663	0.800	0.545
5	0.384	0.646	0.491	0.663	-1.000	0.324	0.464
6	0.308	0.778	0.748	0.800	0.324	-1.000	0.595
7	0.435	0.393	0.380	0.545	0.464	0.595	-1.000

Table 5.17: The quantile version separation index matrix for DAT3 ($\alpha = 0.05$). The partition is obtained by Ward (SCP).

	1	2	3	4	5	6	7
1	-1.000	0.582	0.446	0.525	0.346	0.267	0.396
2	0.582	-1.000	0.456	0.097	0.659	0.788	0.384
3	0.446	0.456	-1.000	0.201	0.451	0.738	0.349
4	0.525	0.097	0.201	-1.000	0.695	0.795	0.510
5	0.346	0.659	0.451	0.695	-1.000	0.362	0.472
6	0.267	0.788	0.738	0.795	0.362	-1.000	0.585
7	0.396	0.384	0.349	0.510	0.472	0.585	-1.000

Table 5.18: The normal version separation index matrix for DAT3 ($\alpha = 0.05$). The partition is obtained by Ward (WCP).

	1	2	3	4	5	6	7
1	-1.000	0.620	0.631	0.208	0.617	0.140	0.767
2	0.620	-1.000	0.337	0.717	0.735	0.383	0.264
3	0.631	0.337	-1.000	0.546	0.588	0.424	0.420
4	0.208	0.717	0.546	-1.000	0.748	0.484	0.765
5	0.617	0.735	0.588	0.748	-1.000	0.509	0.574
6	0.140	0.383	0.424	0.484	0.509	-1.000	0.614
7	0.767	0.264	0.420	0.765	0.574	0.614	-1.000

Table 5.19: The quantile version separation index matrix for DAT3 ($\alpha = 0.05$). The partition is obtained by Ward (WCP).

	1	2	3	4	5	6	7
1	-1.000	0.643	0.636	0.164	0.621	0.039	0.754
2	0.643	-1.000	0.317	0.707	0.707	0.381	0.207
3	0.636	0.317	-1.000	0.518	0.606	0.425	0.359
4	0.164	0.707	0.518	-1.000	0.759	0.480	0.761
5	0.621	0.707	0.606	0.759	-1.000	0.469	0.590
6	0.039	0.381	0.425	0.480	0.469	-1.000	0.618
7	0.754	0.207	0.359	0.761	0.590	0.618	-1.000

Table 5.20: Average Type I and Type II errors for simulated data sets obtained by the SEQCLUST algorithm implemented with CP.

data	Type I error	Type II error
close	0.035 (0.162)	0.049 (0.118)
separated	0.013 (0.111)	0.008 (0.033)
well-separated	0.000 (0.000)	0.015 (0.041)

the SEQCLUST method to the remaining variables after eliminating noisy variables by using the SCP method. The third method is obtained by applying the SEQCLUST method to the variables weighted by the WCP method.

The total numbers and sizes of under- and over-estimate of the number of clusters are shown in Table 5.21. And the average values of the five external indexes for the simulated data sets are shown in Table 5.22. Both SCP and WCP performed much better than the equal weighting method,

Table 5.21: The total numbers and sizes of under- and over-estimate of the number of clusters for simulated data sets obtained by the SEQCLUST algorithm implemented with CP (m_- and s_- are total the number and size of underestimates while m_+ and s_+ are the total number and size of overestimates).

Data (Method)	m_- (s_-)	m_+ (s_+)
close	27 (145)	11 (33)
close (SCP)	14 (58)	1 (1)
close (WCP)	58 (300)	2 (7)
separated	8 (43)	2 (7)
separated (SCP)	1 (1)	0 (0)
separated (WCP)	6 (9)	0 (0)
well-separated	9 (45)	1 (2)
well-separated (SCP)	0 (0)	0 (0)
well-separated (WCP)	2 (2)	0 (0)

Table 5.22: Average values of the external indexes for the simulated data sets obtained by the SEQCLUST algorithm implemented with CP

data (method)	HA	MA	Rand	FM	Jaccard
close	0.529 (0.369)	0.529 (0.369)	0.703 (0.336)	0.688 (0.212)	0.534 (0.274)
close (SCP)	0.702 (0.263)	0.702 (0.263)	0.850 (0.229)	0.789 (0.153)	0.661 (0.204)
close (WCP)	0.287 (0.351)	0.288 (0.351)	0.480 (0.345)	0.564 (0.207)	0.371 (0.260)
separated	0.901 (0.234)	0.901 (0.234)	0.944 (0.165)	0.929 (0.157)	0.889 (0.224)
separated (SCP)	0.984 (0.015)	0.984 (0.015)	0.995 (0.005)	0.987 (0.012)	0.974 (0.023)
separated (WCP)	0.945 (0.062)	0.945 (0.062)	0.984 (0.017)	0.955 (0.050)	0.917 (0.083)
well-separated	0.908 (0.253)	0.908 (0.253)	0.945 (0.172)	0.939 (0.166)	0.910 (0.240)
well-separated (SCP)	0.999 (0.002)	0.999 (0.002)	1.000 (0.001)	0.999 (0.002)	0.999 (0.004)
well-separated (SCP)	0.991 (0.026)	0.991 (0.026)	0.997 (0.008)	0.993 (0.020)	0.986 (0.038)

except that the WCP method performed poorly for the data sets with close cluster structures. This indicates that the effects of noisy variables were downweighted. Again, as the degree of separation among clusters increases, the effect of noisy variables decreases and we can get better partitions

and weight vectors.

For the three pen digit data sets, we also compare the results obtained by equal weighting, SCP, and WCP. The estimates of the number of clusters and the values of the five external indexes for the 3 pen digits data sets are shown in Tables 5.23, 5.24, and 5.25 respectively. The interval estimates of the number of clusters are shown in Tables 5.26, 5.27, and 5.28 respectively. For the data sets DAT2 and DAT3, the estimated numbers of clusters are larger than the original number of clusters. By the analysis in Section 4.9.4, we know that these overestimations are reasonable since some classes of digit samples contain distinct subclasses.

We observe that the performance of the three methods (equal weighting, SCP, and WCP) are similar for the data set DAT1 in which the 3 clusters are well-separated. For DAT2, whether SCP and WCP can improve the recovery rates depends on the clustering method. For DAT3, both SCP and WCP did not perform as well as the equal weighting method did. We also observe that the SEQCLUST method implemented with CP could get better results than other clustering method combined with the CP method (results in Table 5.23 are better than those in Table 5.9; results in Table 5.24 are better than those in Table 5.10; results in Table 5.25 are better than those in Table 5.11;). For these three data sets, the original variables all are on the same scale and theoretically no variable is noisy, but some variables should be more important than others.

For the data set DAT1, the normal and quantile versions of the separation index matrix with $\alpha = 0.01$ for the partition obtained by SEQCLUST with Ward (SCP) are shown in Table 5.29. [1]

The corresponding two tables for the partition obtained by the SEQCLUST method with Ward (WCP) are shown in Table 5.30 ($\alpha = 0.015$). [2]

For the data set DAT1, the separation indexes show that the 3 clusters obtained by the SEQCLUST method with Ward (SCP) and with Ward (WCP) are well-separated.

For the data set DAT2, the normal and quantile versions of the separation index matrix with $\alpha = 0.015$ for the partition obtained by SEQCLUST with Ward (SCP) are shown in Table 5.31. [3]

The corresponding two tables for the partition obtained by the SEQCLUST method with Ward (WCP) are shown in Table 5.32 ($\alpha = 0.015$). [4]

For the data set DAT2, the separation indexes show that the 4 clusters obtained by the

[1]The α value is output with the final partition by the SEQCLUST method with Ward (SCP).
[2]The α value is output with the final partition by the SEQCLUST method with Ward (WCP).
[3]The α value is output with the final partition by the SEQCLUST method with Ward (SCP).
[4]The α value is output with the final partition by the SEQCLUST method with Ward (WCP).

Table 5.23: Estimated numbers of clusters and external index values of DAT1 obtained by the SEQCLUST algorithm implemented with CP (true $k_0 = 3$)

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard
SEQCLUST (kmeans)*	3	0.951	0.951	0.978	0.967	0.937
kmeans (SCP)	3	0.961	0.961	0.983	0.974	0.949
kmeans (WCP)	3	0.947	0.947	0.976	0.964	0.931
SEQCLUST (MKmeans)*	3	0.951	0.951	0.978	0.967	0.937
MKmeans (SCP)	3	0.951	0.951	0.978	0.967	0.937
MKmeans (WCP)	3	0.970	0.971	0.987	0.980	0.961
SEQCLUST (PAM/CLARA)*	3	0.951	0.951	0.978	0.968	0.937
PAM/CLARA (SCP)	3	0.956	0.956	0.980	0.971	0.943
PAM/CLARA (WCP)	3	0.995	0.995	0.998	0.997	0.993
SEQCLUST (Ward)*	3	1.000	1.000	1.000	1.000	1.000
Ward (SCP)	3	1.000	1.000	1.000	1.000	1.000
Ward (WCP)	3	1.000	1.000	1.000	1.000	1.000
SEQCLUST (EMclust)*	3	1.000	1.000	1.000	1.000	1.000
EMclust0 (SCP)	3	1.000	1.000	1.000	1.000	1.000
EMclust0 (WCP)	3	1.000	1.000	1.000	1.000	1.000
SEQCLUST (Mclust)*	3	1.000	1.000	1.000	1.000	1.000
Mclust0 (SCP)	3	1.000	1.000	1.000	1.000	1.000
Mclust0 (WCP)	3	1.000	1.000	1.000	1.000	1.000

* The results are the same as those in Table 4.9.

SEQCLUST method with Ward (SCP) and with Ward (WCP) are well-separated.

For the data set DAT3, the normal and quantile versions of the separation index matrix with $\alpha = 0.02$ for the partition obtained by SEQCLUST with Ward (SCP) are shown in Tables 5.33 and 5.34. [5] The table shows that the 10 clusters are separated.

The corresponding two tables for the partition obtained by the SEQCLUST method with Ward (WCP) are shown in Tables 5.35 and 5.36 ($\alpha = 0.02$). [6] The table shows that the 6 clusters are separated.

For the data set DAT3, the averages of the separation indexes are listed in the following table:

version	SCP (Ward)	WCP (Ward)
normal	0.593	0.484
quantile	0.591	0.462

From the above table, we can see that the partition obtained by SCP (Ward) is better than

[5] The α value is output with the final partition by the SEQCLUST method with Ward (SCP).
[6] The α value is output with the final partition by the SEQCLUST method with Ward (WCP).

Table 5.24: Estimated numbers of clusters and external index values of DAT2 obtained by the SEQCLUST algorithm implemented with CP (true $k_0 = 3$)

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard
SEQCLUST (kmeans)*	4	0.861	0.861	0.941	0.906	0.823
kmeans (SCP)	5	0.813	0.814	0.922	0.874	0.766
kmeans (WCP)	4	0.853	0.853	0.937	0.901	0.814
SEQCLUST (MKmeans)*	5	0.812	0.813	0.921	0.873	0.766
MKmeans (SCP)	5	0.813	0.814	0.922	0.874	0.766
MKmeans (WCP)	4	0.853	0.853	0.937	0.901	0.814
SEQCLUST (PAM/CLARA)*	5	0.791	0.792	0.912	0.859	0.742
PAM/CLARA (SCP)	5	0.789	0.790	0.911	0.857	0.739
PAM/CLARA (WCP)	4	0.806	0.807	0.918	0.869	0.761
SEQCLUST (Ward)*	5	0.833	0.834	0.930	0.888	0.789
Ward (SCP)	5	0.837	0.838	0.931	0.891	0.794
Ward (WCP)	4	0.881	0.882	0.949	0.921	0.848
SEQCLUST (EMclust)*	4	0.821	0.822	0.923	0.879	0.781
EMclust0 (SCP)	4	0.848	0.849	0.935	0.897	0.809
EMclust0 (WCP)	4	0.847	0.847	0.932	0.893	0.801
SEQCLUST (Mclust)*	4	0.872	0.872	0.945	0.914	0.836
Mclust0 (SCP)	4	0.872	0.872	0.945	0.914	0.836
Mclust0 (WCP)	4	0.784	0.785	0.907	0.854	0.743

* The results are the same as those in Table 4.12.

that obtained by WCP (Ward) since the average separation index value of the former is greater that of the latter. This conclusion is consistent with the comparison results by using the five external indexes shown in Table 5.25. Note that unlike Rand indexes, the comparisons based on the separation index matrices are comparisons without using the known class structures.

The clustering methods seem to produce well-separated clusters for DAT1, DAT2 and DAT3. The reason that different methods get quite different partitions is probably because of the sparseness in high dimensional space.

5.13 Discussion

In this chapter, we propose a method called CP to do variable weighting and selection. The weight vector is based on the linear combination of eigenvectors of the product of the between-cluster distance matrix and the within-cluster distance matrix. The weight vector has 1-moment-noisy-variable-detection and scale-equivariance properties. We use weight-vector-averaging technique to improve CP so that we can calculate the weight vector without the specification of the true number

Table 5.25: Estimated numbers of clusters and external index values of DAT3 obtained by the SEQCLUST algorithm implemented with CP (true $k_0 = 7$)

Method	$\hat{k}_0$	HA	MA	Rand	FM	Jaccard
SEQCLUST (kmeans)*	10	0.686	0.688	0.927	0.730	0.573
kmeans (SCP)	10	0.688	0.690	0.930	0.732	0.572
kmeans (WCP)	8	0.576	0.578	0.892	0.640	0.470
SEQCLUST (MKmeans)*	10	0.758	0.760	0.946	0.794	0.651
MKmeans (SCP)	8	0.525	0.528	0.875	0.600	0.427
MKmeans (WCP)	5	0.428	0.430	0.817	0.551	0.361
SEQCLUST (PAM/CLARA)*	9	0.636	0.638	0.912	0.687	0.523
PAM/CLARA (SCP)	8	0.510	0.512	0.866	0.593	0.417
PAM/CLARA (WCP)	8	0.576	0.579	0.892	0.640	0.470
SEQCLUST (Ward)*	10	0.784	0.786	0.951	0.816	0.684
Ward (SCP)	10	0.750	0.752	0.944	0.787	0.642
Ward (WCP)	6	0.552	0.553	0.851	0.580	0.399
SEQCLUST (EMclust)*	13	0.770	0.771	0.928	0.709	0.528
EMclust0 (SCP)	9	0.732	0.733	0.925	0.739	0.586
EMclust0 (WCP)	8	0.577	0.579	0.896	0.638	0.468
SEQCLUST (Mclust)*	10	0.782	0.783	0.914	0.678	0.511
EMclust0 (SCP)	12	0.724	0.726	0.935	0.748	0.588
Mclust0 (WCP)	9	0.626	0.628	0.908	0.680	0.515

* The results are the same as those in Table 4.15.

of clusters. A preliminary theoretical validation of the weight-vector-averaging technique shows that under certain conditions, the distributions of the noisy variables do not change if we merge k_0 clusters into $k_0 - 1$ clusters or if we split k_0 clusters into $k_0 + 1$ clusters where k_0 is the true number of clusters.

We applied CP to 243 simulated data sets generated from the design in Section 3.9 and 3 pen digits data sets studied in Chapter 4. The results shows that CP has good performance on noisy-variable detection. The average Type I and II errors for the 243 simulated data set are small. The SCP method has consistent good performance for the 243 simulated data sets in that it improves the recover of the true cluster structures. The WCP method has good performance for simulated data sets with separated and well-separated cluster structures. However its performance for simulated data sets with close cluster structures is poor. The performances of the SCP and WCP method for the 3 pen digits are mixed. From these observations we conclude that the performance of a variable weighting/selection method depends on the partition (clustering method) and cluster structures. For clusters convex in shape, SCP and WCP have good performance. For other cluster

Table 5.26: Interval estimates of the number of clusters obtained by the SEQCLUST method implemented with CP for DAT1

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[3, 3]	SEQCLUST (Ward)	[3, 4]
kmeans (SCP)	[3, 3]	Ward (SCP)	[3, 4]
kmeans (WCP)	[2, 4]	Ward (WCP)	[3, 3]
SEQCLUST (MKmeans)	[3, 3]	SEQCLUST (EMclust)	[3, 4]
MKmeans (SCP)	[3, 3]	EMclust (SCP)	[3, 4]
MKmeans (WCP)	[3, 3]	EMclust (WCP)	[3, 3]
SEQCLUST (PAM/CLARA)	[3, 4]	SEQCLUST (Mclust)	[3, 3]
PAM/CLARA (SCP)	[3, 4]	Mclust (SCP)	[3, 3]
PAM/CLARA (WCP)	[3, 3]	Mclust (WCP)	[3, 3]

Table 5.27: Interval estimates of the number of clusters obtained by the SEQCLUST method implemented with CP for DAT2

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[4, 5]	SEQCLUST (Ward)	[5, 5]
kmeans (SCP)	[5, 6]	Ward (SCP)	[5, 5]
kmeans (WCP)	[4, 4]	Ward (WCP)	[4, 4]
SEQCLUST (MKmeans)	[5, 5]	SEQCLUST (EMclust)	[4, 4]
MKmeans (SCP)	[5, 6]	EMclust (SCP)	[4, 4]
MKmeans (WCP)	[4, 5]	EMclust (WCP)	[4, 4]
SEQCLUST (PAM/CLARA)	[4, 5]	SEQCLUST (Mclust)	[4, 4]
PAM/CLARA (SCP)	[4, 5]	Mclust (SCP)	[4, 4]
PAM/CLARA (WCP)	[4, 4]	Mclust (WCP)	[4, 6]

shapes, SCP and WCP might not work well. We will study in future research on how to do variable weighting/selection for non-convex-shaped cluster structures.

For the CP method, there are still some questions unsolved. For example, how many iterations do we need in CP? Currently, we set $maxiter = 2$.

Another example is how to choose an appropriate value of the threshold ϵ_w which is used to determine if a variable is noisy or not. One possible way is to take the same criterion used in Montanari and Lizzani (2001). The rationale is given below.

If variables $x_1, \ldots, x_q$ are non-noisy and $x_{q+1}, \ldots, x_p$ are noisy, then we can represent the within- and between-cluster distance matrices A and B as

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 & 0 \\ 0 & \boldsymbol{A}_2 \end{pmatrix}, \boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_1 & 0 \\ 0 & 0 \end{pmatrix},$$

where $\boldsymbol{B}_1 = \sum_{k_1=1}^{k_0-1} \sum_{k_2=k_1+1}^{k_0} \pi_{k_1} \pi_{k_2} (\boldsymbol{\theta}_{k_1,1} - \boldsymbol{\theta}_{k_2,1}) (\boldsymbol{\theta}_{k_1,1} - \boldsymbol{\theta}_{k_2,1})^T$, and the $q \times 1$ vector $\boldsymbol{\theta}_{k,1}$ is the

Table 5.28: Interval estimates of the number of clusters obtained by the SEQCLUST method implemented with CP for DAT3

Method	Interval	Method	Interval
SEQCLUST (kmeans)	[9, 11]	SEQCLUST (Ward)	[10, 12]
kmeans (SCP)	[9, 10]	Ward (SCP)	[8, 10]
kmeans (WCP)	[7, 9]	Ward (WCP)	[5, 6]
SEQCLUST (MKmeans)	[10, 13]	SEQCLUST (EMclust)	[3, 13]
MKmeans (SCP)	[7, 8]	EMclust (SCP)	[6, 11]
MKmeans (WCP)	[3, 7]	EMclust (WCP)	[7, 10]
SEQCLUST (PAM/CLARA)	[7, 11]	SEQCLUST (Mclust)	[6, 10]
PAM/CLARA (SCP)	[7, 8]	Mclust (SCP)	[9, 12]
PAM/CLARA (WCP)	[5, 8]	Mclust (WCP)	[9, 9]

Table 5.29: The separation index matrix for DAT1 ($\alpha = 0.01$). The partition is obtained by the SEQCLUST method with Ward (SCP).

Normal version

	1	2	3
1	-1.000	0.618	0.491
2	0.618	-1.000	0.459
3	0.491	0.459	-1.000

Quantile version

	1	2	3
1	-1.000	0.632	0.478
2	0.632	-1.000	0.472
3	0.478	0.472	-1.000

mean vector of non-noisy variables of the k-th cluster. Then

$$\boldsymbol{A}^{-1}\boldsymbol{B} = \begin{pmatrix} \boldsymbol{A}_1^{-1}\boldsymbol{B}_1 & 0 \\ 0 & 0 \end{pmatrix}.$$

By definition of the eigenvalue and eigenvector, we have

$$\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{\alpha}_1 = \lambda_1\boldsymbol{\alpha}_1,$$

where λ_1 and $\boldsymbol{\alpha}_1$ is the largest eigenvalue and corresponding eigenvector of the matrix $\boldsymbol{A}^{-1}\boldsymbol{B}$. That

Table 5.30: The separation index matrix for DAT1 ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with Ward (WCP).

Normal version

	1	2	3
1	-1.000	0.618	0.491
2	0.618	-1.000	0.459
3	0.491	0.459	-1.000

Quantile version

	1	2	3
1	-1.000	0.632	0.478
2	0.632	-1.000	0.472
3	0.478	0.472	-1.000

Table 5.31: The separation index matrix for DAT2 ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with Ward (SCP).

Normal version

	1	2	3	4	5
1	-1.000	0.487	0.696	0.456	0.462
2	0.487	-1.000	0.815	0.440	0.517
3	0.696	0.815	-1.000	0.816	0.850
4	0.456	0.440	0.816	-1.000	0.323
5	0.462	0.517	0.850	0.323	-1.000

Quantile version

	1	2	3	4	5
1	-1.000	0.466	0.712	0.434	0.470
2	0.466	-1.000	0.794	0.447	0.529
3	0.712	0.794	-1.000	0.814	0.859
4	0.434	0.447	0.814	-1.000	0.295
5	0.470	0.529	0.859	0.295	-1.000

Table 5.32: The separation index matrix for DAT2 ($\alpha = 0.015$). The partition is obtained by the SEQCLUST method with Ward (WCP).

Normal version

	1	2	3	4
1	-1.000	0.817	0.368	0.488
2	0.817	-1.000	0.848	0.696
3	0.368	0.848	-1.000	0.457
4	0.488	0.696	0.457	-1.000

Quantile version

	1	2	3	4
1	-1.000	0.808	0.321	0.511
2	0.808	-1.000	0.857	0.712
3	0.321	0.857	-1.000	0.469
4	0.511	0.712	0.469	-1.000

Table 5.33: The normal version separation index matrix for DAT3 ($\alpha = 0.02$). The partition is obtained by the SEQCLUST method with Ward (SCP).

	1	2	3	4	5	6	7	8	9	10
1	-1.000	0.380	0.504	0.595	0.530	0.595	0.636	0.655	0.464	0.419
2	0.380	-1.000	0.543	0.323	0.552	0.748	0.594	0.482	0.491	0.629
3	0.504	0.543	-1.000	0.559	0.525	0.831	0.895	0.385	0.727	0.756
4	0.595	0.323	0.559	-1.000	0.274	0.810	0.770	0.437	0.698	0.697
5	0.530	0.552	0.525	0.274	-1.000	0.792	0.833	0.231	0.708	0.684
6	0.595	0.748	0.831	0.810	0.792	-1.000	0.790	0.830	0.324	0.278
7	0.636	0.594	0.895	0.770	0.833	0.790	-1.000	0.593	0.715	0.584
8	0.655	0.482	0.385	0.437	0.231	0.830	0.593	-1.000	0.707	0.748
9	0.464	0.491	0.727	0.698	0.708	0.324	0.715	0.707	-1.000	0.342
10	0.41	0.629	0.756	0.697	0.684	0.278	0.584	0.748	0.342	-1.000

Table 5.34: The quantile version separation index matrix for DAT3 ($\alpha = 0.02$). The partition is obtained by the SEQCLUST method with Ward (SCP).

	1	2	3	4	5	6	7	8	9	10
1	-1.000	0.349	0.494	0.594	0.512	0.585	0.617	0.642	0.472	0.417
2	0.349	-1.000	0.571	0.347	0.576	0.738	0.612	0.467	0.451	0.646
3	0.494	0.571	-1.000	0.605	0.514	0.817	0.910	0.416	0.737	0.764
4	0.594	0.347	0.605	-1.000	0.245	0.818	0.747	0.451	0.716	0.710
5	0.512	0.576	0.514	0.245	-1.000	0.797	0.809	0.152	0.712	0.678
6	0.585	0.738	0.817	0.818	0.797	-1.000	0.777	0.827	0.362	0.237
7	0.617	0.612	0.910	0.747	0.809	0.777	-1.000	0.622	0.716	0.572
8	0.642	0.467	0.416	0.451	0.152	0.827	0.622	-1.000	0.727	0.767
9	0.472	0.451	0.737	0.716	0.712	0.362	0.716	0.727	-1.000	0.283
10	0.417	0.646	0.764	0.710	0.678	0.237	0.572	0.767	0.283	-1.000

Table 5.35: The normal version separation index matrix for DAT3 ($\alpha = 0.02$). The partition is obtained by the SEQCLUST method with Ward (WCP).

	1	2	3	4	5	6
1	-1.000	0.544	0.767	0.651	0.208	0.140
2	0.544	-1.000	0.255	0.325	0.709	0.321
3	0.767	0.255	-1.000	0.426	0.765	0.614
4	0.651	0.325	0.426	-1.000	0.619	0.429
5	0.208	0.709	0.765	0.619	-1.000	0.484
6	0.140	0.321	0.614	0.429	0.484	-1.000

Table 5.36: The quantile version separation index matrix for DAT3 ($\alpha = 0.02$). The partition is obtained by the SEQCLUST method with Ward (WCP).

	1	2	3	4	5	6
1	-1.000	0.562	0.754	0.653	0.164	0.039
2	0.562	-1.000	0.204	0.316	0.697	0.327
3	0.754	0.204	-1.000	0.362	0.761	0.618
4	0.653	0.316	0.362	-1.000	0.569	0.418
5	0.164	0.697	0.761	0.569	-1.000	0.480
6	0.039	0.327	0.618	0.418	0.480	-1.000

is,

$$\begin{pmatrix} (\boldsymbol{A}_1^{-1}\boldsymbol{B}_1)\,\boldsymbol{\alpha}_{11} \\ 0 \end{pmatrix} = \begin{pmatrix} \lambda_1\boldsymbol{\alpha}_{11} \\ \lambda_1\boldsymbol{\alpha}_{12} \end{pmatrix}.$$

Since $\lambda_1 > 0$, thus $\boldsymbol{\alpha}_{12} = 0$. If all variables are noisy variables, then $\boldsymbol{B}_1 = 0$. Thus, $\boldsymbol{A}^{-1}\boldsymbol{B} = 0$ and $\boldsymbol{\alpha}_1$ can be any direction on the hyper-ball with radius 1. Hence $\boldsymbol{w} = \boldsymbol{\alpha}_1$ can be any direction on the hyper-ball with radius 1. According to Theorem 4 . 21 in Joe (1997, page 128), the marginal distribution of w_j has density

$$g_p(u) = \left[B\left(\frac{1}{2}, \frac{p-1}{2}\right)\right]^{-1} \left(1-u^2\right)^{(p-3)/2}, \ |u| \leq 1,$$

where B is the beta function and p is the number of variables. Thus we can take the same criterion used in Montanari and Lizzani (2001). That is, if

$$|w_j| < q_{10}(p),$$

then regard variable x_j is noisy, where $q_{10}(p)$ is the 10-th percentile of the random variable $Y = |U|$, where U has density g_p. The distribution function of Y is

$$F(y) = \Pr(|U| \leq y) = \Pr(-y \leq U \leq y) = \int_{-y}^{y} \left[B\left(\frac{1}{2}, \frac{p-1}{2}\right)\right]^{-1} \left(1-u^2\right)^{(p-3)/2} du, \ 0 \leq y \leq 1.$$

Finally, we want to point out that subject-matter knowledge should be used to help make decision in variable weighting/selection; for example, we need to use subject-matter knowledge to check if it makes sense that the variable declared as noisy are in fact less important.

Chapter 6

Summary and Future Research

There exist many clustering methods, but there are issues such as estimating the number of clusters, and variable selection/weighting that have not been well studied and/or not included in clustering programs in statistical software. In this dissertation, we study these issues with a theoretical basis. In particular, we addressed the following topics in cluster analysis for continuous data:

- Measuring the quality of a partition obtained from a clustering method.
- Determining when noisy variables hinder the discovery of cluster structure
- Eliminating or downweighting the effects of noisy variables
- Determining the number of clusters
- Visualizing cluster structure in lower dimensional space
- Generating challenging simulated data sets for clustering algorithms

We have addressed these issues based on a separation index and a compact-projection-based variable weighting/selection method. The separation index directly measures the magnitude of the sparse area between a pair of clusters and can be used to validate the partitions, derive a low dimensional visualization method, assign partial memberships to data points, control the distances among clusters when we generating simulated data sets, and estimate the number of clusters. The compact-projection-based variable weighting/selection method is to maximize the compactness of

projected clusters, where the compactness is based on the within- and between-cluster distance matrix. Both the separation index and the variable weighting/selection method have desired properties under certain conditions.

The key assumptions of the separation index and the compact-projection-based variable weighting and selection method are:

- variables are continuous;
- clusters are convex in shape;
- there are no missing values in data sets;

Also our methods might be sensitive to outliers since both the separation index and the compact-projection-based variable weighting and selection method depend on the cluster mean vectors and covariance matrices which are sensitive to outliers.

In our future research, we would like to extend our methods to have the ability to

- handle of mixed type data (some continuous and some categorical variables)
- handle outliers and make methods less sensitive to them.
- handle missing values

The majority of the literature on cluster analysis deals with continuous type data. However, it is quite common in real life problems that cases are described in terms of variables of mixed type (binary, nominal, ordinal and continuous variables). There are mainly two approaches to this problem (Gordon 1981, Section 2.42.): (1) Employ a general similarity coefficient which can incorporate information from different types of variables (e.g. Gower 1971; Huang 1998; Guha et al. 1999; Chiu et al. 2001); (2) Carry out separate analyses of the same set of cases, each analysis involving variables of a single type, and to attempt to synthesize the results from the different studies.

To extend our method to handle categorical variables, the covariance matrix might be replaced with a matrix of associations and distance would be replaced by dissimilarity.

In real data sets, it is common that a small portion of cases are scattered in-between and/or outside the clusters. These cases, called outliers, may affect the discovery of true cluster structures.

For continuous type data, researchers have proposed quite a few robust clustering algorithms to separate outliers from, or to downweight outliers' effects on, the majority cases. A simple method is to regard clusters whose sizes are small as outliers (e.g. Zhang et al. 1996). Zhuang et al. (1996) assumed a statistical model, in which the cases are divided into two parts: one part is the "majority" which is the part of interest and the other part is the "noise". Zhuang et al. (1996) iteratively applied this model to extract one cluster at a time. The cases left are regarded as outliers. Some researchers (e.g. Fukunaga and Hostetler 1975; Guha et al. 1998; Comaniciu and Meer 2002) shrunk cases toward cluster centers to downweight the effects of outliers. Frigui and Krishnapuram (1999) gave a review on robust clustering algorithms and proposed a new robust clustering algorithm by adding a penalty term and robustify the loss function in the objective function of the fuzzy c-means clustering algorithm.

A simple extension of our method to handle a small portion of outliers is to replace the covariance matrix with robust covariance matrix and use `MAD` to standardize data instead of `SD`.

Missing values are another commonly encountered problem in Statistics. The fact that few assumptions can be made hinders the development of the techniques to deal with missing data in cluster analysis (Gordon 1981, Section 2.4.3), and there is little cluster analysis literature addressing this problem. A general strategy mentioned in Gordon (1981) is to use weighted similarity coefficients, where missing values are assigned zero weights. Another method is to impute the missing values.

To extend our method to handle missing values, a possible approach is to appropriately modify the distance.

As we get more experience in applying our methods to larger and larger data sets, we will study computational complexity and improve the time-consuming parts of our algorithms.

Appendix A

Visualizing More Than Two Clusters

In cluster analysis, data sets are usually in high dimensional space (more than 3 dimension) and we could not visualize how separated the clusters are.[1] Many methods have been proposed to project high dimensional data into lower dimensions. A brief review is given in Dhillon et al. (2002). In this appendix, we propose a new low-dimensional visualization method and compare it with principal component analysis (PCA) and Dhillon et al.'s (2002) method.

Suppose that we obtain a partition for a high dimensional data set. We want to visualize how far apart the clusters are from a projection of the data into a low t-dimensional space ($t = 2$ or 3). We want the clusters to be as separated as possible in the projection. That is, we try to find a $p \times t$ ($t < p$) column-orthogonal matrix $\boldsymbol{A}^*$ (with $(\boldsymbol{A}^*)^T \boldsymbol{A}^* = \boldsymbol{I}_t$) such that

$$\boldsymbol{A}^* = \arg \max_{\boldsymbol{A}_{p\times t}, \boldsymbol{A}^T\boldsymbol{A}=\boldsymbol{I}_t} \text{tr}\left(\boldsymbol{A}^T \boldsymbol{B} \boldsymbol{A}\right), \tag{A.0.1}$$

where

$$\boldsymbol{B} = \frac{2}{k_0(k_0-1)} \sum_{i=1}^{k_0-1} \sum_{j=i+1}^{k_0} \text{E}\left[(\mathbf{Y}_i - \mathbf{Y}_j)(\mathbf{Y}_i - \mathbf{Y}_j)^T\right] \tag{A.0.2}$$

$$= \frac{2}{k_0} \sum_{i=1}^{k_0} \boldsymbol{\Sigma}_i + \frac{2}{k_0(k_0-1)} \sum_{i<j} (\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)^T, \tag{A.0.3}$$

$\mathbf{Y}_i$ is the random point in cluster i, k_0 is the number of clusters, $\boldsymbol{\theta}_i$ and $\boldsymbol{\Sigma}_i$ are the population mean vector and covariance matrix of the i-th cluster. When dealing with data, the sample mean vectors $\hat{\boldsymbol{\theta}}_i$ and covariance matrices $\hat{\boldsymbol{\Sigma}}_i$, $i = 1, \ldots, k_0$, are used. Note that if $\boldsymbol{A}$ is a projection matrix in

[1] This appendix refers to Figure 2.20 (page 32) and Figure 2.21 (page 35).

t-dimensional space, then $\mathrm{tr}\,(\boldsymbol{A}^T\boldsymbol{B}\boldsymbol{A})$ is the average over pairs of clusters of the average distance between projected points of two different clusters.

It is straightforward to show that the t columns of the matrix $\boldsymbol{A}$ correspond to the orthogonal eigenvectors of the first t largest eigenvalues of $\boldsymbol{B}$. The value of t is a good choice if $\sum_{i=1}^{t}\lambda_i/\sum_{j=1}^{p}\lambda_j$ is sufficiently large, where $\lambda_1 \geq \cdots \geq \lambda_p$ are the eigenvalues of $\boldsymbol{B}$. That is, the projection to t dimensions should be a good representation of the clusters in p-dimensional space if this ratio is large. Note that with $t = 3$, rotating 3-dimensional scatterplots can be used to see the projected points.

The low dimensional method we propose is similar to the method proposed by Dhillon et al. (2002) and to PCA. Instead of using the matrix $\boldsymbol{B}$ in the optimization problem (A.0.1), Dhillon et al. (2002) used the matrix

$$\boldsymbol{D} = \sum_{i=2}^{k_0}\sum_{j=1}^{i-1} n_i n_j(\hat{\boldsymbol{\theta}}_i - \hat{\boldsymbol{\theta}}_j)(\hat{\boldsymbol{\theta}}_i - \hat{\boldsymbol{\theta}}_j)^T,$$

while PCA uses the overall covariance matrix

$$\boldsymbol{C} = \frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{y}_i - \bar{\boldsymbol{y}})(\boldsymbol{y}_i - \bar{\boldsymbol{y}})^T,$$

where $\boldsymbol{y}_i$ is the i-the data point, $\bar{\boldsymbol{y}} = \sum_{i=1}^{n}\boldsymbol{y}_i/n$, n_i is the size of i-th cluster, and n is the total number of data points.

We use two examples to illustrate the performances of these three methods. The first example is the wine data set we used in Subsection 2.4.2. The 2-dimensional projections of the 3 clusters obtained by the clustering method `CLARA` are shown in Figure A.1. Data are scaled before applying `CLARA`. These three projections look similar and shows that cluster 2 is close to cluster 1 and separated from cluster 3 and that clusters 1 and 3 are well separated.

To compare the performances of these three projections, we calculate the separation index matrices for the projected three classes. Denote J_1, J_2, and J_3 as the separation index matrices for the projections obtained by our method, Dhillon et al.'s method, and PCA method respectively.

Then

$$J_1 = \begin{pmatrix} -1.00 & -0.02 & 0.33 \\ -0.02 & -1.00 & 0.09 \\ 0.33 & 0.09 & -1.00 \end{pmatrix}, J_2 = \begin{pmatrix} -1.00 & -0.01 & 0.33 \\ -0.01 & -1.00 & 0.08 \\ 0.33 & 0.08 & -1.00 \end{pmatrix},$$

$$J_3 = \begin{pmatrix} -1.00 & -0.06 & 0.31 \\ -0.06 & -1.00 & 0.07 \\ 0.31 & 0.07 & -1.00 \end{pmatrix}.$$

The minimum separation indexes of the three separation index matrices (-0.02, -0.01, and -0.06) indicate that the projected 3-cluster structure obtained by Dhillon et al. 's method are slightly more separated than those obtained by other two methods.

The second example is the Iris data set (Anderson 1935) which contains 3 clusters, each having 50 data points in a 4-dimensional space. The 2-dimensional projections of the 3 clusters obtained by **CLARA** are shown in Figure A.2.

The separation index matrices of the 3 projected cluster structures are given below:

$$J_1 = \begin{pmatrix} -1.00 & 0.59 & 0.41 \\ 0.59 & -1.00 & -0.14 \\ 0.41 & -0.14 & -1.00 \end{pmatrix}, J_2 = \begin{pmatrix} -1.00 & 0.58 & 0.41 \\ 0.58 & -1.00 & -0.15 \\ 0.41 & -0.15 & -1.00 \end{pmatrix},$$

$$J_3 = \begin{pmatrix} -1.00 & 0.57 & 0.41 \\ 0.57 & -1.00 & -0.16 \\ 0.41 & -0.16 & -1.00 \end{pmatrix}.$$

These three projections shows that cluster 1 is well separated from the other two clusters and that clusters 2 and 3 are close to each other. From the plots, we could not distinguish which projected cluster structure is more separated. The minimum separation indexes of the three separation index matrices (-0.14, -0.15, and -0.16) indicate that the projected 3-cluster structure obtained by our method are slightly more separated than those obtained by other two methods.

The two examples show that the three methods have the similar performance to visualize cluster structures. Note that

$$\sum_{i=1}^{k_0}\sum_{j=1}^{k_0} (\boldsymbol{y}_i - \boldsymbol{y}_j)(\boldsymbol{y}_i - \boldsymbol{y}_j)^T = 2\sum_{i=1}^{k_0} (\boldsymbol{y}_i - \bar{\boldsymbol{y}})(\boldsymbol{y}_i - \bar{\boldsymbol{y}})^T.$$

This explains why $\boldsymbol{B}$ and $\boldsymbol{C}$ lead to similar results. $\boldsymbol{D}$ is similar to $\boldsymbol{B}$ when clusters are separated enough so that $\boldsymbol{D}$ dominates $2\sum_{i=1}^{k_0} \boldsymbol{\Sigma}_i/k_0$.

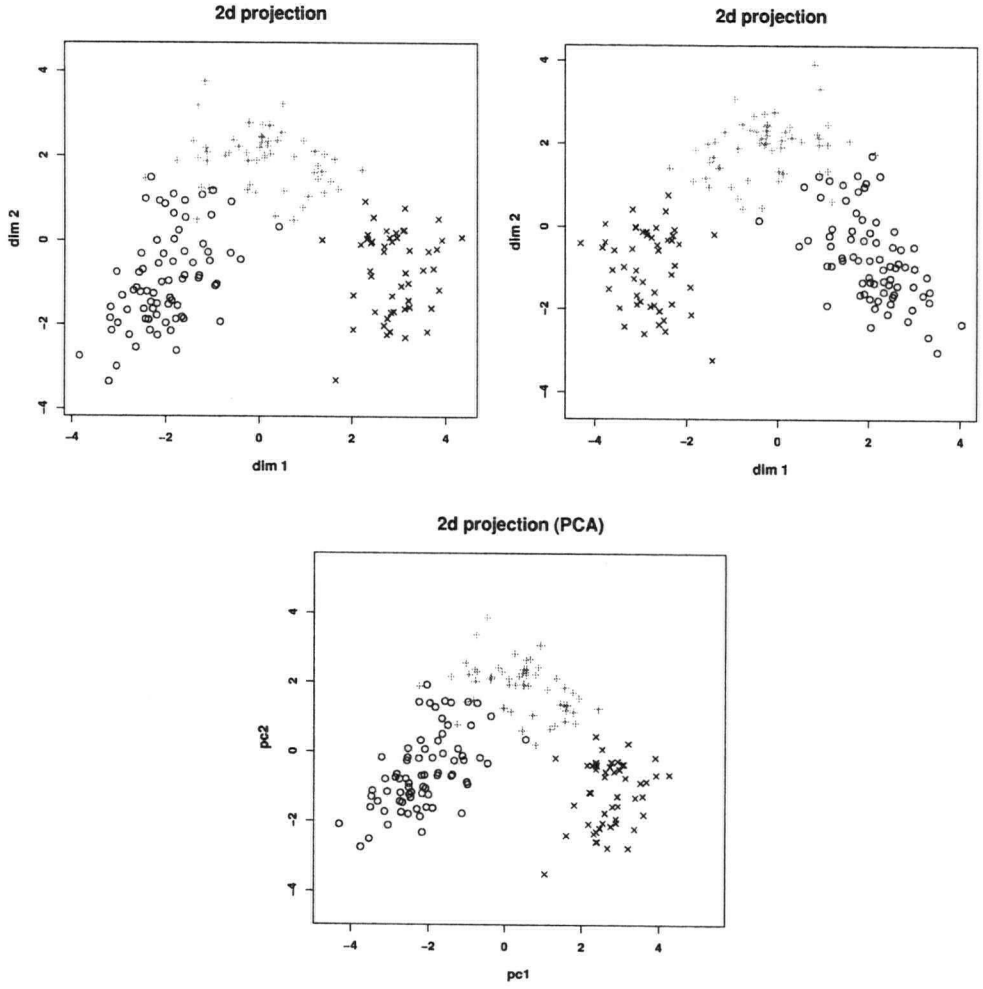


Figure A.1: A 2-dimension projection of the 3 clusters of the wine data. The circles represent points from cluster 1, the symbol "+"'s represent cluster 2, while the symbol "×"'s represent cluster 3. Top left: Using our visualization method. Top right: Using Dhillon et al. 's (2002) method. Bottom: Using PCA. The 3-cluster partition is obtained by **CLARA**.

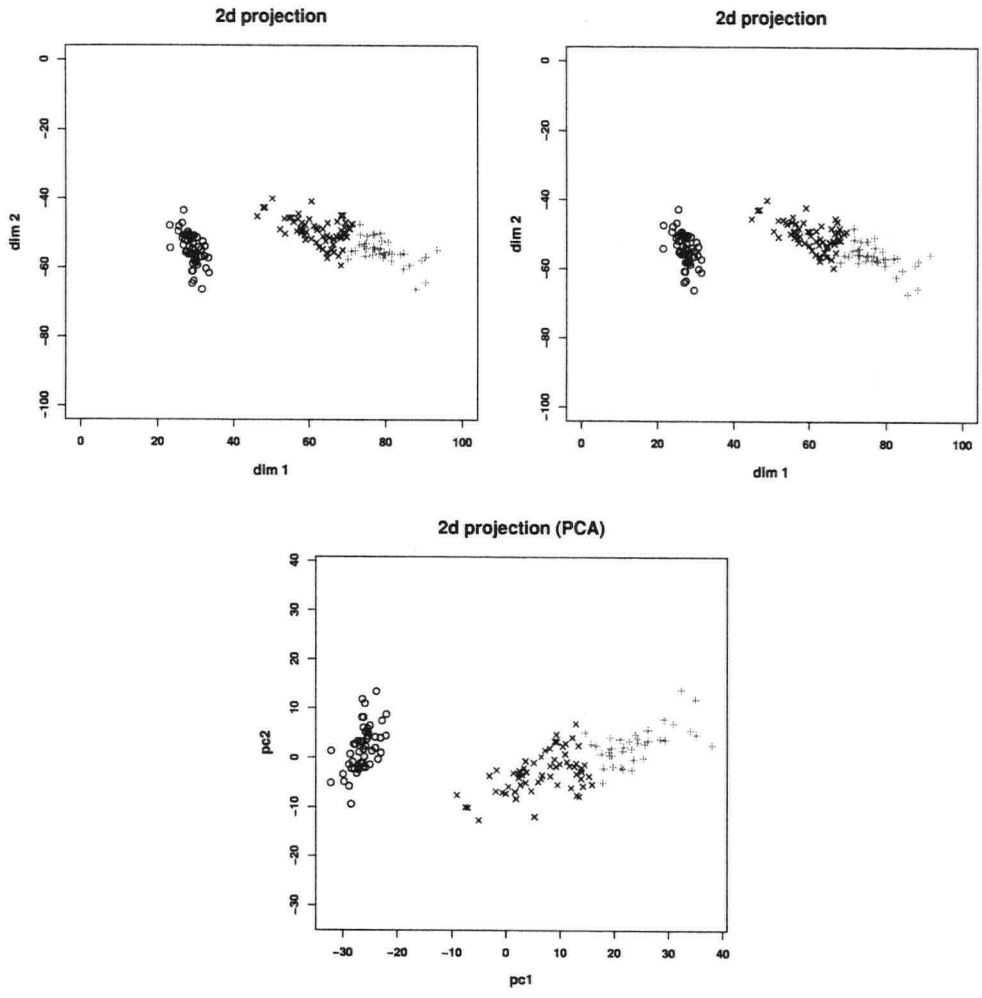


Figure A.2: A 2-dimension projection of the 3 clusters of the Iris data. The circles represent points from cluster 1, the symbol "+"'s represent cluster 2, while the symbol "×" represent cluster 3. Top left: using our visualization method; Top right: Using Dhillon et al. 's (2002) method. Bottom: Using PCA. The 3-cluster partition is obtained by `CLARA`.

Appendix B

Carman and Merickel's (1990) Implementation of the ISODATA Method

The CAIC criterion was proposed by Bozdogan (1987) . The general formula of the CAIC is

$$CAIC(t) = -2\log(L(\hat{\boldsymbol{\theta}}_t)) + t[\log(n) + 1],$$

where $L(\hat{\boldsymbol{\theta}}_t)$ is the likelihood function evaluated at the maximum likelihood estimates of the parameters $\boldsymbol{\theta}_t$, n is the total number of data points, and t is the number of free parameters.

The CAIC formula in Carman and Merickel (1990) is derived under the assumptions that data points in a cluster are from a multivariate normal distribution with diagonal covariance matrix and that all data points are independent to each other. Suppose that data points $\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{kn_k}$ are from the k-th cluster, $k = 1, \ldots, k_0$, where k_0 is the number of clusters. Then the two assumptions are equivalent to assume that $\boldsymbol{x}_{ki} \sim \mathrm{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\Sigma}_k$ is the $p \times p$ diagonal covariance matrix, p is the number of variables, and that $\boldsymbol{x}_{ki}$; $k = 1, \ldots, k_0$, $i = 1, \ldots, n_k$ are all independent to each other. And the likelihood function is

$$\begin{aligned} L(\boldsymbol{\theta}_t) &= \prod_{k=1}^{k_0} \prod_{i=1}^{n_k} \left\{ (2\pi)^{-p/2} \left|\boldsymbol{\Sigma}_k\right|^{-1/2} \exp\left[-\frac{(\boldsymbol{x}_{ki} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_{ki} - \boldsymbol{\mu}_k)}{2} \right] \right\} \\ &= \prod_{k=1}^{k_0} \prod_{i=1}^{n_k} \left\{ (2\pi)^{-p/2} \left|\boldsymbol{\Sigma}_k\right|^{-1/2} \exp\left[-\frac{\sum_{j=1}^{p} (x_{kij} - \mu_{kj})^2 \boldsymbol{\Sigma}_{kjj}^{-1}}{2} \right] \right\}, \end{aligned}$$

where $\boldsymbol{\theta}_t$ is the $2kp \times 1$ parameter vector containing the k_0p means μ_{kj} and the k_0p variances $\boldsymbol{\Sigma}_{kjj}$, $k = 1, \ldots, k_0$, $j = 1, \ldots, p$. Then

$$-2\log(L(\boldsymbol{\theta}_t)) = np\log(2\pi) + \sum_{k=1}^{k_0} n_k \log|\boldsymbol{\Sigma}_k| + \sum_{k=1}^{k_0}\sum_{j=1}^{p} \boldsymbol{\Sigma}_{kjj}^{-1} \sum_{i=1}^{n_k} (x_{kij} - \mu_{kj})^2.$$

When plugging in the maximum likelihood estimates of μ_{kij} and $\boldsymbol{\Sigma}_{kjj}$, $k = 1, \ldots, k_0$, $i = 1, \ldots, n_k$, and $j = 1, \ldots, p$, we can get

$$\begin{aligned} -2\log(L(\hat{\boldsymbol{\theta}}_t)) &= np\log(2\pi) + \textstyle\sum_{k=1}^{k_0} n_k \log|\hat{\boldsymbol{\Sigma}}_k| + np \\ &= np(1 + \log(2\pi)) + \textstyle\sum_{k=1}^{k_0} n_k \log|\hat{\boldsymbol{\Sigma}}_k|. \end{aligned}$$

Hence the CAIC in this special case is equal to

$$CAIC(2k_0p) = np(1 + \log(2\pi)) + \sum_{k=1}^{k_0} n_k \log|\hat{\boldsymbol{\Sigma}}_k| + 2k_0p[\log(n) + 1].$$

The last term of CAIC in Carman and Merickel (1990) is $2k_0p\log(n)$ instead of $2k_0p[\log(n) + 1]$.

Although Carman and Merickel (1990)'s improvement is appealing, the criterion CAIC seems not appropriate to determine if the merging or splitting is attempted. First of all, the change of the second term in the CAIC formula

$$T_2 = \sum_{k=1}^{k_0} n_k \log|\hat{\boldsymbol{\Sigma}}_k|$$

is usually significantly larger than that of the term $T_3 = 2k_0p[\log(n) + 1]$. And T_2 tends to be monotone decreasing as the number of clusters k_0 increases. This explains why Carman and Merickel (1990)'s improvement tends to overestimate the number of clusters. To illustrate this, we generate a cluster with 100 data points from the bivariate normal distribution $\mathrm{N}(\mathbf{0}, I_2)$, where $\mathbf{0} = (0, 0)^T$ and I_2 is the 2-dimensional identity matrix. The CAIC value for this cluster is 584.606 ($T_2 = -5.390$, $T_3 = 22.421$). If we split the cluster into two subclusters by using the Ward hierarchical clustering algorithm, then the CAIC value is 507.717 ($T_2 = -104.670$ and $T_3 = 44.841$). By this splitting criterion, we accept the attempt to split the cluster into two subclusters. Therefore, the Carman and Merickel (1990)'s improvement overestimates the number of clusters in this example.

Moreover, the diagonal covariance matrix is not common in real data sets. If we allow the covariance matrices to be non-diagonal, then the number of free parameters is $t = k_0p(p + 3)/2$ instead of $t = 2k_0p$. We recalculate the CAIC values in the previous example which are shown in Table B.1. The number of clusters is still overestimated.

Table B.1: CAIC values in a small example when we use the general formula of CAIC.

k_0	T_2	T_3	CAIC
1	-5.988	28.026	589.613
2	-105.113	56.0517	518.514

Bibliography

[1] Alimoglu, F. and Alpaydin, E. Methods of combining multiple classifiers based on different representations for pen-based handwriting recognition. In *Proceedings of the Fifth Turkish Artificial Intelligence and Artificial Neural Networks Symposium (TAINN 96)*. Istanbul, Turkey, 1996.

[2] Anderberg, M. R. *Cluster Analysis for Applications*. Academic Press, 1973.

[3] Anderson, E. The Irises of the gaspe peninsula. *Bulletin of the American Iris Society*, 59:2–5, 1935.

[4] Art, D., Gnanadesikan, R., and Kettenring, J. R. Data-based metrics for cluster analysis. *Utilitas Mathematica*, 21A:75–99, 1982.

[5] Bagirov, A. M., Ferguson, B., Ivkovic, S., Saunders, G., and Yearwood, J. New algorithms for multi-class cancer diagnosis using tumor gene expression signatures. *Bioinformatics*, 19(14):1800–1807, 2003.

[6] Ball, G. H. Classification analysis. Stanford Research Insitute, SRI Project 5533, 1971.

[7] Ball, G. H. and Hall, D. J. ISODATA, a novel method of data analysis and pattern classification. AD 699616. Stanford Res. Inst., Menlo Park, California, 1965.

[8] Beale, E. M. L., Kendall, M. G., and Mann, D. W. The discarding of variables in multivariate analysis. *Biometrika*, 54(3 and 4):357–366, 1967.

[9] Bezdek, J. C. Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1:57–71, 1974a.

[10] Bezdek, J. C. Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3:58–72, 1974b.

[11] Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981.

[12] Blake, C. L. and Merz, C. J. UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[13] Bozdogan, H. Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

[14] Bozdogan, H. Choosing the number of component clusters in the mixture-model using a new informational complexity criterion of the inverse-Fisher information matrix. In Opitz, O., Lausen, B., and Klar, R., editors, *Information and Classification*, pages 40–54. Springer, Heidelberg, 1993.

[15] Brusco, M. J. and Cradit, J. D. A variable-selection heuristic for k-means clustering. *Psychometrika*, 66(2):249–270, 2001.

[16] Calinski, R. B. and Harabasz, J. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

[17] Carbonetto, P., Freitas, N. D., Gustafson, P., and Thompson, N. Bayesian feature weighting for unsupervised learning with application to object recognition. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[18] Carman, C. S. and Merickel, M. B. Supervising ISODATA with an information theoretic stopping rule. *Pattern Recognition*, 23(12):185–197, 1990.

[19] Carmone, F., J. Jr., Alikara, A., and Maxwell, S. HINoV: A new model to improve market segment definition by identifying noisy variables. *Journal of Marketing Research*, XXXVI:501–509, 1999.

[20] Cheng, Y. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[21] Chiu, T., Fang, D., Chen, J., Wang, Y., and Jeris, C. A robust and scalable clustering algorithm for mixed type attributes in large database environment. In *Proceedings of the 7 th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 263–268, 2001.

[22] Comaniciu, D. and Meer, P. Mean shift analysis and applications. *Proc. Seventh Int'l Conf. Computer Vision*, I:1197–1203, 1999.

[23] Comaniciu, D. and Meer, P. Real-time tracking of non-rigid objects using mean shift. *Proc. 2000 IEEE Conf. Computer Vision and Pattern Recognition*, II:142–149, 2000.

[24] Comaniciu, D. and Meer, P. The variable bandwidth mean shift and data-driven scale selection. *Proc. Eighth Int'l Conf. Computer Vision*, I:438–445, 2001.

[25] Comaniciu, D. and Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[26] Cormack, R. M. A review of classification. *Journal of the Royal Statistical Society, Series A*, 134:321–353, 1971.

[27] De Soete, G. Optimal variable weighting for ultrametric and additive tree clustering. *Quality and Quantity*, 20:169–180, 1986.

[28] De Soete, G. OVWTRE: A program for optimal variable weighting for ultrametric and additive tree fitting. *Journal of Classification*, 5:101–104, 1988.

[29] De Soete, G., DeSarbo, W. S., and Carroll, J. D. Optimal variable weighting for hierarchical clustering: An alternating least squares approach. *Journal of Classification*, 2:173–192, 1985.

[30] Desarbo, W. S., Carroll, J. D., and Clark, L. A. Synthesized clustering: A method for amalgamating alternative clustering based with differential weighting of variables. *Psychometrika*, 49(1):57–78, 1984.

[31] Dhillon, I. S., Modha, D. S., and Spangler, W. S. Class visualization of high-dimensional data with applications. *Computational Statistics & Data Analysis*, 41(1):59–90, 2002.

[32] Ding, C. H. Q. Unsupervised feature selection via two-way ordering in gene expression analysis. *Bioinformatics*, 19(10):1259–1266, 2003.

[33] Donoghue, J. R. Univariate screening measures for cluster analysis. *Multivariate Behavioral Research*, 30(3):385–427, 1995.

[34] Dubes, R. C. How many clusters are best? - an experiment. *Pattern Recognition*, 20(6):645–663, 1987.

[35] Everitt, B. *Cluster Analysis.* Heinemann: London, 1974.

[36] Fang, K., Kotz, S., and Ng, K. *Symmetric Multivariate and Related Distributions.* Chapman & Hall, New York, 1990.

[37] Fowlkes, E. B., Gnanadesikan, R., and Kettenring, J. R. Variable selection in clustering. *Journal of Classification*, 5(2):205–228, 1988.

[38] Fraley, C. and Raftery, A. How many clusters?; Which clustering method? — Answers via model-based cluster analysis. *The Computer Journal*, 41(8):329–349, 1998.

[39] Fraley, C. and Raftery, A. E. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002a.

[40] Fraley, C. and Raftery, A. E. MCLUST: Software for model-based clustering, density estimation and discriminant analysis. Technical report, Department of Statistics, University of Washington, 2002b.

[41] Friedman, J. H. and Meulman, J. J. Clustering objects on subsets of attributes. Journal of the Royal Statistical Society. Series B. To appear, 2004.

[42] Frigui, H. and Krishnapuram, R. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7):1109–1119, 1997.

[43] Frigui, H. and Krishnapuram, R. A robust competitive clustering algorithm with applications in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):450–465, 1999.

[44] Fukunaga, K. and Hostetler, L. D. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. Information Theory*, 21:32–40, 1975.

[45] Gnanadesikan, R. *Methods for Statistical Data Analysis of Multivariate Observations.* Wiley, New York, 1977.

[46] Gnanadesikan, R., Kettenring, J. R., and Tsao, S. L. Weighting and selection of variables for cluster analysis. *Journal of Classification*, 12:113–136, 1995.

[47] Gordon, A. D. *Classification: Methods for the Exploratory Analysis of Multivariate Data.* Chapman and Hall, 1981.

[48] Gower, J. C. A general coefficient of similarity and some of its properties. *Biometrics*, 27:857–874, 1971.

[49] Green, P. E., Carmone, F. J., and Kim, J. A preliminary study of optimal variable weighting in k-means clustering. *Journal of Classification*, 7:271–285, 1990.

[50] Guha, S., Rastogi, R., and Shim, K. CURE: An efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 73–84, 1998.

[51] Guha, S., Rastogi, R., and Shim, K. A robust clustering algorithm for categorical attributes. In *Proceedings of the International Conference on Data Engineering (ICDE)*, pages 512–521, 1999.

[52] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. On clustering validation techniques. *Journal of Intelligent Information System*, 17:107–145, 2001.

[53] Hartigan, J., A. *Clustering Algorithms.* John Wiley & Sons, Inc., 1975.

[54] Hastie, T., Tibshirani, R., Eisen, M. B., Alizadeh, A., Levy, R., Staudt, L., Chan, W. C., Botstein, D., and Brown, P. 'gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, 1(2):research0003.1–0003.21, 2000.

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

[56] Höppner, K., Klawonn, F., and Runkler, T. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition.* Wiley, New York, 1999.

[57] Huang, K. Y. A synergistic automatic clustering technique (SYNERACT) for multispectral image analysis. *Photogrammetric Engineering & Remote Sensing*, 68(1):33–40, January 2002a.

[58] Huang, K. Y. The use of a newly developed algorithm of divisive hierarchical clustering for remote sensingimage analysis. *International Journal of Remote Sensing*, 23(16):3149–3168, 2002b.

[59] Huang, Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.

[60] Huber, P. J. Projection pursuit. *The Annals of Statistics*, 13(2):435–525, 1985.

[61] Hubert, L. and Arabie, P. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[62] Joe, H. *Multivariate Models and Dependence Concepts.* Chapman & Hall, 1997.

[63] Jolliffe, I. T. Discarding variables in a principal component analysis. I: Artificial data. *Applied Statistics*, 21(2):160–173, 1972.

[64] Jones, M. C. and Sibson, R. What is projection pursuit. *Journal of the Royal Statistical Society. Series A (General)*, 150(1):1–37, 1987.

[65] Kaufman, L. and Rousseeuw, P. J. *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley, New York., 1990.

[66] Kim, D-W, Lee, K. H., and Lee, D. Fuzzy cluster validation index based on inter-cluster proximity. *Pattern Recognition Letters*, 24(15):2561–2574, 2003.

[67] Kotz, S. and Johnson, N. L., editors. *Encyclopedia of Statistical Sciences.* Wiley, New York, 1983.

[68] Krishnapuram, R. and Freg, C. P. Fitting an unknown number of lines and planes to image data through compatible cluster merging. *Pattern Recognition*, 25(4):385–400, 1992.

[69] Kruskal, J. B. Linear transformation of multivariate data to reveal clustering. In Shepard, R. N., Romney, A. K., and Nerlove, S. B., editors, *Multidimensional Scaling*, pages 179–191. Seminar Press, 1972.

[70] Krzanowski, W. J. and Lai, Y. T. A criterion for determining the number of groups in a data set using sum of squares clustering. *Biometrics*, 44:23–34, 1988.

[71] Kundu, S. Gravitational clustering: A new approach based on the spatial distribution of the points. *Pattern Recognition*, 32:1149–1160, 1999.

[72] Li, W., Fan, M., and Xiong, M. SamCluster: An integrated scheme for automatic discovery of sample classes using gene expression profile. *Bioinformatics*, 19(7):811–817, 2003.

[73] Lin, C. R. and Chen, M. S. A robust and efficient clustering algorithm based on cohesion self-merging. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 582–587. Edmonton, Alberta, Canada, 2002.

[74] Liu, J. S., Zhang, J. L., Palumbo, M. J., and Lawrence, C. E. Bayesian clustering with variable and transformation selections. *Bayesian Statistics*, 7:249–275, 2003.

[75] Lumelsky, Vladimir, J. A combined algorithm for weighting the variables and clustering in the clustering problem. *Pattern Recognition*, 15(2):53–60, 1982.

[76] Makarenkov, V. and Legendre, P. Optimal variable weighting for ultrametric and additive trees and k-means partitioning: methods and software. *Journal of Classification*, 18:245–271, 2001.

[77] Milioli, M. A. Variable selection in fuzzy clustering. In Vichi, M. and Opitz, O., editors, *Classification and Data Analysis: Theory and Application*, pages 63–70. Springer-Verlag Berlin · Heidelberg, 1999.

[78] Miller, A. *Subset Selection in Regression.* Chapman & Hall/CRC, 2nd edition, 2002.

[79] Milligan, G. W. An examination of the effect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika*, 45(3):325–342, 1980.

[80] Milligan, G. W. A Monte Carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*, 46(2):187–199, 1981.

[81] Milligan, G. W. An algorithm for generating artificial test clusters. *Psychometrika*, 50(1):123–127, 1985.

[82] Milligan, G. W. and Cooper, M. C. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2):159–179, 1985.

[83] Milligan, G. W. and Cooper, M. C. A study of the comparability of external criteria for hierarchical cluster analysis. *Multivariate Behavioral Research*, 21:441–458, 1986.

[84] Milligan, G. W. and Cooper, M. C. A study of standardization of variables in cluster analysis. *Journal of Classification*, 5:181–204, 1988.

[85] Montanari, A. and Lizzani, L. A projection pursuit approach to variable selection. *Computational Statistics & Data Analysis*, 35:463–473, 2001.

[86] Peña, D. and Prieto, F. J. Cluster identification using projections. *Journal of the American Statistical Association*, 96(456):1433–1445, 2001.

[87] Peck, R., Fisher, L., and Van Ness, J. Approximate confidence intervals for the number of clusters. *Journal of the American Statistical Association*, 84(405):184–191, 1989.

[88] Rezaee, M. R. A new cluster validity index for the fuzzy c-mean. *Pattern Recognition Letter*, 19:237–246, 1998.

[89] Richardson, S. and Green, P. J. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society. Series B.*, 59(4):731–792, 1997.

[90] Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[91] Ruspini, E. H. Numerical methods for fuzzy clustering. *Inform. Sci.*, 2:319–350, 1970.

[92] Sato, Y. An autonomous clustering technique. In Kiers, A. L. H., Rasson, J.-P., Groenen, P. J. E., and Schader, M., editors, *Data Analysis, Classification, and Related Methods*. Springer, 2000.

[93] Schaffer, C. M. and Green, P. E. An empirical comparison of variable standardization methods in cluster analysis. *Multivariate Behavioral Research*, 31(2):149–167, 1996.

[94] Schlattmann, P. On bootstrapping the number of components in finite mixture models: The special case of homogeneity. Freie Universität Berlin, Berlin, Germany. Email: peter.schlattmann@medizin.fu berlin.de, 2002.

[95] Simpson, J. J., McIntire, T. J., and Sienkp, M. An improved hybrid clustering algorithm for natural scenes. *IEEE Transactions on Geoscience and Remote Sensing*, 38(2):1016–1032, 2000.

[96] Simpson, J. J., McIntire, T. J., Stitt, J. R., and Hufford, G. L. Improved cloud detection in AVHRR daytime and night-time scenes over the ocean. *International Journal of Remote Sensing*, 22(13):2585–2615, 2001.

[97] Stephens, M. Bayesian analysis of mixture models with an unknown number of components — an algernative to reversible jump methods. *Annals of Statistics*, 28(1):40–74, 2000.

[98] Sugar, C. A. *Techniques for Clustering and Classification with Applications to Medical Problems.* PhD thesis, Department of Statistics, Stanford University, 1998.

[99] Sugar, C. A. and James, G. M. Finding the number of clusters in a dataset: An information-theoretic approach. *Journal of the American Statistical Association*, 98(463):750–763, 2003.

[100] Sugar, C. A., Lenert, L., and Olshen, R. An application of cluster analysis to health services research: empirically defined health states for depression from the SF-12. Technical report, Department of Statistics, Stanford University, 1999.

[101] Tallis, G. M. Plane truncation in normal populations. *Journal of the Royal Statistical Society. Series B.*, 27(2):301–307, 1965.

[102] Tibshirani, R., Walther, G., and Hastie, T. Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society: Series B*, 63(2):411–423, 2001.

[103] Vapnik, V. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1996.

[104] Wang, J.-H. and Rau, J.-D. VQ-agglomeration: A novel approach to clustering. *IEE Proceedings-Vision, Image and Signal Processing*, 148(1):36–44, February 2001.

[105] Wang, S., Qiu, W.-L., and Zamar, R. H. An iterative non-parametric clustering algorithm based on local shrinking. unpublished manuscript, 2003.

[106] Wang, Song-Gui and Chow, Shein-Chung. *Advanced Linear Models: Theory and Applications.* Marcel Dekker, Inc., 1994.

[107] Wright, W. E. Gravitational clustering. *Pattern Recognition*, 9:151–166, 1977.

[108] Xie, X. L. and Beni, G. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847, 1991.

[109] Xing, E. P. and Karp, R. M. Cliff: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Bioinformatics*, 17, Suppl. 1:S306–S315, 2001.

[110] Zhang, T., Ramakrishnan, R., and Livny, M. A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.

[111] Zhuang, X., Huang, Y., Palaniappan, K., and Zhao, Y. Gaussian mixture density modeling, decomposition, and applications. *IEEE Transactions on Image Processing*, 5(9):1293–1302, 1996.