PROBABILISTIC DISTANCE CLUSTERING

BY CEM IYIGUN

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Operations Research

Written under the direction of

Professor Adi Ben–Israel

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

January, 2008

ABSTRACT OF THE DISSERTATION

Probabilistic Distance Clustering

by Cem Iyigun

Dissertation Director: Professor Adi Ben–Israel

We present a new iterative method for probabilistic clustering of data. Given clusters, their **centers**, and the **distances** of data points from these centers, the **probability** of cluster membership at any point is assumed inversely proportional to the distance from (the center of) the cluster in question. This assumption is our working **principle**.

The method is a generalization, to several centers, of the **Weiszfeld method** for solving the Fermat–Weber location problem. At each iteration, the distances (Euclidean, Mahalanobis, etc.) from the cluster centers are computed for all data points, and the centers are updated as convex combinations of these points, with weights determined by the above principle. Computations stop when the centers stop moving.

Progress is monitored by the **joint distance function (JDF)**, a measure of distance from all cluster centers, that evolves during the iterations, and captures the data in its low contours.

There are problems where the cluster sizes are given (as in capacitated facility location problems) and there are problems where the cluster sizes are unknowns to be estimated. The probabilistic distance clustering approach works well in both cases. The probabilistic distance clustering method adjusted for cluster size (called **PDQ method**) method is described, and applied to location problems, and mixtures of distributions, where it is a viable alternative to the **EM method**.

The method is simple, fast (requiring a small number of cheap iterations) and insensitive to outliers.

An important issue in clustering is the "right" number of clusters that best fits a data set. The JDF is used successfully to settle this issue and determine the correct number of clusters for a given data set.

Acknowledgements

I would like to thank my advisor Adi Ben–Israel for his patience and constant support in writing this thesis, for his kind advices, and for all that I learned from him. I will be forever grateful.

I am obliged to Professors Endre Boros, Andras Prekopa, Zachary Stoumbos, Marina Arav and W. Art Chaovalitwongse for being kind enough to participate in this thesis committee. I am thankful to the DIMACS (Discrete Mathematics and Theoretical Computer Science) Center, Rutgers University, for financial support which enabled much of the research presented here. Special thanks to Professor Kaan Ozbay from Civil Engineering, Rutgers University, who supported my research.

I am grateful to many people at RUTCOR for their help and kindness during my studies. Special thanks to Clare Smietana, Lynn Agre, Terry Hart, Tiberius Bonates, Marcin Kaminski, Gabriel Tavares, Konrad Borys, Diego Andrade, and Savas Bilican.

I would like to thank my Turkish friends, Ilker Dastan, Bekir Bartin, Ozlem Yanmaz–Tuzel, Baris E. Icin, Tugrul Diri, Ibrahim Bakir who made my stay here pleasurable while I was miles away from home.

Lastly, however mostly, I thank my family in Turkey, for their emotional support, and for bearing the distance and the limited contact that we have had over the last few years. They have always offered me so much and asked me for so little.

Dedication

I dedicate this thesis to my father, the medical doctor Professor Ibrahim Iyigun and my mother, teacher Ulker Iyigun; they have always offered me so much and asked me for so little. Their unlimited patience, kindness and endless support were invaluable for me. I express here my profound respect and gratitude to them. No one could ask for better parents!

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis presents a new approach to clustering, called probabilistic distance clustering, its algorithms, and selected applications. The thesis is divided into five parts.

Part I: Preliminaries

The present chapter contains a description of the thesis.

Chapter 2 gives a brief survey of the clustering concepts, notation and terminology that are relevant for this thesis. The approaches of **center–based clustering** and **hierarchical clustering** are compared, and the main algorithms are described briefly.

Part II: Probabilistic Distance Clustering

This part develops the models and algorithms for probabilistic clustering of data.

The main idea is presented in Chapter 3, with a new iterative method for probabilistic clustering of data. The method is based on a principle, or a model of the relationship between distances and probabilities. Given the clusters, their centers, and the distances of data points from these centers, the probability of cluster membership at any point is assumed inversely proportional to the distance from (the center of) the cluster in question. The cluster centers and cluster membership probabilities of the data points are updated using this principle. This is the basis of the **probabilistic distance clustering method** described in Section 3.4.

The progress of the algorithm is monitored by the **joint distance function (JDF)**, a measure of distance from all cluster centers, that evolves during the iterations, and

captures the data in its low contours, see Subsection 3.2.2. The proposed method is simple, fast (requiring a small number of cheap iterations) and insensitive to outliers. We also discuss various relations between probabilities and distances, resulting in different ways of clustering.

The algorithm presented in Section 3.4 takes no account of the cluster size. In cases where the cluster sizes differ greatly, or the cluster sizes themselves are unknowns that need to be estimated, the above algorithm can be modified to take into account the cluster sizes. This is done in Chapter 4, Section 4.3.

The probabilistic distance clustering adjusted for the cluster size is called here **PDQ Method**, where P stands for probability, D for distance and Q for cluster size in short.

Chapter 5 is about finding the "right" number of clusters that best fits a data set which is an important issue in clustering, called as **clustering validity**. The JDF, introduced in Chapter 3 is used successfully to settle this issue and determines the correct number of clusters for a given data set. This is illustrated in different examples, using simulated data sets. In the remainder of the chapter, we briefly survey other validity criteria used in the literature.

Part III: Related Problems

This part studies two problems which are closely related to distance clustering, and can be solved using the results of Part II with few modifications.

In Chapter 6, an important application of PDQ method in estimating the parameters of a **mixture of distributions** is presented. In such problems the cluster sizes are unknown and need to be estimated. We first describe the problem of mixtures of distributions and introduce the **EM Algorithm**, a well–known method for the solution of this type of problems. The PDQ method may serve as an alternative to that method, or as a preprocessor giving the EM Method a good start. We apply the algorithm to the estimation of the parameters of Gaussian mixtures, and compare it to the EM method. We conclude the chapter with the results of a number of computational experiments, comparing the running time and solution quality of our algorithm with the EM Method.

In Chapter 7, we present an iterative method for **multiple–facility location problems**, based on the PDQ method. The multiple facility location problem is to locate certain facilities so as to serve optimally a given set of customers, whose locations and demands are known. In some situations, there are upper bounds (capacities) on the demands that a facility can handle where the problem is called as capacitated multiple–facility location problem. The chapter starts with the Fermat–Weber Location problem, which is a single facility location problem and describe the **Weizsfeld Method**, the standard, best–known method for the solving Fermat–Weber problem. The probabilistic distance clustering in Chapter 3 is presented as a generalization to several facilities of the classical Weiszfeld Method. In the case where the facilities have the capacity constraints, the cluster size in the PDQ algorithm plays the role of facility capacity and the algorithm gives an approximate solutions to the capacitated multiple–facility location problem. The chapter ends with several numerical examples.

Part IV: Applications

This part is devoted to two applications, representing the diverse uses of probabilistic distance clustering.

In Chapter 8, we apply our method to clustering similarity data. Two examples of this type are considered and analyzed. The first example is the liberal–conservative clustering of the **Rehnquist Supreme Court**. The data used in the analysis is given as a similarity matrix, showing the percentages of non–unanimous decisions in which pairs of judges agreed with each other. The second example is from a political science study where pairwise dissimilarity measures between 12 countries are given.

Chapter 9 deals with determining the spatial clusters of accidents along a highway using different weights for the types of accidents. Identifying such spatial clusters of accidents can provide useful insights to various operational and safety issues. This study uses the New Jersey Turnpike (NJTPK) crash data sets for various years.

Part V: Semi–Supervised Clustering

Chapter 10 presents an approach to reconcile clustering (unsupervised learning) and classification (supervised learning, i.e. with prior information on the data.)

Chapter 2

Basics of Clustering

2.1 Introduction

Clustering can be defined as follows

> **Clustering** is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure. **Data clustering** is a common technique for statistical data analysis, which is used in many fields, including machine learning, data mining, pattern recognition, image analysis and bioinformatics. [1]

The ideas and methods of clustering are used in many areas, including **statistics** [56], **machine learning** [32], **data mining** [31], **operations research** ([16], [38]), **medical diagnostics**, **facility location**, and across multiple application areas including genetics, taxonomy, medicine, marketing, finance and e-commerce (see [12], [9], [35] and [52] for applications of clustering). It is therefore useful to begin by stating our notation and terminology. We then survey some of the methods, and results, that are relevant for our study.

2.2 Notation and Terminology

2.2.1 Data

The objects of clustering are **data points** (also **observations**, and in facility location, **customers**.) Each data point is an ordered list of **attributes** (or **features**), such as

height, weight, blood pressure, etc. Assuming p attributes, a data point $\mathbf{x}$ is thus a p–dimensional vector, $\mathbf{x} = (x_1, x_2, \ldots, x_p)$, with the attributes x_i for components.

The vector analogy cannot be carried too far, since in general vector operations (such as vector addition, scalar multiplication) do not apply to data points. Also, the attributes are not necessarily of the same algebraic type, some may be categorical, and some are reals. However, we can always imbed the data points in a p–dimensional real vector space $\mathbb{R}^p$, and for convenience we denote by $\mathbf{x} \in \mathbb{R}^p$ the fact that the data point $\mathbf{x}$ has p attributes.

We assume N data points $\mathbf{x}_i$, collected in a set

$$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^p \ , \tag{2.1}$$

called the **data set**. We sometimes represent $\mathcal{D}$ by an $N \times p$ matrix

$$D = (x_{ij}), \text{ where } x_{ij} \text{ is the j}^{th} \text{ component of the data point } \mathbf{x}_i. \tag{2.2}$$

2.2.2 The Problem

Given the data set $\mathcal{D}$, and integer K, $1 \leq K \leq N$, the **clustering problem** is to partition the data set $\mathcal{D}$ into K disjoint clusters

$$\mathcal{D} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \cdots \cup \mathcal{C}_K \ , \quad \text{with} \quad \mathcal{C}_j \cap \mathcal{C}_k = \emptyset \text{ if } j \neq k \ , \tag{2.3}$$

each cluster consisting of points that are **similar** (in some sense) and points of different clusters are dissimilar. We take here **similar** to mean **close** in the sense of **distances** $d(\mathbf{x}, \mathbf{y})$ between points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$. The **number of clusters** denoted by K is given, however there are problems where the "right" number of clusters (to fit best the data) is to be determined. The cases $K = 1$ (the whole $\mathcal{D}$ is one cluster) and $K = N$ (every point is a separate cluster) are included for completeness.

2.2.3 Cluster Membership

A clustering is **hard** (or **crisp**, **rigid**, **deterministic**) if each data point $\mathbf{x}$ is assigned to one, and only one, cluster $\mathcal{C}$, so that the statement $\mathbf{x} \in \mathcal{C}$ is unambiguous.

A point $\mathbf{x}$ is **labeled** if its cluster $\mathcal{C}$ is known, in which case $\mathcal{C}$ is the **label** of $\mathbf{x}$.

In **soft** (or **fuzzy**, **probabilistic**) **clustering** the rigid assignment $\mathbf{x} \in \mathcal{C}$ is replaced by a **cluster membership function** $u(\mathbf{x}, \mathcal{C})$ representing the **belief** that x belongs to C. The numbers $u(\mathbf{x}, \mathcal{C}_k)$ are often taken as probabilities that $\mathbf{x}$ belongs to $\mathcal{C}_k$, so that

$$\sum_{k=1}^{K} u(\mathbf{x}, \mathcal{C}_k) = 1 \ , \text{ and } u(\mathbf{x}, \mathcal{C}_k) \geq 0 \text{ for all } k = 1, \cdots, K \ . \tag{2.4}$$

2.2.4 Classification

In **classification**, or **supervised learning**, the number K of clusters is given, and a certain subset $\mathcal{T}$ of the data set $\mathcal{D}$ is given as labeled, i.e. for each point $\mathbf{x} \in \mathcal{T}$ it is known to which cluster it belongs. The subset $\mathcal{T}$ is called the **training set**.

The information obtained from the training set, is then used to find a rule of classifying the remaining data $\mathcal{D} \setminus \mathcal{T}$ (called the **testing set**), and any future data of the same type, to the K clusters. The **classification rule** r is a function from $\mathcal{D}$ (and by extension $\mathbb{R}^p$) to the integers $\{1, 2, \cdots, K\}$, so that

$$r(\mathbf{x}) = k \Longleftrightarrow \mathbf{x} \in \mathcal{C}_k \ .$$

In analogy, clustering is called **unsupervised learning** to emphasize the absence of prior information.

2.2.5 Distances

Assuming a norm $\| \cdot \|$ on the space $\mathbb{R}^p$, a distance between two points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ is defined by

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| \ , \tag{2.5}$$

for example, the Euclidean norm gives the distance between $\mathbf{x} = (x_1, \ldots, x_p)$ and $\mathbf{y} = (y_1, \ldots, y_p)$ as

$$d(\mathbf{x}, \mathbf{y}) := (\sum_{j=1}^{p} (x_j - y_j)^2)^{1/2} \ , \text{ the } \textbf{Euclideam distance} \ , \tag{2.6}$$

and the ℓ_1–norm gives

$$d(\mathbf{x}, \mathbf{y}) := \sum_{j=1}^{p} |x_j - y_j| \ , \text{ the } \ell_1\textbf{–distance} \ , \tag{2.7}$$

also called the **Manhattan** or **taxicab distance**.

The **standard inner product** of $\mathbf{x} = (x_1, \ldots, x_p)$ and $\mathbf{y} = (y_1, \ldots, y_p)$ is defined by

$$\langle \mathbf{x}, \mathbf{y} \rangle \ = \sum_{i=1}^{p} x_i \, y_i \ . \tag{2.8}$$

If Q is a positive–definite $p \times p$ matrix, then $\sqrt{\langle \mathbf{x}, Q\mathbf{x} \rangle}$ is a norm on $\mathbb{R}^p$, and the corresponding distance is

$$d(\mathbf{x}, \mathbf{y}) := \sqrt{\langle \mathbf{x} - \mathbf{y}, Q(\mathbf{x} - \mathbf{y}) \rangle} \ , \text{ an } \textbf{elliptic distance}, \tag{2.9}$$

depending on the choice of Q. For $Q = I$, the identity matrix, (2.9) gives the Euclidean distance (2.6). Another common choice is $Q = \Sigma^{-1}$, where Σ is the covariance matrix of the data in question, in which case (2.9) gives

$$d(\mathbf{x}, \mathbf{y}) := \langle \mathbf{x} - \mathbf{y}, \Sigma^{-1}(\mathbf{x} - \mathbf{y}) \rangle \ , \text{ the } \textbf{Mahalanobis distance}, \tag{2.10}$$

that is used commonly in multivariate statistics.

Distances associated with norms satisfy the **triangle inequality**,

$$d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y}) \ , \text{ for all } \mathbf{x}, \mathbf{y}, \mathbf{z} \ . \tag{2.11}$$

However distance functions violating (2.11) are also used.

2.2.6 Similarity Data

Given a distance function $d(\cdot,\cdot)$ in $\mathbb{R}^p$, and the data set $\mathcal{D}$, the **similarity** (or **proximity**) **matrix** of the data is the $N \times N$ matrix

$$S = (d_{ij}) \ , \ \text{where } d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j), \ i,j = 1,\ldots,N. \tag{2.12}$$

It is sometimes convenient to work with the **dissimilarity matrix**,

$$N = (g(d_{ij})) \ , \ \text{where } g(\cdot) \text{ is a decreasing function.} \tag{2.13}$$

2.2.7 Representatives of Clusters

In many clustering methods a cluster is represented by a typical point, called its **center** (also **representative**, **prototype**, and in facility location, **facility**.) A common choice for the center is the **centroid** of the points in the cluster. In general the center does not fall on any of the data points in the cluster.[1]

The center of the kth cluster $\mathcal{C}_k$ is denoted by $\mathbf{c}_k$, and the distance $d(\mathbf{x}, \mathcal{C}_k)$ of a point $\mathbf{x}$ from that cluster is defined as its distance from the center $\mathbf{c}_k$,

$$d(\mathbf{x}, \mathcal{C}_k) := d(\mathbf{x}, \mathbf{c}_k) \ , \tag{2.14}$$

and denoted $d_k(\mathbf{x})$ if the center is understood.

2.3 Objective Based Clustering

Sometimes the "goodness" of clustering can be expressed by an **objective function** of the given data $\mathcal{D}$ and the clusters $\{\mathcal{C}_1,\ldots,\mathcal{C}_K\}$. For example,

$$f(\mathcal{D}, \{\mathcal{C}_1,\ldots,\mathcal{C}_K\}) = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} d(\mathbf{x}_i, \mathbf{c}_k) \tag{2.15}$$

[1] In facility location problems, such a case requires special analysis.

is the sum of distances of data points to the centers of their respective clusters, while

$$f(\mathcal{D}, \{\mathcal{C}_1, \ldots, \mathcal{C}_K\}) = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} d(\mathbf{x}_i, \mathbf{c}_k)^2 \tag{2.16}$$

is the sum of squares of these distances. Both of these objectives are in use, and we call them the d–**model**, and the d^2–**model**, respectively. See section 3.2.7.

In such cases, clustering reduces to an **optimization problem**, that without loss of generality, is considered a **minimization problem**,

$$\min_{\mathcal{C}_1,\ldots,\mathcal{C}_K} f(\mathcal{D}, \{\mathcal{C}_1, \ldots, \mathcal{C}_K\}) \ , \tag{2.17}$$

that is often hard (combinatorial, non–smooth), but approximate solutions of (2.17) may be acceptable.

2.4 Center–Based Clustering Methods

Center–based clustering algorithms construct the clusters using the distances of data points from the cluster centers.

The best–known and most commonly used center–based algorithm is the k–**means algorithm** ([63], [39]) which (implicitly) minimizes the objective

$$\sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{x}_i - \mathbf{c}_k\|^2 \tag{2.18}$$

where $\mathbf{c}_k$ is the centroid of the k^{th} cluster. Other names like hard k–means, ISODATA ([7], [8]), etc. have also been used in the literature.

Algorithm 2.1. k-means Clustering Algorithm

Step 0	**Initialization**: Given data set $\mathcal{D}$, integer K, $2 \leq K < N$, **select** K initial centers $\{\mathbf{c}_k\}$
Step 1	**compute** the distances $d(\mathbf{x}_i, \mathbf{c}_k)$, $i = 1, \ldots, N,\ k = 1, \ldots, K$.
Step 2	**partition** the data set $\mathcal{D} = \mathcal{C}_1 \cup \mathcal{C}_2 \cup \cdots \cup \mathcal{C}_K$ by assigning each data point to the cluster whose center is the nearest
Step 3	**re–compute** the cluster centers.
Step 4	**if** the centers have not changed, **stop**. **else go to** Step 1.

Notes:

(a) The initial "centers" in Step 0 are just points, and not yet associated with clusters. They can be selected randomly as any K points of $\mathcal{D}$.

(b) In Step 3 the center of each cluster is computed using the points assigned to that cluster.

(c) The stopping rule in Step 4 implies that there are no further re–assignments.

(d) The center updates in the iterations are computed by

$$\mathbf{c}_k = \frac{\sum_{i=1}^{N} u_{ik}\, \mathbf{x}_i}{\sum_{i=1}^{N} u_{ik}}, \quad k = 1, \ldots, K \tag{2.19}$$

where $u_{ik} = 1$ if $\mathbf{x}_i \in \mathcal{C}_k$, and $u_{ik} = 0$ otherwise. Equation (2.19) gives the centers as the geometrical centroids of the data points of the cluster.

(e) Using Euclidean distances, iterating Steps 2 and 3 leads to the minimization of the objective (2.18).

2.4.1 Variants of the k–means Algorithm

Several variants of k–means algorithm have been reported in the literature ([33], [4]). Some of them attempt to select a good initial partition so that the algorithm is more likely to find the global minimum value [27]. An important variant of the algorithm is to permit splitting and merging of the resulting clusters (see [7]) in Step 2 of the algorithm.

Some variants of the algorithm use different criteria. Diday [24] used different representatives of the clusters (other than the cluster centers), and the Mahalanobis distance is used instead of the Euclidean distance in [61], [18] and elsewhere.

The k**-modes algorithm** [44] is a recent center–based algorithm for categorical data. Another variant, the k**-prototypes algorithm** [44], incorporates real and categorical data.

2.4.2 Fuzzy k–means

The k–means algorithm can be adapted to soft clustering, see section 2.2.3. A well–known center–based algorithm for soft clustering is the **Fuzzy k–means algorithm**, ([15], [72]).

The objective function minimized in this algorithm is:

$$f = \sum_{i=1}^{N}\sum_{k=1}^{K} u_{ik}^{m}\, d_{ik}^{2} \;=\; \sum_{i=1}^{N}\sum_{k=1}^{K} u_{ik}^{m}\, \|\mathbf{x}_i - \mathbf{v}_k\|^2$$

where u_{ik} are the membership functions of $\mathbf{x}_i \in \mathcal{C}_k$, and typically satisfy (2.4), and m is a real number, $m > 1$, known as **fuzzifier**.

The equation for finding the centers is similar to equation (2.19) of k–means algorithm, but u_{ik} takes values between 0 and 1.

$$\mathbf{c}_k = \frac{\sum_{i=1}^{N} u_{ik}^{m}\, \mathbf{x}_i}{\sum_{i=1}^{N} u_{ik}^{m}}\;,\; k = 1, \ldots, K. \tag{2.20}$$

When m tends to 1, the algorithm converges to the k–means method.

2.4.3 Probabilistic Methods

The title refers to data sets whose points come from a known statistical distribution, whose parameters have to be estimated. Specifically, the data may come from a mixture of several distributions and the weights of the distributions in the mixture, and their parameters, have to be determined.

The best–known probabilistic method is the **Expectation-Maximization (EM) algorithm** [62] where log–likelihood of the data points drawn from a given mixture model. The underlying probability model and its parameters determine the membership function of the data points. The algorithm starts with initial guesses for the mixture model parameters. These values are then used to calculate the cluster membership functions for the data points. In turn, these membership functions are used to re–estimate the parameters, and the process is repeated, see section 6.2.

Probabilistic methods depend critically on their assumed priors. If the assumption are correct, one gets good results. A drawback of these algorithms is that they are computationally expensive. Another problem found in this approach is called the **overfitting**, see [40].

2.5 Hierarchical Clustering Algorithms

Hierarchical clustering algorithms are an important class of clustering methods that are not center–based, but instead use similarity data.

These algorithms transform a similarity data set into a tree–like structure which is called a **dendogram** [53]. The dendogram is constructed as a sequence of partitions such that its root is a cluster covering all the points and the leaves are clusters containing only one point. In the middle, child clusters partition the points assigned to their common parent according to a dissimilarity level. This is illustrated in Figure 2.1 (Note that the dendogram is not a binary tree.) The dendogram is most useful up to a few levels deep, as the clustering becomes more trivial as the tree depth increases. Hierarchical clustering methods are categorized into two major methods as **agglomerative** and **divisive** methods ([52] & [56]).

Agglomerative clustering is a bottom–up way of constructing the dendogram. The hierarchical structure begins with N clusters, one per point, and grows a sequence of clusterings until all N observations are in a single cluster. *Divisive* clustering on the other hand is a top–down way of constructing the dendogram. The structure begins with one cluster containing all N points and successively divides clusters until N clusters

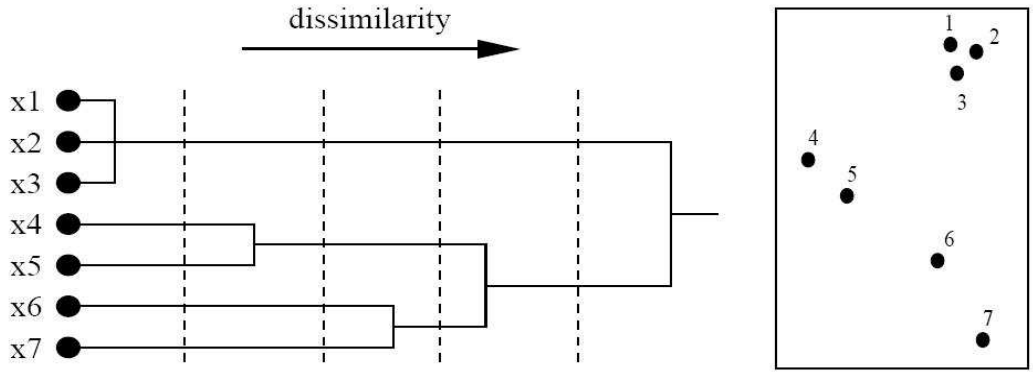


Figure 2.1: An example of the dendogram that might be produced by a hierarchical algorithm from the data shown on the right. The dotted lines indicate different partitions at different levels of dissimilarity.

are achieved.

Agglomerative hierarchical clustering is computationally less complex and, for this reason, it is more commonly used than divisive hierarchical clustering. For agglomerative hierarchical techniques, the criterion is typically to merge the "closest" pair of clusters, where "close" is defined by a specified measure of cluster proximity. There are three definitions of the closeness between two clusters: **single-link, complete-link** and **average–link**. The *single–link* similarity between two clusters is the similarity between the two most similar instances, one of which appears in each cluster. Single link is good at handling non–elliptical shapes, but is sensitive to noise and outliers. The *complete–link* similarity is the similarity between the two most dissimilar instances, one from each cluster. Complete link is less susceptible to noise and outliers, but can break large clusters, and has trouble with convex shapes. The *average–link* similarity is a compromise between the two.

The advantages of agglomerative and divisive algorithms are: (i) they do not require the number of clusters to be known in advance, (ii) they compute a complete hierarchy of clusters, (iii) good result visualizations are integrated into the methods, and (iv) a "flat" partition can be derived afterwards (using a cut through the dendrogram). However, both methods suffer from their inability to perform adjustments once the splitting or merging decision is made.

In both methods if, say, at one point during the construction of the dendogram, a

misclassification is made, it is built on until the end of the process. At some point of the dendograms growth an observation may be designated as belonging to a cluster in the hierarchy. It remains associated with the successors of that cluster till the dendogram is finished. It is impossible to correct this misclassification while the clustering process is still on.

After the tree has been produced, a multitude of possible clustering interpretations are available. A practical problem with hierarchical clustering, thus, is: at which value of dissimilarity should the dendogram be cut, or in other words, at which level should the tree be cut. One heuristic commonly used is to choose that value of dissimilarity where there is a large "gap" in the dendogram. This assumes that a cluster that merges at a much higher value of dissimilarity than that at which it was formed is more "meaningful". However, this heuristic does not work all the time [51].

Some of the hierarchical clustering algorithms recently presented in the literature are: Balanced Iterative Reducing and Clustering using Hierarchies - BIRCH [86], Clustering Using Representatives - CURE [36], and CHAMELEON [55]. More recently, a novel incremental hierarchial clustering algorithm (GRIN) for numerical data sets is presented in [17]. A survey and comparison of these algorithms are in [57] and [11].

2.6 Dispersion Statistics

The partitioning (2.3) of the data points $\mathbf{x}_i$ (which are the rows of the $N \times p$ data matrix D of (2.2)), gives rise to the $p \times p$ **total dispersion matrix**,

$$T = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} (\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})', \tag{2.21}$$

where the p–dimensional vector $\overline{\mathbf{x}}$ is the mean of all the data points. The total dispersion matrix T can be partitioned as

$$T = W + B \tag{2.22}$$

where W is the **within–cluster dispersion matrix**,

$$W = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} (\mathbf{x}_{ik} - \overline{\mathbf{x}}_k)(\mathbf{x}_{ik} - \overline{\mathbf{x}}_k) \tag{2.23}$$

here $\overline{\mathbf{x}}_k$ is the mean of the data points in the cluster $\mathcal{C}_k$, and B is the **between–clusters dispersion matrix**,

$$B = \sum_{k=1}^{K} N_k (\overline{\mathbf{x}}_k - \overline{\mathbf{x}})(\overline{\mathbf{x}}_k - \overline{\mathbf{x}})' \,, \tag{2.24}$$

where N_k is the number of data points in $\mathcal{C}_k$.

For univariate data ($p = 1$), equation (2.22) represents the division of the total sum of squares of a variable into the within- and between-clusters sum of squares. In the univariate case a natural criterion for grouping would be to choose the partition corresponding to the minimum value of the within-group sum of squares or, equivalently, the maximum value of the between-cluster sum of squares.

In the multivariate case ($p > 1$) the derivation of a clustering criterion from the equation (2.22) is not so clear-cut as the univariate case, and several alternatives have been suggested.

2.7 Dispersion Objectives

The dispersion statistics of section 2.6 suggest several different objectives for clustering.

2.7.1 Minimization of trace (W)

The trace of the matrix W in (2.23) is the sum of the within–cluster variances. Minimizing this trace works to make the clusters more homogeneous, thus the problem,

$$\min\{\operatorname{trace} W\}, \tag{2.25}$$

which, by (2.22) is equivalent to

$$\max\{\operatorname{trace} B\}. \tag{2.26}$$

This can be shown to be equivalent to minimizing the sum of the squared Euclidean distances between data points and their cluster mean which is used in k-means algorithms. The criterion can also be derived on the basis of the distance matrix:

$$E = \sum_{k=1}^{K} \frac{1}{2N_k} \sum_{i=1}^{N_k} \sum_{j=1, j\neq i}^{N_k} d_{ij}^2, \tag{2.27}$$

where d_{ij} is the Euclidean distance between ith and jth data points in cluster $\mathcal{C}_k$. Thus the minimization of $trace(\mathbf{W})$ is equivalent to the minimization of the homogeneity criterion $h_1(\mathcal{C}_k)/N_k$ for Euclidean distances and $n = 2$ [30].

2.7.2 Minimization of $\det(W)$

The differences in cluster mean vectors are based on the ratio of the determinants of the total and within-cluster dispersion matrices. Large values of $\det(T)/\det(W)$ indicate that the cluster mean vectors differ. Thus, a clustering criterion can be constructed as the maximization of this ratio;

$$\min\left\{\frac{\det(T)}{\det(W)}\right\}. \tag{2.28}$$

Since T is the same for all partitions of N data points into K clusters, this problem is equivalent to

$$\min\{\det(W)\}. \tag{2.29}$$

2.7.3 Maximization of trace(BW^{-1})

A further criterion considered is a combination of dispersion matrices:

$$\max\{\text{trace}\,(\frac{B}{W})\}. \tag{2.30}$$

This criterion is obtained from the product of the between-clusters dispersion matrix and the inverse of the within-clusters dispersion matrix. This function is also a further test criterion used in the context of multivariate analysis of variance, with large values

of trace (BW^{-1}) indicating that the cluster mean vectors differ.

2.7.4 Comparison of the Clustering Criteria

Of the three clustering criteria mentioned above, the criterion (2.30) is perhaps the one most commonly used. However it suffers from some serious problems [30]. Firstly, the method is not scale–invariant. Different solutions may be obtained from the raw data and from the data standardized in some way. Clearly this is of considerable practical importance because of the need for standardization in many applications. Another problem with the use of this criterion is that it may impose a *spherical* structure on the observed clusters even when the natural clusters in the data are of other shapes. The alternative criteria in equations (2.25) and (2.30) are not affected by scaling which is the main motivation behind of these criteria. Moreover, the criterion in equation (2.29) which has been widely used does not restrict clusters to being spherical. It can also identify elliptical clusters. On the other hand, this criteria assumes that all clusters in the data have the same shape i.e. the same orientation. Finally, both the criteria in equations (2.25) and (2.29) produce clusters that contain roughly equal numbers of data points.

2.8 Other Clustering Methods

In this section, we briefly describe other clustering methods developed in the data clustering area. For comprehensive explanations and further details, see the cited references.

2.8.1 Density–based Clustering

Density–based methods consider that clusters are dense sets of data points separated by less dense regions; clusters may have arbitrary shape and data points can be arbitrarily distributed. Many methods, such as DBSCAN [29] (further improved in [57]), rely on the study of the density of points in the neighborhood of each point.

One can consider within the category of density-based methods the **grid–based**

solutions, such as DENCLUE [42] or CLIQUE [3], mostly developed for spatial data mining. These methods quantize the space of the data points into a finite number of cells (attention is shifted from data points to space partitioning) and only retain for further processing the cells having a high density of points; isolated data points are thus ignored. Quantization steps and density thresholds are common parameters for these methods.

2.8.2 Graph–Theoretic Clustering

Another clustering method is the **graph–theoretic clustering** method where the data points are represented as nodes in a graph and the dissimilarity between two points is the "length" of the edge between the corresponding nodes. In several methods, a cluster is a subgraph that remains connected after the removal of the longest edges of the graph [52]; for example, in [85] (the best–known graph–theoretic clustering algorithm) the minimal spanning tree of the original graph is built and then the longest edges are deleted. Some other graph–theoretic methods rely on the extraction of **cliques** and are then more related to center–based methods, see [66].

2.8.3 Volume Based Clustering

To overcome the difficulties like clustering with equal size or spherical shapes, we can use Mahalanobis distances (see section 2.2.5) instead of Euclidean distance [76]. For example, if the covariance Σ is known, then the similarity within that cluster, with center $\mathbf{c}$ would be measured by $\|\mathbf{x} - \mathbf{c}\|_{\Sigma^{-1}}$. This measure is scale invariant and can deal with asymmetric, non-spherical clusters. A difficulty in using Mahalanobis distances is getting a good estimate of the covariance matrices in question.

A promising alternative scale–invariant metric of cluster quality is **minimum volume ellipsoids**, where data points are allocated into clusters so that the volumes of the covering ellipsoids for each cluster is minimized. The problem of finding the minimum volume ellipsoid can be formulated as a **semidefinite programming** problem and an efficient algorithm for solving the problem has been proposed by [77].

2.9 Support Vector Machines

Support vector machines (SVMs) are a set of related supervised learning methods (see section 2.2.4) used for classification and regression [2]. Support Vector Machines (SVMs) is to find an optimal plan that separates data into two groups, say $\mathcal{X}$ and $\mathcal{Y}$. The optimal plane is first obtained from training data that has been labeled, which means we know which group each entity comes from. Then the plane can be used for classifying new observations. All entities from $\mathcal{X}$ and $\mathcal{Y}$ will be separated by the plan under the assumption that $\mathcal{X}$ and $\mathcal{Y}$ are separable. This assumption can be achieved if there exits a proper *kernel function* that projects all entities from $\mathcal{X}$ and $\mathcal{Y}$ into a high dimensional space. The projection into sufficiently high dimensional space will lead to a separable data set. A set of data of two groups may have many possible separating plans. However, there is one optimal SVM hyperplane for a data set.

The support vector machine algorithm can be interpreted as the construction of a linear classifier in a very high-dimensional space (called the feature space), obtained by transformation of the original input space.

The key ingredient of the algorithm is a kernel function that allows the training phase and the classification of new observations to be carried out in the feature space without the need to actually perform the transforming computations.

The typical support vector classifier (for two-class problems) consists of a linear discriminant function that separates the training data. A quadratic optimization model is used to optimize the weights, so that the margin of separation between the two classes if maximized. The margin of separation is simply the smallest distance from a point in one of the classes to the separating hyperplane, plus the smallest distance from a point in the other class to the separating hyperplane.

The formulation of the underlying optimization model is such that the only information required about the feature space utilized is the inner product between every pair of (transformed) observations in the training data set. The kernel function is chosen in such a way that it provides, with low computational costs, the inner product between two observations mapped into the feature space. Clearly, one is interested in choosing

a feature space in which a better separation of the two classes is possible than that obtained in the input space.

In practice, the optimization model takes into account a penalty term, in order to allow some observations in the training data set to be incorrectly classified. The so–called ψ-parameter dictates how much importance the model should give to the perfect separation of the training data, as opposed to the maximization of the margin of separation of "most" observations. The value of ψ is a critical parameter in tuning the support vector machines algorithm.

Another important parameter of the algorithm is the kernel function used, or in other words the feature space chosen. Many different kernel functions have been proposed for specific types of data. Among the general–purpose kernel functions frequently used we cite the polynomial and radial basis function kernels.

A very similar variant of the optimization model utilized for training allows the use of the same algorithm for regression tasks, resulting in the so–called support vector regression algorithm. For a comprehensive treatment of support vector machines, the reader is referred to [73].

Chapter 3

Probabilistic Distance Clustering

3.1 Introduction

A **cluster** is a set of data points that are similar, in some sense, and **clustering** is a process of partitioning a data set into disjoint clusters.

We take data points to be vectors $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^p$, and interpret "similar" as "close", in terms of a **distance** function $d(\mathbf{x}, \mathbf{y})$ in $\mathbb{R}^p$, such as

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|, \ \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^p, \tag{3.1}$$

where the norm $\|\cdot\|$ is **elliptic**, defined for $\mathbf{u} = (u_i)$ by

$$\|\mathbf{u}\| = \langle \mathbf{u}, Q\mathbf{u} \rangle^{1/2}, \tag{3.2}$$

with $\langle \cdot, \cdot \rangle$ the standard inner product, and Q a positive definite matrix. In particular, $Q = I$ gives the **Euclidean norm**,

$$\|\mathbf{u}\| = \langle \mathbf{u}, \mathbf{u} \rangle^{1/2}, \tag{3.3}$$

and the **Mahalanobis distance** corresponds to $Q = \Sigma^{-1}$, where Σ is the covariance matrix of the data involved.

Example 3.1. A data set in $\mathbb{R}^2$ with $N = 200$ data points is shown in Figure 3.1. The data was simulated, from normal distributions $N(\boldsymbol{\mu_i}, \Sigma_i)$, with:

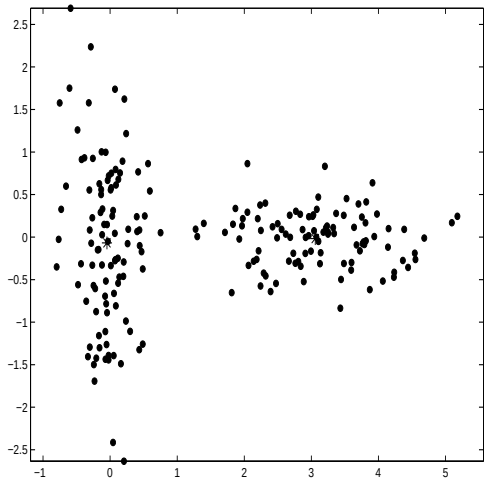


Figure 3.1: A data set in $\mathbb{R}^2$

$$\boldsymbol{\mu}_1 = (0,0),\ \Sigma_1 = \begin{pmatrix} 0.1 & 0 \\ 0 & 1 \end{pmatrix},\ (100 \text{ points}),$$

$$\boldsymbol{\mu}_2 = (3,0),\ \Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 0.1 \end{pmatrix},\ (100 \text{ points}).$$

This data will serve to illustrate Examples 3.2–3.5 below.

The **clustering problem** is, given a **dataset** $\mathcal{D}$ consisting of N data points

$$\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^p,$$

and an integer K, $1 < K < N$, to partition $\mathcal{D}$ into K **clusters** $\mathcal{C}_1, \ldots, \mathcal{C}_K$.

Data points are assigned to clusters using a **clustering criterion**. In **distance clustering**, abbreviated **d–clustering**, the clustering criterion is metric: With each cluster $\mathcal{C}_k$ we associate a **center** $\mathbf{c}_k$, for example its centroid, and each data point is assigned to the cluster to whose center it is the nearest. After each such assignment, the cluster centers may change, resulting in re–assignments. Such an algorithm will therefore iterate between updating the centers and re–assignments.

A commonly used clustering criterion is the sum–of–squares of Euclidean distances,

$$\sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \| \mathbf{x}_i - \mathbf{c}_k \|^2, \tag{3.4}$$

to be minimized by the sought clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$. The well known k–**means clustering algorithm** [39] uses this criterion.

In **probabilistic clustering** the assignment of points to clusters is "soft", in the sense that the membership of a data point $\mathbf{x}$ in a cluster $\mathcal{C}_k$ is given as a **probability**, denoted by $p_k(\mathbf{x})$. These are subjective probabilities, indicating strength of belief in the event in question.

Let a **distance function**

$$d_k(\,\cdot\,,\,\cdot\,) \tag{3.5}$$

be defined for each cluster $\mathcal{C}_k$. These distance functions are, in general, different from one cluster to another. For each data point $\mathbf{x} \in \mathcal{D}$, we then compute:

• the **distance** $d_k(\mathbf{x}, \mathbf{c}_k)$, also denoted by $d_k(\mathbf{x})$ (since d_k is used only for distances from $\mathbf{c}_k$), or just d_k if $\mathbf{x}$ is understood, and

• a **probability** that $\mathbf{x}$ is a member of $\mathcal{C}_k$, denoted by $p_k(\mathbf{x})$, or just p_k.

Various relations between probabilities and distances can be assumed, resulting in different ways of clustering the data. In our experience, the following assumption has proved useful: For any point $\mathbf{x}$, and all $k = 1, \cdots, K$

$$p_k(\mathbf{x})\, d_k(\mathbf{x}) = \text{constant, depending on } \mathbf{x}\ .$$

This model is our working **principle** in what follows, and the basis of the **probabilistic d–clustering** approach of section 3.2.

The above principle owes its versatility to the different ways of choosing the distances $d_k(\cdot)$. It is also natural to consider increasing functions of such distances, and one useful choice is

$$p_k(\mathbf{x}) e^{d_k(\mathbf{x})} = \text{constant, depending on } \mathbf{x}\ ,$$

giving the **probabilistic exponential d–clustering** approach of section 3.3.

The probabilistic d–clustering algorithm is presented in section 3.4. It is a generalization, to several centers, of the Weizsfeld method for solving the Fermat–Weber location problem, see section 3.2.5, and convergence follows as in [59]. The updates of the centers use an extremal principle, described in section 3.2.3. The progress of the algorithm is monitored by the **joint distance function**, a distance function that captures the data in its low contours, see section 3.2.2. The centers updated by the algorithm are stationary points of the joint distance function.

For other approaches to probabilistic clustering see the surveys in [43], [78], and the seminal article [79] unifying clustering methods in the framework of modern optimization theory.

3.2 Probabilistic d–clustering

There are several ways to model the relationship between distances and probabilities. The simplest model, and our working **principle** (or axiom), is the following:

Principle 3.1. For each $\mathbf{x} \in \mathcal{D}$, and each cluster $\mathcal{C}_k$,

$$p_k(\mathbf{x})\, d_k(\mathbf{x}) = \text{constant, depending on } \mathbf{x}\ . \tag{3.6}$$

Cluster membership is thus more probable the closer the data point is to the cluster center. Note that the constant in (3.6) is independent of the cluster k.

3.2.1 Probabilities

From Principle 3.1, and the fact that probabilities add to one, we get

Theorem 3.1. Let the cluster centers $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$ be given, let $\mathbf{x}$ be a data point, and let $\{d_k(\mathbf{x}) : k = 1, \ldots, K\}$ be its distances from the given centers. Then the

membership probabilities of $\mathbf{x}$ are

$$p_k(\mathbf{x}) = \frac{\prod\limits_{j\neq k} d_j(\mathbf{x})}{\sum\limits_{t=1}^{K} \prod\limits_{j\neq t} d_j(\mathbf{x})} \ , \ k = 1, \ldots, K. \tag{3.7}$$

Proof. Using (3.6) we write for t, k

$$p_t(\mathbf{x}) = \left(\frac{p_k(\mathbf{x}) d_k(\mathbf{x})}{d_t(\mathbf{x})} \right) \ .$$

Since $\sum\limits_{t=1}^{K} p_t(\mathbf{x}) = 1$,

$$p_k(\mathbf{x}) \sum_{t=1}^{K} \left(\frac{d_k(\mathbf{x})}{d_t(\mathbf{x})} \right) = 1.$$

$$\therefore \ p_k(\mathbf{x}) = \frac{1}{\sum\limits_{t=1}^{K} \left(\frac{d_k(\mathbf{x})}{d_t(\mathbf{x})} \right)} = \frac{\prod\limits_{j\neq k} d_j(\mathbf{x})}{\sum\limits_{t=1}^{K} \prod\limits_{j\neq t} d_j(\mathbf{x})} \ .$$

□

In particular, for K=2,

$$p_1(\mathbf{x}) = \frac{d_2(\mathbf{x})}{d_1(\mathbf{x}) + d_2(\mathbf{x})} \ , \ p_2(\mathbf{x}) = \frac{d_1(\mathbf{x})}{d_1(\mathbf{x}) + d_2(\mathbf{x})} \ , \tag{3.8}$$

and for $K = 3$,

$$p_1(\mathbf{x}) = \frac{d_2(\mathbf{x}) d_3(\mathbf{x})}{d_1(\mathbf{x}) d_2(\mathbf{x}) + d_1(\mathbf{x}) d_3(\mathbf{x}) + d_2(\mathbf{x}) d_3(\mathbf{x})} \ , \text{ etc.} \tag{3.9}$$

Note: See [41] for related work in a different context. In particular, our equation (3.8) is closely related to [41, Eq. (5)].

3.2.2 The Joint Distance Function

We denote the constant in (3.6) by $D(\mathbf{x})$, a function of $\mathbf{x}$. Then

$$p_k(\mathbf{x}) = \frac{D(\mathbf{x})}{d_k(\mathbf{x})} \ , \ k = 1, \ldots, K.$$

Since the probabilities add to one we get,

$$D(\mathbf{x}) = \frac{\prod_{k=1}^{K} d_k(\mathbf{x}, \mathbf{c}_k)}{\sum_{t=1}^{K} \prod_{j \neq t} d_j(\mathbf{x}, \mathbf{c}_j)} \ . \tag{3.10}$$

The function $D(\mathbf{x})$, called the **joint distance function** (abbreviated **JDF**) of $\mathbf{x}$, has the dimension of distance, and measures the distance of $\mathbf{x}$ from all cluster centers. Here are special cases of (3.10), for $K = 2$,

$$D(\mathbf{x}) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x})}{d_1(\mathbf{x}) + d_2(\mathbf{x})} \ , \tag{3.11}$$

and for $K = 3$,

$$D(\mathbf{x}) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x})\, d_3(\mathbf{x})}{d_1(\mathbf{x})\, d_2(\mathbf{x}) + d_1(\mathbf{x})\, d_3(\mathbf{x}) + d_2(\mathbf{x})\, d_3(\mathbf{x})} \ . \tag{3.12}$$

The JDF of the whole data set $\mathcal{D}$ is the sum of (3.10) over all points, and is a function of the K cluster centers, say,

$$F(\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_K) = \sum_{i=1}^{N} \frac{\prod_{k=1}^{K} d_k(\mathbf{x}_i, \mathbf{c}_k)}{\sum_{t=1}^{K} \prod_{j \neq t} d_j(\mathbf{x}_i, \mathbf{c}_j)} \ . \tag{3.13}$$

Example 3.2. Figure 3.2 shows level sets of the JDF (3.11), with Mahalanobis distances

$$d_k(\mathbf{x}, \mathbf{c}_k) = \sqrt{(\mathbf{x} - \mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{x} - \mathbf{c}_k)} \ , \tag{3.14}$$

$\mathbf{c}_1 = \boldsymbol{\mu}_1, \ \mathbf{c}_2 = \boldsymbol{\mu}_2$, and $\Sigma_1, \ \Sigma_2$ as in Example 3.1.

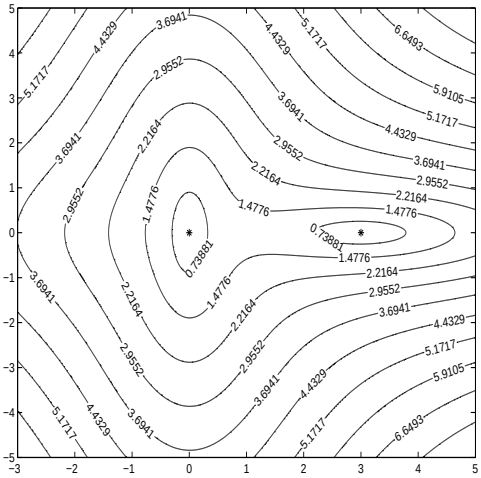


Figure 3.2: Level sets of a joint distance function

Notes:

(a) The JDF $D(\mathbf{x})$ of (3.10) is a measure of the classifiability of the point $\mathbf{x}$ in question. It is zero if and only if $\mathbf{x}$ coincides with one of the cluster centers, in which case $\mathbf{x}$ belongs to that cluster with probability 1. If all the distances $d_k(\mathbf{x}, \mathbf{c}_k)$ are equal, say equal to d, then $D(\mathbf{x}) = d/k$ and all $p_k(\mathbf{x}) = 1/K$, showing indifference between the clusters. As the distances $d_k(\mathbf{x})$ increase, so does $D(\mathbf{x})$, indicating greater uncertainty about the cluster where $\mathbf{x}$ belongs.

(b) The JDF (3.10) is, up to a constant, the **harmonic mean** of the distances involved, see [5] for an elucidation of the role of the harmonic mean in contour approximation of data. A related concept in ecology is the **home range**, shown in [25] to be the harmonic mean of the area moments in question.

3.2.3 An Extremal Principle

For simplicity consider the case of two clusters (the results are easily extended to the general case.)

Let $\mathbf{x}$ be a given data point with distances $d_1(\mathbf{x})$, $d_2(\mathbf{x})$ to the cluster centers. Then

the probabilities in (3.8) are the optimal solutions p_1, p_2 of the extremal problem

$$\begin{aligned} \text{Minimize} \quad & d_1(\mathbf{x})\, p_1^2 + d_2(\mathbf{x})\, p_2^2 \\ \text{subject to} \quad & p_1 + p_2 = 1 \\ & p_1,\, p_2 \geq 0 \end{aligned} \tag{3.15}$$

Indeed, the **Lagrangian** of this problem is

$$L(p_1, p_2, \lambda) = d_1(\mathbf{x})\, p_1^2 + d_2(\mathbf{x})\, p_2^2 - \lambda(p_1 + p_2 - 1) \tag{3.16}$$

and setting the partial derivatives (with respect to p_1, p_2) equal to zero gives the principle (3.6),

$$p_1\, d_1(\mathbf{x}) = p_2\, d_2(\mathbf{x})\ .$$

Substituting the probabilities (3.8) in the Lagrangian (3.16) we get the optimal value of (3.15),

$$L^*(p_1(\mathbf{x}), p_2(\mathbf{x}), \lambda) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x})}{d_1(\mathbf{x}) + d_2(\mathbf{x})}\ , \tag{3.17}$$

which is the JDF (3.11) again.

The extremal problem for a data set $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^p$ is, accordingly,

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{N} \left(d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2 + d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2 \right) \\ \text{subject to} \quad & p_1(\mathbf{x}_i) + p_2(\mathbf{x}_i) = 1, \\ & p_1(\mathbf{x}_i),\, p_2(\mathbf{x}_i) \geq 0,\ i = 1, \ldots, N. \end{aligned} \tag{3.18}$$

This problem separates into N problems like (3.15), and its optimal value is

$$\sum_{i=1}^{N} \frac{d_1(\mathbf{x}_i)\, d_2(\mathbf{x}_i)}{d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i)} \tag{3.19}$$

the JDF (3.13) of the data set, with $K = 2$.

Note: The explanation for the strange appearance of "probabilities squared" above, is that (3.15) is a smoothed version of the "real" clustering problem, namely,

$$\min\{d_1, d_2\},$$

which is nonsmooth, see [79] for a unified development of smoothed clustering methods.

3.2.4 Centers

We write (3.18) as a function of the cluster centers $\mathbf{c}_1, \mathbf{c}_2$,

$$f(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \left(d_1(\mathbf{x}_i, \mathbf{c}_1)\, p_1(\mathbf{x}_i)^2 + d_2(\mathbf{x}_i, \mathbf{c}_2)\, p_2(\mathbf{x}_i)^2 \right) . \tag{3.20}$$

If a point $\mathbf{x}_i$ coincides with a center, say $\mathbf{x}_i = \mathbf{c}_1$, then $d_1(\mathbf{x}_i) = 0$, $p_1(\mathbf{x}_i) = 1$ and $p_2(\mathbf{x}_i) = 0$. This point contributes zero to the summation.

For the special case of Euclidean distances, the minimizers of (3.20) assume a simple form as convex combinations of the data points.

Theorem 3.2. Let the distance functions d_1, d_2 in (3.20) be Euclidean,

$$d_k(\mathbf{x}, \mathbf{c}_k) = \|\mathbf{x} - \mathbf{c}_k\| \ , \ k = 1, 2 \ , \tag{3.21}$$

so that

$$f(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1,\ldots,N} \left(\|\mathbf{x}_i - \mathbf{c}_1\|\, p_1(\mathbf{x}_i)^2 + \|\mathbf{x}_i - \mathbf{c}_2\|\, p_2(\mathbf{x}_i)^2 \right) , \tag{3.22}$$

and let the probabilities $p_1(\mathbf{x}_i), p_2(\mathbf{x}_i)$ be given for $i = 1, \ldots, N$. We make the following assumption about the minimizers $\mathbf{c}_1, \mathbf{c}_2$ of (3.22):

$$\mathbf{c}_1, \mathbf{c}_2 \text{ do not coincide with any of the points } \mathbf{x}_i, \ i = 1, \ldots, N. \tag{3.23}$$

Then the minimizers $\mathbf{c}_1, \mathbf{c}_2$ are given by

$$\mathbf{c}_k = \sum_{i=1,\ldots,N} \left(\frac{u_k(\mathbf{x}_i)}{\sum\limits_{j=1,\ldots,N} u_k(\mathbf{x}_j)} \right) \mathbf{x}_i \ , \tag{3.24}$$

where

$$u_k(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \ , \tag{3.25}$$

for $k = 1, 2$, or equivalently, using (3.8),

$$\begin{aligned} u_1(\mathbf{x}_i) &= \frac{d_2(\mathbf{x}_i, \mathbf{c}_2)^2}{d_1(\mathbf{x}_i, \mathbf{c}_1)\left(d_1(\mathbf{x}_i, \mathbf{c}_1) + d_2(\mathbf{x}_i, \mathbf{c}_2)\right)^2} \ , \\ u_2(\mathbf{x}_i) &= \frac{d_1(\mathbf{x}_i, \mathbf{c}_1)^2}{d_2(\mathbf{x}_i, \mathbf{c}_2)\left(d_1(\mathbf{x}_i, \mathbf{c}_1) + d_2(\mathbf{x}_i, \mathbf{c}_2)\right)^2} \ . \end{aligned} \tag{3.26}$$

Proof. The gradient of $d(\mathbf{x}, \mathbf{c}) = \|\mathbf{x} - \mathbf{c}\|$ with respect to $\mathbf{c}$ is, for $\mathbf{x} \neq \mathbf{c}$,

$$\nabla_{\mathbf{c}} \|\mathbf{x} - \mathbf{c}\| = -\frac{\mathbf{x} - \mathbf{c}}{\|\mathbf{x} - \mathbf{c}\|} = -\frac{\mathbf{x} - \mathbf{c}}{d(\mathbf{x}, \mathbf{c})} \ . \tag{3.27}$$

By Assumption (3.23), the gradient of (3.22) with respect to $\mathbf{c}_k$ is

$$\begin{aligned} \nabla_{\mathbf{c}_k} f(\mathbf{c}_1, \mathbf{c}_2) &= -\sum_{i=1,\ldots,N} \frac{\mathbf{x}_i - \mathbf{c}_k}{\|\mathbf{x}_i - \mathbf{c}_k\|} \, p_k(\mathbf{x}_i)^2 \\ &= -\sum_{i=1,\ldots,N} \frac{\mathbf{x}_i - \mathbf{c}_k}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \, p_k(\mathbf{x}_i)^2 \ , k = 1, 2 \ . \end{aligned} \tag{3.28}$$

Setting the gradient equal to zero, and summing like terms, we get

$$\sum_{i=1,\ldots,N} \left(\frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \right) \mathbf{x}_i = \left(\sum_{i=1,\ldots,N} \frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \right) \mathbf{c}_k \ , \tag{3.29}$$

proving (3.24)–(3.26). □

The same formulas for the centers $\mathbf{c}_1, \mathbf{c}_2$ hold if the norm used in (3.22) is elliptic.

Corollary 3.1. Let the distance functions d_1, d_2 in (3.20) be elliptic,

$$d_k(\mathbf{x}, \mathbf{c}_k) = \langle (\mathbf{x} - \mathbf{c}_k), Q_k(\mathbf{x} - \mathbf{c}_k) \rangle^{1/2} , \tag{3.30}$$

with positive–definite matrices Q_k. Then the minimizers $\mathbf{c}_1, \mathbf{c}_2$ of (3.20) are given by (3.24)–(3.26).

Proof. The gradient of $d(\mathbf{x}, \mathbf{c}) = \langle (\mathbf{x} - \mathbf{c}), Q(\mathbf{x} - \mathbf{c}) \rangle^{1/2}$ with respect to $\mathbf{c}$ is, for $\mathbf{x} \neq \mathbf{c}$,

$$\nabla_{\mathbf{c}}\, d(\mathbf{x}, \mathbf{c}) = -\frac{Q(\mathbf{x} - \mathbf{c})}{d(\mathbf{x}, \mathbf{c})} .$$

Therefore the analog of (3.28) is

$$\nabla_{\mathbf{c}_k} f(\mathbf{c}_1, \mathbf{c}_2) = -Q_k \sum_{i=1,\ldots,N} \frac{\mathbf{x}_i - \mathbf{c}_k}{d_k(\mathbf{x}_i, \mathbf{c}_k)}\, p_k(\mathbf{x}_i)^2 , \tag{3.31}$$

and since Q_k is nonsingular, it can be "cancelled" when we set the gradient equal to zero. The rest of the proof is as in Theorem 3.2. □

Corollary 3.1 applies, in particular, to the Mahalanobis distance (3.14)

$$d_k(\mathbf{x}, \mathbf{c}_k) = \sqrt{(\mathbf{x} - \mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{x} - \mathbf{c}_k)} ,$$

where Σ_k is the covariance matrix of the cluster $\mathcal{C}_k$.

The formulas (3.24)–(3.25) are also valid in the general case of K clusters, where the analog of (3.20) is

$$f(\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_K) = \sum_{i=1,\ldots,N} \sum_{k=1}^{K} d_k(\mathbf{x}_i, \mathbf{c}_k)\, p_k(\mathbf{x}_i)^2 . \tag{3.32}$$

Corollary 3.2. Let the distance functions d_k in (3.32) be elliptic, as in (3.30), and let the probabilities $p_k(\mathbf{x}_i)$ be given. Then the minimizers $\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_K$ of (3.32) are given by (3.24)–(3.25) for $k = 1, 2, \cdots, K$.

Proof. The proof of Corollary 3.1 holds in the general case, since the minimizers are

calculated separately. □

3.2.5 The Weiszfeld Method

In the case of one cluster (where the probabilities are all 1 and therefore of no interest) the center formulas (3.24)–(3.25) reduce to

$$\mathbf{c} = \sum_{i=1,\dots,N} \left(\frac{1/d(\mathbf{x}_i, \mathbf{c})}{\sum\limits_{j=1,\dots,N} 1/d(\mathbf{x}_j, \mathbf{c})} \right) \mathbf{x}_i \ , \tag{3.33}$$

giving the minimizer of $f(\mathbf{c}) = \sum\limits_{i=1}^{N} d(\mathbf{x}_i, \mathbf{c})$. Formula (3.33) can be used iteratively to update the center $\mathbf{c}$ (on the left) as a convex combination of the points $\mathbf{x}_i$ with weights depending on the current center. This iteration is the **Weiszfeld method** [82] for solving the Fermat–Weber location problem, see [82], [60]. Convergence of Weiszfeld's method was established in Kuhn [59] by modifying the gradient $\nabla f(\mathbf{c})$ so that it is always defined, see [54] for further details. However, the modification is not carried out in practice since, as shown by Kuhn, the set of initial points $\mathbf{c}$ for which it ever becomes necessary is denumerable.

In what follows we use the formulas (3.24)–(3.25) iteratively to update the centers. Convergence can be proved by adapting the arguments of Kuhn [59], but as there it requires no special steps in practice.

3.2.6 The Centers and the Joint Distance Function

The centers given by (3.24)–(3.25) are related to the JDF (3.13) of the data set. Consider first the case of $K = 2$ clusters, where (3.13) reduces to

$$F(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \frac{d_1(\mathbf{x}_i, \mathbf{c}_1)\, d_2(\mathbf{x}_i, \mathbf{c}_2)}{d_1(\mathbf{x}_i, \mathbf{c}_1) + d_2(\mathbf{x}_i, \mathbf{c}_2)} \ . \tag{3.34}$$

The points $\mathbf{c}_k$ where $\nabla_{\mathbf{c}_k} F(\mathbf{c}_1, \mathbf{c}_2) = \mathbf{0}, k = 1, 2$, are called **stationary points** of (3.34).

Theorem 3.3. Let the distances d_1, d_2 in (3.34) be elliptic, as in (3.30). Then the stationary points of $F(\mathbf{c}_1, \mathbf{c}_2)$ are given by (3.24)–(3.26).

Proof. Let the distances d_k be Euclidean, $d_k(\mathbf{x}) = \|\mathbf{x} - \mathbf{c}_k\|$. It is enough to prove the theorem for one center, say $\mathbf{c}_1$. Using (3.27) we derive

$$\nabla_{\mathbf{c}_1} F(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \frac{(d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))\, d_2(\mathbf{x}_i) \left(-\dfrac{\mathbf{x}_i - \mathbf{c}_1}{d_1(\mathbf{x}_i)} \right) + d_1(\mathbf{x}_i)\, d_2(\mathbf{x}_i) \left(\dfrac{\mathbf{x}_i - \mathbf{c}_1}{d_1(\mathbf{x}_i)} \right)}{(d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2}$$

$$= \sum_{i=1}^{N} \frac{-d_2(\mathbf{x}_i)^2\, (\mathbf{x}_i - \mathbf{c}_1)}{d_1(\mathbf{x}_i)\, (d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2} \,. \quad (3.35)$$

Setting (3.35) equal to zero, and summing like terms, we get

$$\left(\sum_{j=1}^{N} \frac{d_2(\mathbf{x}_j)^2}{d_1(\mathbf{x}_j)\, (d_1(\mathbf{x}_j) + d_2(\mathbf{x}_j))^2} \right) \mathbf{c}_1 = \sum_{i=1}^{N} \left(\frac{d_2(\mathbf{x}_i)^2}{d_1(\mathbf{x}_i)\, (d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2} \right) \mathbf{x}_i \,,$$

duplicating (3.24)–(3.26). If the distances are elliptic, as in (3.30), then the analog of (3.35) is,

$$\nabla_{\mathbf{c}_1} F(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \frac{-d_2(\mathbf{x}_i)^2\, Q_1\, (\mathbf{x}_i - \mathbf{c}_1)}{d_1(\mathbf{x}_i)\, (d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2}$$

and since Q_1 is nonsingular, it can be "cancelled" when the gradient is set equal to zero. □

In the above proof the stationary points $\mathbf{c}_1, \mathbf{c}_2$ are calculated separately, and the calculation does not depend on there being 2 clusters. We thus have:

Corollary 3.3. Consider a data set with K clusters, and elliptic distances d_k. Then the stationary points of the JDF (3.13) are the centers $\mathbf{c}_k$ given by (3.24)–(3.25). □

Note: The JDF (3.10) is zero exactly at the K centers $\{\mathbf{c}_k\}$, and is positive elsewhere. These centers are therefore the global minimizers of (3.10). However, the function (3.10) is not convex, not even quasi–convex, and may have other stationary points, that are necessarily saddle points.

3.2.7 Why d and not d^2?

The extremal principle (3.18), which is the basis of our work, is linear in the distances d_k,

$$\text{Minimize} \quad \sum_k d_k \, p_k^2.$$

We refer to this as the d–**model**.

In clustering, and statistics in general, it is customary to use the distances squared in the objective function,

$$\text{Minimize} \quad \sum_k d_k^2.$$

We call this the d^2–**model**.

The d^2–model has a long tradition, dating back to Gauss, and is endowed with a rich statistical theory. There are geometrical advantages (Pythagoras Theorem), as well as analytical (linear derivatives).

The d–model is suggested by the analogy between clustering and location problems, where sums of distances (not distances squared) are minimized. Our center formulas (3.24)–(3.25) are thus generalizations of the Weiszfeld Method to several facilities, see section 3.2.5.

An advantage of the d–model is its robustness. Indeed the formula (3.25), which does not follow from the d^2–model, guarantees that outliers will not affect the center locations.

3.2.8 Other Principles

There are alternative ways of modelling the relations between distances and probabilities. For example:

Principle 3.2. For each $\mathbf{x} \in \mathcal{D}$, and each cluster $\mathcal{C}_k$, the probability $p_k = p_k(\mathbf{x})$ and distance $d_k = d_k(\mathbf{x})$ are related by

$$p_k^\alpha \, d_k^\beta = \text{constant, depending on } \mathbf{x} \, . \tag{3.36}$$

where the exponents α, β are positive.

For the case of 2 clusters we get, by analogy with (3.8) and (3.18) respectively, the probabilities

$$p_1(\mathbf{x}) = \frac{d_2(\mathbf{x})^{\beta/\alpha}}{d_1(\mathbf{x})^{\beta/\alpha} + d_2(\mathbf{x})^{\beta/\alpha}} \ , \ p_2(\mathbf{x}) = \frac{d_1(\mathbf{x})^{\beta/\alpha}}{d_1(\mathbf{x})^{\beta/\alpha} + d_2(\mathbf{x})^{\beta/\alpha}} \ , \tag{3.37}$$

and an extremal principle,

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{N} \left(d_1(\mathbf{x}_i)^{\beta} \, p_1(i)^{\alpha+1} + d_2(\mathbf{x}_i)^{\beta} \, p_2(i)^{\alpha+1} \right) \\ \text{subject to} \quad & p_1(i) + p_2(i) = 1 \\ & p_1(i), \, p_2(i) \geq 0 \end{aligned} \tag{3.38}$$

where $p_1(i), p_2(i)$ are the cluster probabilities at $\mathbf{x}_i$.

The **Fuzzy Clustering Method** [14], [15], which is an extension of k-means method, uses $\beta = 2$ and allows different choices of α. For $\alpha = 2$, it gives the same probabilities as (3.7), however the center updates are different than (3.24)–(3.25).

3.3 Probabilistic Exponential d–clustering

Any increasing function of the distance can be used in Principle 3.1. The following model, with probabilities decaying exponentially as distances increase, has proved useful in our experience.

Principle 3.3. For each $\mathbf{x} \in \mathcal{D}$, and each cluster $\mathcal{C}_k$, the probability $p_k(\mathbf{x})$ and distance $d_k(\mathbf{x})$ are related by

$$p_k(\mathbf{x}) \, e^{d_k(\mathbf{x})} = E(\mathbf{x}), \text{ a constant depending on } \mathbf{x} \ . \tag{3.39}$$

Most results of section 3.2 hold also for Principle 3.3, with the distance $d_k(\mathbf{x})$ replaced

by $e^{d_k(\mathbf{x})}$. Thus the analog of the probabilities (3.8) is

$$p_1(\mathbf{x}) = \frac{e^{d_2(\mathbf{x})}}{e^{d_1(\mathbf{x})} + e^{d_2(\mathbf{x})}} \ , \ p_2(\mathbf{x}) = \frac{e^{d_1(\mathbf{x})}}{e^{d_1(\mathbf{x})} + e^{d_2(\mathbf{x})}} \ , \tag{3.40}$$

or equivalently

$$p_1(\mathbf{x}) = \frac{e^{-d_1(\mathbf{x})}}{e^{-d_1(\mathbf{x})} + e^{-d_2(\mathbf{x})}} \ , \ p_2(\mathbf{x}) = \frac{e^{-d_2(\mathbf{x})}}{e^{-d_1(\mathbf{x})} + e^{-d_2(\mathbf{x})}} \ . \tag{3.41}$$

Similarly, since the probabilities add to 1, the constant in (3.39) is

$$E(\mathbf{x}) = \frac{e^{d_1(\mathbf{x})+d_2(\mathbf{x})}}{e^{d_1(\mathbf{x})} + e^{d_2(\mathbf{x})}} \ , \tag{3.42}$$

called the **exponential JDF**.

3.3.1 An Extremal Principle

The probabilities (3.40) are the optimal solutions of the problem

$$\min_{p_1,p_2} \quad \left\{ e^{d_1} p_1^2 + e^{d_2} p_2^2 \ : \ p_1 + p_2 = 1 \ , \ p_1, \, p_2 \geq 0 \right\} \ , \tag{3.43}$$

whose optimal value, obtained by substituting the probabilities (3.40), is again the exponential JDF (3.42).

The extremal problem for a data set $\mathcal{D} = \{\mathbf{x}_1, \, \mathbf{x}_2, \, \ldots, \, \mathbf{x}_N\} \subset \mathbb{R}^p$, partitioned into 2 clusters, is the following analog of (3.18)

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{N} \left(e^{d_1(\mathbf{x}_i)} \, p_1(i)^2 + e^{d_2(\mathbf{x}_i)} \, p_2(i)^2 \right) \\ \text{subject to} \quad & p_1(i) + p_2(i) = 1 \\ & p_1(i), \, p_2(i) \geq 0 \end{aligned} \tag{3.44}$$

where $p_1(i), p_2(i)$ are the cluster probabilities at $\mathbf{x}_i$. The problem separates into N problems like (3.43), and its optimal value is

$$\sum_{i=1}^{N} \frac{e^{d_1(\mathbf{x}_i)+d_2(\mathbf{x}_i)}}{e^{d_1(\mathbf{x}_i)} + e^{d_2(\mathbf{x}_i)}} , \tag{3.45}$$

the exponential JDF of the whole data set.

Alternatively, (3.39) follows from the "smoothed" extremal principle

$$\min_{p_1,p_2} \quad \left\{ \sum_{k=1}^{2} p_k \, d_k + \sum_{k=1}^{2} p_k \, \log p_k \, : \, p_1 + p_2 = 1 \, , \; p_1, \, p_2 \geq 0 \right\} , \tag{3.46}$$

obtained by adding an entropy term to $\sum p_k \, d_k$. Indeed the Lagrangian of (3.46) is

$$L(p_1, p_2, \lambda) = \sum_{k=1}^{2} p_k \, d_k + \sum_{k=1}^{2} p_k \, \log p_k - \lambda \, (p_1 + p_2 - 1) \, .$$

Differentiation with respect to p_k, and equating to 0, gives

$$d_k + 1 + \log p_k - \lambda = 0$$

which is (3.39).

3.3.2 Centers

We write (3.44) as a function of the cluster centers $\mathbf{c}_1, \mathbf{c}_2$,

$$f(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \left(e^{d_1(\mathbf{x}_i, \mathbf{c}_1)} \, p_1(\mathbf{x}_i)^2 + e^{d_2(\mathbf{x}_i, \mathbf{c}_2)} \, p_2(\mathbf{x}_i)^2 \right) \tag{3.47}$$

and for elliptic distances we can verify, as in Theorem 3.2, that the minimizers of (3.47) are given by,

$$\mathbf{c}_k = \sum_{i=1}^{N} \left(\frac{u_k(\mathbf{x}_i)}{\sum\limits_{j=1}^{N} u_k(\mathbf{x}_j)} \right) \mathbf{x}_i \, , \tag{3.48}$$

where (compare with (3.25)),

$$u_k(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)^2\, e^{d_k(\mathbf{x}_i)}}{d_k(\mathbf{x}_i)} \ , \tag{3.49}$$

or equivalently,

$$u_1(\mathbf{x}_i) = \frac{e^{-d_1(\mathbf{x}_i)}/d_1(\mathbf{x}_i)}{(e^{-d_1(\mathbf{x}_i)} + e^{-d_2(\mathbf{x}_i)})^2} \ , \quad u_2(\mathbf{x}_i) = \frac{e^{-d_2(\mathbf{x}_i)}/d_2(\mathbf{x}_i)}{(e^{-d_1(\mathbf{x}_i)} + e^{-d_2(\mathbf{x}_i)})^2} \ . \tag{3.50}$$

As in Theorem 3.3, these minimizers are the stationary points of the JDF, given here as

$$F(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \frac{e^{d_1(\mathbf{x}_i,\mathbf{c}_1)+d_2(\mathbf{x}_i,\mathbf{c}_2)}}{e^{d_1(\mathbf{x}_i,\mathbf{c}_1)} + e^{d_2(\mathbf{x}_i,\mathbf{c}_2)}} \ . \tag{3.51}$$

Finally we can verify, as in Corollary 3.2, that the results hold in the general case of K clusters.

3.4 A Probabilistic d–clustering Algorithm

The ideas of section 3.2–3.3 are implemented in the following algorithm for unsupervised clustering of data. A schematic description, presented – for simplicity – for the case of 2 clusters, follows.

Algorithm 3.1. Probabilistic D–clustering

Initialization:	given data $\mathcal{D}$, any two points $\mathbf{c}_1, \mathbf{c}_2$, and $\epsilon > 0$
Iteration:	
Step 1	**compute** distances $d_1(\mathbf{x})$, $d_2(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{D}$
Step 2	**update** the centers $\mathbf{c}_1^+, \mathbf{c}_2^+$
Step 3	**if** $\|\mathbf{c}_1^+ - \mathbf{c}_1\| + \|\mathbf{c}_2^+ - \mathbf{c}_2\| < \epsilon$ **stop**
	return to step 1

The algorithm iterates between the cluster **centers**, (3.24) or (3.48), and the **distances** of the data points to these centers. The cluster **probabilities**, (3.8) or (3.40), are not used explicitly.

Notes:

(a) The distance used in Step 1 can be Euclidean or elliptic (the formulas (3.24)–(3.26), and (3.48)–(3.50), are valid in both cases.)

(b) In Step 2, the centers are updated by (3.24)–(3.26) if Principle 3.1 is used, and by (3.48)–(3.50) for Principle 3.3.

(c) In particular, if the Mahalanobis distance (3.14)

$$d(\mathbf{x}, \mathbf{c}_k) = \sqrt{(\mathbf{x} - \mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{x} - \mathbf{c}_k)}$$

is used, the covariance matrix Σ_k of the k^{th}–cluster, can be estimated at each iteration by

$$\Sigma_k = \frac{\sum_{i=1}^{N} u_k(\mathbf{x}_i)(\mathbf{x}_i - \mathbf{c}_k)(\mathbf{x}_i - \mathbf{c}_k)^T}{\sum_{i=1}^{N} u_k(\mathbf{x}_i)} \tag{3.52}$$

with $u_k(\mathbf{x}_i)$ given by (3.26) or (3.50)

(d) The computations stop (in Step 3) when the centers stop moving, at which point the cluster membership probabilities may be computed by (3.8) or (3.40). These probabilities are not needed in the algorithm, but may be used for classifying the data points, after the cluster centers have been computed.

(e) Using the arguments of [59] it can be shown that the objective function (3.32) decreases at each iteration, and the Algorithm converges.

(f) The cluster centers and distance functions change at each iteration, and so does the function (3.13) itself, which decreases at each iteration. The JDF may have stationary points that are not minimizers, however such points are necessarily saddle points, and will be missed by the Algorithm with probability 1.

Example 3.3. We apply the algorithm, using d–clustering as in section 3.2 and Mahalanobis distance, to the data of Example 3.1. Figure 3.3 shows the evolution of the joint distance function, represented by its level sets. The initial function, shown in the top-left pane, corresponds to the (arbitrarily chosen) initial centers and initial covariances $\Sigma_1 = \Sigma_2 = I$. The covariances are updated at each iteration using (3.52), and by

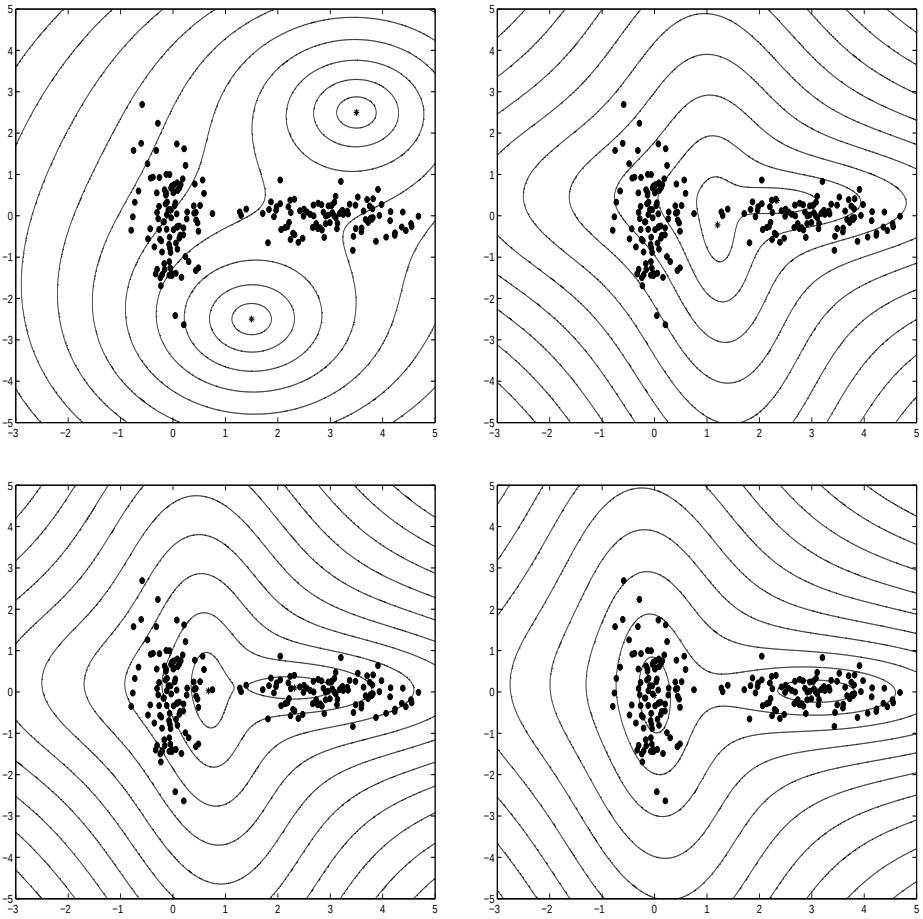


Figure 3.3: The level sets of the evolving **joint distance function** at **iteration 0** (top left), **iteration 1** (top right), **iteration 2** (bottom left) and **iteration 12** (bottom right)

iteration 8 the function is already very close to its final form, shown in the bottom-right pane. For a tolerance of $\epsilon = 0.01$ the algorithm terminated in 12 iterations.

Example 3.4. In Figure 3.4 we illustrate the movement of the cluster centers for different initial centers. The centers at each run are shown with the final level sets of the joint distance function found in Example 3.3.

The algorithm gives the correct cluster centers, for all initial starts. In particular, the two initial centers may be arbitrarily close, as shown in the top–left pane of Fig. 3.4.

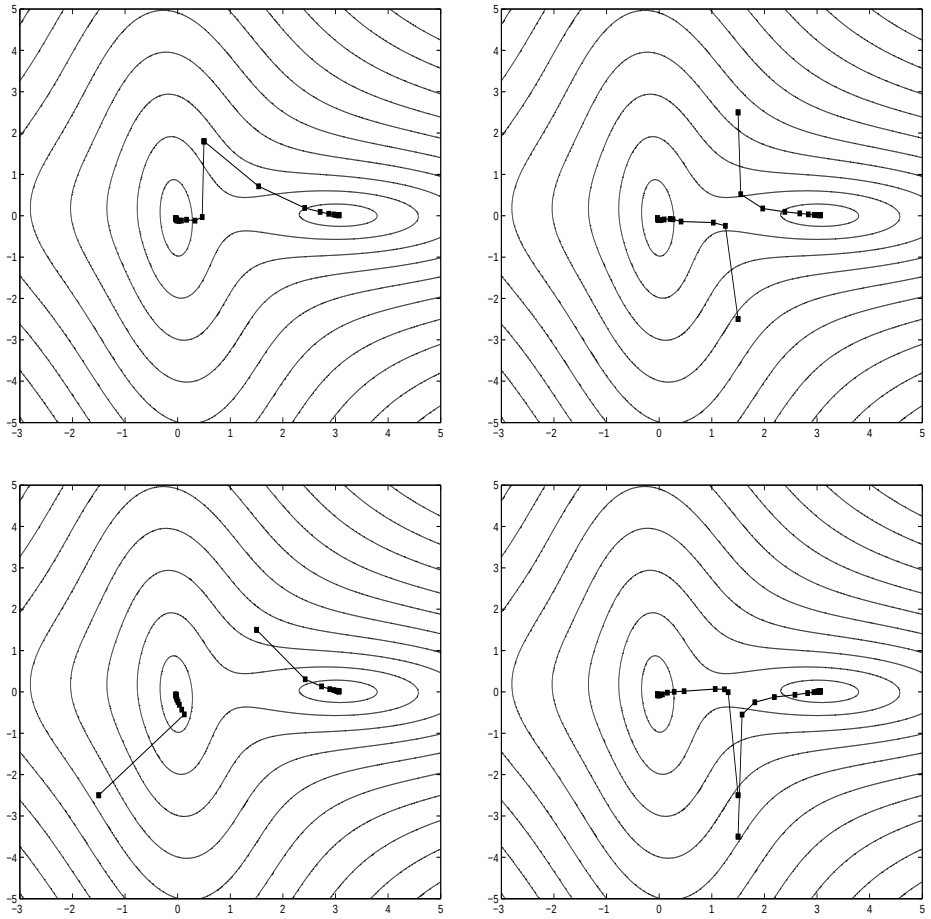


Figure 3.4: Movements of the cluster centers for different starts. The top–right pane shows the centers corresponding to Fig. 3.3. The top–left pane shows very close initial centers.

Example 3.5. The class membership probabilities (3.8) were then computed using the centers determined by the algorithm. The level sets of the probability $p_1(\mathbf{x})$ are shown in Figure 3.5. The curve $p_1(\mathbf{x}) = 0.5$, the thick curve shown in the left pane of Fig. 3.5,

may serve as the clustering rule. Alternatively, the 2 clusters can be defined as

$$\mathcal{C}_1 = \{\mathbf{x} : p_1(\mathbf{x}) \geq 0.6\},\ \mathcal{C}_2 = \{\mathbf{x} : p_1(\mathbf{x}) \leq 0.4\}\ ,$$

with points $\{\mathbf{x} : 0.4 < p_1(\mathbf{x}) < 0.6\}$ left unclassified, see the right pane of Fig. 3.5.

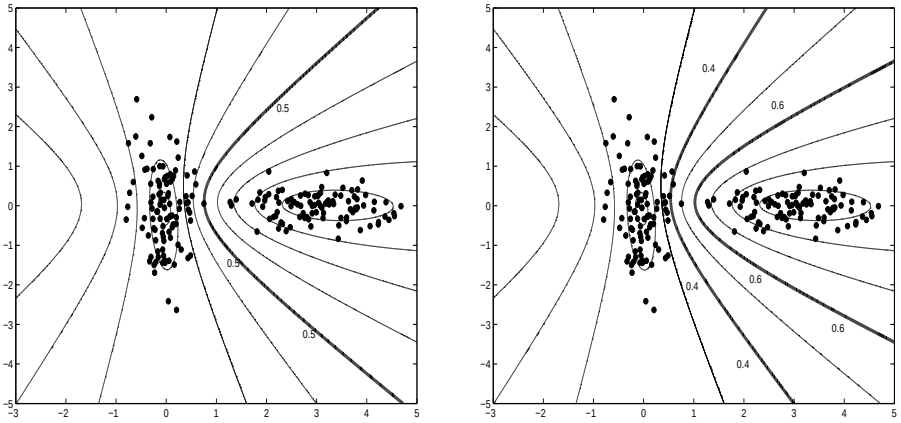


Figure 3.5: The level sets of the probabilities $p_1(\mathbf{x})$ and two clustering rules.

3.5 Related Work

There are applications where the **cluster sizes** (ignored here) need to be estimated. An important example is parameter estimation in mixtures of distributions. The above method, adjusted for cluster sizes, is applicable, and in particular presents a viable alternative to the EM method, see [49] and [50].

As noted at the end of section 3.2.4, our method allows an extension of the classical Weiszfeld method to several facilities. This is the subject of [46], giving the solution of **multi-facility location problems**, including the capacitated case (which corresponds to given cluster sizes.)

A simple and practical criterion for **clustering validity**, determining the "right" number of clusters that fit a given data, is given in [47]. This criterion is based on the monotonicity of the JDF (3.13) as a function of the number of clusters.

Semi–supervised clustering is a framework for reconciling **supervised learning**, using any prior information ("labels") on the data, with **unsupervised clustering**, based on the intrinsic properties and geometry of the data set. A new method for semi-supervised clustering, combining probabilistic distance clustering for the unlabelled data points and a least squares criterion for the labelled ones, is given in [48].

3.6 Conclusions

The **probabilistic distance clustering algorithm** presented here is simple, fast (requiring a small number of cheap iterations), robust (insensitive to outliers), and gives a high percentage of correct classifications.

It was tried on hundreds of problems with both simulated and real data sets. In simulated examples, where the answers are known, the algorithm, starting at random initial centers, always converged – in our experience – to the true cluster centers.

Results of our numerical experiments, and comparisons with other distance–based clustering algorithms, will be reported elsewhere.

Chapter 4

Probabilistic Clustering Adjusted for Cluster Size

4.1 Introduction

A method for probabilistic clustering of data, proposed in [10], is based on the assumption that the probability of a point belonging to a cluster is inversely proportional to its distance from the cluster center. The resulting clustering algorithm is fast and efficient, and works best if the cluster sizes are about equal.

In cases where the cluster sizes differ greatly, or the cluster sizes themselves are unknowns that need to be estimated (as in de–mixing problems), the above assumption can be modified to take into account the cluster sizes. This modification is the objective of this chapter.

We take data points to be vectors $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^p$, and consider a **dataset** $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$. A **cluster** is a set of data points that are similar, in some sense, and **clustering** is a process of partitioning a data set into disjoint clusters.

In **distance clustering** (or d–clustering), "similarity" is interpreted in terms of a **distance** function $d(\mathbf{x}, \mathbf{y})$ in $\mathbb{R}^p$, for example,

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|, \ \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^p,$$

where $\|\cdot\|$ is a norm. A common choice is the **Mahalanobis distance** with the norm

$$\|\mathbf{u}\| = \langle \mathbf{u}, \Sigma^{-1}\mathbf{u} \rangle^{1/2},$$

where Σ is the covariance matrix of the data in question.

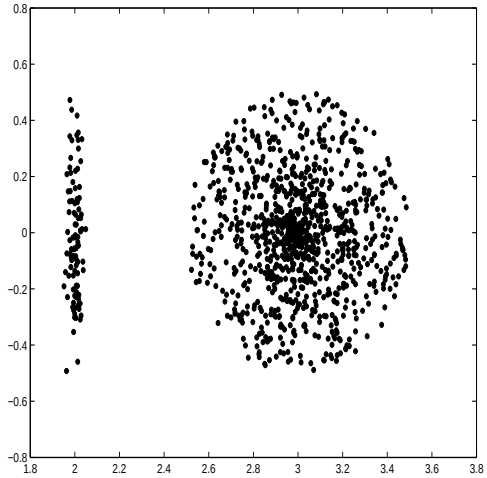


Figure 4.1: A data set in $\mathbb{R}^2$

Example 4.1. A data set in $\mathbb{R}^2$ with $N = 1100$ data points is shown in Figure 4.1. The data on the left was simulated from a normal distribution $N(\boldsymbol{\mu}, \Sigma)$, with

$$\boldsymbol{\mu}_1 = (2, 0),\ \Sigma_1 = \begin{pmatrix} 0.0005 & 0 \\ 0 & 0.05 \end{pmatrix},\ (100 \text{ points}) ,$$

and the data on the right consist of 1000 points simulated in a circle of diameter 1 centered at $\boldsymbol{\mu}_2 = (3, 0)$, from a radially symmetric distribution with $\text{Prob}\{\|\mathbf{x} - \boldsymbol{\mu}_2\| \leq r\} = 2\,r$. This data will serve as illustration in Examples 4.2–4.3 below.

Points are assigned to clusters using a **clustering criterion**. In d–clustering each point is assigned to the cluster with the nearest center. After each assignment, the cluster centers may change, resulting in further re–classifications. A d–clustering algorithm will therefore iterate between centers and re–assignments. The best known such method is the k–**means clustering algorithm** (see section 2.4 and also [39]).

In **probabilistic clustering** the assignment of points to clusters is "soft", and cluster membership is replaced by probabilities $p_k(\mathbf{x}) = \text{Prob}\{\mathbf{x} \in \mathcal{C}_k\}$, that a data point $\mathbf{x}$ belongs to the cluster $\mathcal{C}_k$. Probabilistic d–clustering is when the probabilities depend on the relevant distances.

Probabilistic d–clustering adjusted for the cluster size is called **probabilistic dq–clustering**, or **PDQ clustering** for short.

An algorithm for probabilistic dq–clustering is presented in section 4.3. The centers are updated as optimal solutions of the extremal problem in section 4.2.3. These centers are also stationary points of the **joint distance function**, a function that approximates the data in its lowest level sets, see section 4.2.2. The cluster sizes (if not given) are updated using the extremal problem of section 4.2.4

For other approaches to probabilistic clustering see sections 2.4.2, 2.4.3 and the surveys in Höppner et al. [43], Tan et al. [78].

4.2 Probabilistic dq–clustering

Let a data set $\mathcal{D} \subset \mathbb{R}^p$ be partitioned into K clusters $\{\mathcal{C}_k : k = 1, \cdots, K\}$,

$$\mathcal{D} = \bigcup_{k=1}^{K} \mathcal{C}_k \ ,$$

and let $\mathbf{c}_k$ be the **center** (in some sense) of the cluster $\mathcal{C}_k$. The **size** q_k of $\mathcal{C}_k$ is known in some applications, and is an unknown to be estimated in others. Here the cluster size, or its estimate, is assumed given wherever it appears in the right hand side of a formula.

With each data point $\mathbf{x} \in \mathcal{D}$ and a cluster $\mathcal{C}_k$, we associate:

- a **distance** $d_k(\mathbf{x}, \mathbf{c}_k)$, also denoted $d_k(\mathbf{x})$, and
- a **probability** of membership in $\mathcal{C}_k$, denoted $p_k(\mathbf{x})$.

The distance functions $d_k(\cdot)$, associated with different clusters, are different in general. In particular, we may use a different Mahalanobis distance for each cluster

$$d_k(\mathbf{x}) = \langle \mathbf{x} - \mathbf{c}_k, \Sigma_k^{-1}(\mathbf{x} - \mathbf{c}_k) \rangle^{1/2}, \tag{4.1}$$

where Σ_k is an estimate of the cluster covariance.

There are several ways to model the relationship between distances and probabilities [10], see chapter 3. The following assumption is our basic principle.

Principle 4.1. For each $\mathbf{x} \in \mathcal{D}$ and cluster $\mathcal{C}_k$, the probability $p_k(\mathbf{x})$ satisfies

$$\frac{p_k(\mathbf{x})\, d_k(\mathbf{x})}{q_k} = \text{constant, say } D(\mathbf{x}), \text{ depending on } \mathbf{x}\ . \tag{4.2}$$

Cluster membership is thus more probable the closer the data point is to the cluster center and the bigger is the cluster.

4.2.1 Probabilities

From Principle 4.1 and the fact that probabilities add to one we get:

Theorem 4.1. Let the cluster centers $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$ be given, let $\mathbf{x}$ be a data point, and let $\{d_k(\mathbf{x}) : k = 1, \ldots, K\}$ be its distances from the given centers. Then the membership probabilities of $\mathbf{x}$ are

$$p_k(\mathbf{x}) = \frac{\prod\limits_{j \neq k} \dfrac{d_j(\mathbf{x})}{q_j}}{\sum\limits_{i=1}^{K} \prod\limits_{j \neq i} \dfrac{d_j(\mathbf{x})}{q_j}}\ ,\ k = 1, \ldots, K. \tag{4.3}$$

Proof. Using (4.2) we write for i, k,

$$p_i(\mathbf{x}) = \frac{p_k(\mathbf{x}) d_k(\mathbf{x})/q_k}{d_i(\mathbf{x})/q_i}\ .$$

Since $\sum\limits_{i=1}^{K} p_i(\mathbf{x}) = 1$,

$$p_k(\mathbf{x}) \sum_{i=1}^{K} \left(\frac{d_k(\mathbf{x})/q_k}{d_i(\mathbf{x})/q_i} \right) = 1.$$

$$\therefore\ p_k(\mathbf{x}) = \frac{1}{\sum\limits_{i=1}^{K} \left(\dfrac{d_k(\mathbf{x})/q_k}{d_i(\mathbf{x})/q_i} \right)} = \frac{\prod\limits_{j \neq k} d_j(\mathbf{x})/q_j}{\sum\limits_{i=1}^{K} \prod\limits_{j \neq i} d_j(\mathbf{x})/q_j}\ .$$

□

In particular, for $K = 2$,

$$p_1(\mathbf{x}) = \frac{d_2(\mathbf{x})/q_2}{d_1(\mathbf{x})/q_1 + d_2(\mathbf{x})/q_2} \ , \ p_2(\mathbf{x}) = \frac{d_1(\mathbf{x})/q_1}{d_1(\mathbf{x})/q_1 + d_2(\mathbf{x})/q_2} \ , \tag{4.4}$$

and for $K = 3$,

$$p_1(\mathbf{x}) = \frac{d_2(\mathbf{x})d_3(\mathbf{x})/q_2q_3}{d_1(\mathbf{x})d_2(\mathbf{x})/q_1q_2 + d_1(\mathbf{x})d_3(\mathbf{x})/q_1q_3 + d_2(\mathbf{x})d_3(\mathbf{x})/q_2q_3} \ , \text{ etc.} \tag{4.5}$$

4.2.2 The Joint Distance Function

We denote the constant in (4.2) by $D(\mathbf{x})$, a function of $\mathbf{x}$. Since the probabilities

$$p_k(\mathbf{x}) = \frac{D(\mathbf{x})}{d_k(\mathbf{x})/q_k} \ , \ k = 1, \ldots, K,$$

add to 1 we get,

$$D(\mathbf{x}) = \frac{\prod_{j=1}^{K} \frac{d_j(\mathbf{x})}{q_j}}{\sum_{i=1}^{K} \prod_{j \neq i} \frac{d_j(\mathbf{x})}{q_j}} \ . \tag{4.6}$$

$D(\mathbf{x})$ is called the **joint distance function** of $\mathbf{x}$, and is, up to a constant, the harmonic mean of the K weighted distances $\{d_k(\mathbf{x})/q_k\}$. $D(\mathbf{x})$ has the dimension of distance.

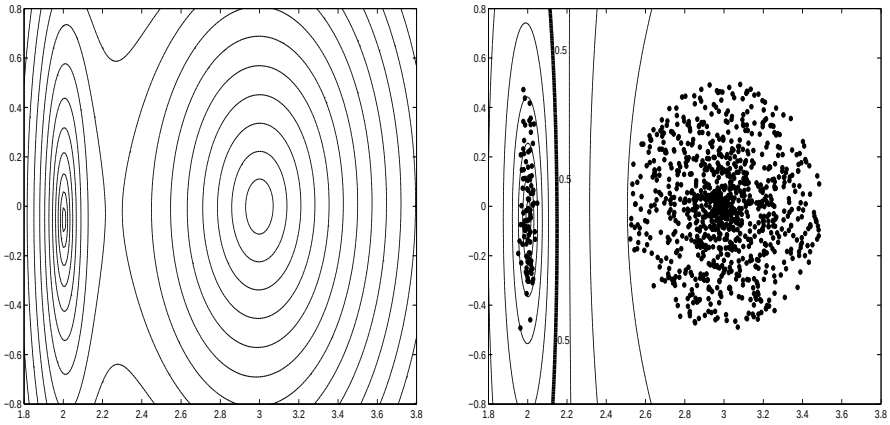
Special cases: for $K = 2$,

$$D(\mathbf{x}) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x})/q_1q_2}{d_1(\mathbf{x})/q_1 + d_2(\mathbf{x})/q_2} \ , \tag{4.7}$$

and $K = 3$,

$$D(\mathbf{x}) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x})\, d_3(\mathbf{x})/q_1q_2q_3}{d_1(\mathbf{x})\, d_2(\mathbf{x})/q_1q_2 + d_1(\mathbf{x})\, d_3(\mathbf{x})/q_1q_3 + d_2(\mathbf{x})\, d_3(\mathbf{x})/q_2q_3} \ . \tag{4.8}$$

Example 4.2. Figure 4.2(a) shows level sets of the joint distance function (4.7) for the



(a) Level sets of the joint distance function (b) Probability level sets

Figure 4.2: Results for the data of Example 4.1

data of Example 4.1.

4.2.3 An Extremal Principle

Equation (4.2) may be derived from an extremal principle. For notational simplicity we consider the case of 2 clusters, with analogous results readily available for several clusters.

Let $\mathbf{x}$ be a given data point with distances $d_1(\mathbf{x})$, $d_2(\mathbf{x})$ to the cluster centers, and assume the cluster sizes q_1, q_2 known. Then the probabilities in (4.4) are the optimal solutions of the extremal problem

$$\min \left\{ \frac{d_1(\mathbf{x})\, p_1^2}{q_1} + \frac{d_2(\mathbf{x})\, p_2^2}{q_2} : \, p_1 + p_2 = 1, \, p_1, \, p_2 \geq 0 \right\} . \tag{4.9}$$

Indeed, the Lagrangian of this problem is

$$L(p_1, p_2, \lambda) = \frac{d_1(\mathbf{x})\, p_1^2}{q_1} + \frac{d_2(\mathbf{x})\, p_2^2}{q_2} - \lambda(p_1 + p_2 - 1) \, , \tag{4.10}$$

and zeroing the partials $\partial L / \partial p_i$ gives the principle (4.2).

Substituting the probabilities (4.4) in the Lagrangian (4.10) we get the optimal value of (4.9)

$$L^*(p_1(\mathbf{x}), p_2(\mathbf{x}), \lambda) = \frac{d_1(\mathbf{x})\, d_2(\mathbf{x}) / q_1 q_2}{d_1(\mathbf{x})/q_1 + d_2(\mathbf{x})/q_2} \, , \tag{4.11}$$

which is again the joint distance function (4.7).

The corresponding extremal problem for the data set $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ is

$$\begin{aligned} \min \quad & \sum_{i=1}^{N} \left(\frac{d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2}{q_1} + \frac{d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2}{q_2} \right) \\ \text{s.t.} \quad & p_1(\mathbf{x}_i) + p_2(\mathbf{x}_i) = 1 \ , \\ & p_1(\mathbf{x}_i),\, p_2(\mathbf{x}_i) \geq 0 \ , \ i = 1, \ldots, N \ , \end{aligned} \tag{4.12}$$

where $p_1(\mathbf{x}_i), p_2(\mathbf{x}_i)$ are the cluster probabilities at $\mathbf{x}_i$ and $d_1(\mathbf{x}_i), d_2(\mathbf{x}_i)$ are the corresponding distances. The problem separates into N problems like (4.9), and its optimal value is

$$\sum_{i=1}^{N} \frac{d_1(\mathbf{x}_i)\, d_2(\mathbf{x}_i)/q_1 q_2}{d_1(\mathbf{x}_i)/q_1 + d_2(\mathbf{x}_i)/q_2} \ , \tag{4.13}$$

the sum of the joint distance functions of all points.

Note: An explanation for the terms p_k^2 (squares of probabilities) in the problem (4.9) is that this problem is a smoothed version of the "real" problem, $\min\{d_1, d_2\}$, which is non–smooth, see [79] for this and other smoothing schemes.

4.2.4 An Extremal Principle for the Cluster Sizes

Taking the cluster sizes as variables in the extremal principle (4.12),

$$\min \left\{ \sum_{i=1}^{N} \left(\frac{d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2}{q_1} + \frac{d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2}{q_2} \right) : \ q_1 + q_2 = N,\ q_1,\ q_2 \geq 0 \right\}$$

with $p_1(\mathbf{x}_i), p_2(\mathbf{x}_i)$ assumed known, we have the Lagrangian

$$L(q_1, q_2, \lambda) = \sum_{i=1}^{N} \left(\frac{d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2}{q_1} + \frac{d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2}{q_2} \right) + \lambda(q_1 + q_2 - N)$$

Zeroing the partials $\partial L / \partial q_k$ gives,

$$q_k^2 = \frac{1}{\lambda} \left(\sum_{i=1}^{N} d_k(\mathbf{x}_i)\, p_k(\mathbf{x}_i)^2 \right), \ k = 1, 2 \ , \tag{4.14}$$

showing that the cluster size q_k is proportional to $\sqrt{\sum_{i=1}^{N} d_k(\mathbf{x}_i)\, p_k(\mathbf{x}_i)^2}$. This holds for any number of clusters. In particular, for 2 clusters we have,

$$q_1 = N\, \frac{\left(\sum_{i=1}^{N} d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2\right)^{1/2}}{\left(\sum_{i=1}^{N} d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2\right)^{1/2} + \left(\sum_{i=1}^{N} d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2\right)^{1/2}}\ , \tag{4.15a}$$

$$q_2 = N - q_1\ , \tag{4.15b}$$

since $q_1 + q_2 = N$.

4.2.5 Centers

Dealing first with the case of 2 clusters, we rewrite (4.12) as a function of the cluster centers,

$$f(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \left(\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)\, p_1(\mathbf{x}_i)^2}{q_1} + \frac{d_2(\mathbf{x}_i, \mathbf{c}_2)\, p_2(\mathbf{x}_i)^2}{q_2} \right) \tag{4.16}$$

and look for centers $\mathbf{c}_1, \mathbf{c}_2$ minimizing f.

Theorem 4.2. Let the distance functions d_1, d_2 in (4.16) be elliptic,

$$d(\mathbf{x}, \mathbf{c}_k) = \big\langle (\mathbf{x} - \mathbf{c}_k), Q_k(\mathbf{x} - \mathbf{c}_k) \big\rangle^{1/2},\ k = 1, 2, \tag{4.17}$$

where Q_1, Q_2 are positive definite, so that

$$f(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \left(\sqrt{\big\langle (\mathbf{x}_i - \mathbf{c}_1), Q_1(\mathbf{x}_i - \mathbf{c}_1) \big\rangle}\, \frac{p_1(\mathbf{x}_i)^2}{q_1} + \sqrt{\big\langle (\mathbf{x}_i - \mathbf{c}_2), Q_2(\mathbf{x}_i - \mathbf{c}_2) \big\rangle}\, \frac{p_2(\mathbf{x}_i)^2}{q_2} \right)\ , \tag{4.18}$$

and let the probabilities $p_k(\mathbf{x}_i)$ and cluster sizes q_k be given. If the minimizers $\mathbf{c}_1, \mathbf{c}_2$ of (4.18) do not coincide with any of the data points $\mathbf{x}_i$, they are given by

$$\mathbf{c}_1 = \sum_{i=1}^{N} \Big(\frac{u_1(\mathbf{x}_i)}{\sum_{t=1}^{N} u_1(\mathbf{x}_t)} \Big) \mathbf{x}_i\ , \quad \mathbf{c}_2 = \sum_{i=1}^{N} \Big(\frac{u_2(\mathbf{x}_i)}{\sum_{t=1}^{N} u_2(\mathbf{x}_t)} \Big) \mathbf{x}_i\ , \tag{4.19}$$

where

$$u_1(\mathbf{x}_i) = \frac{\left(\frac{d_2(\mathbf{x}_i,\mathbf{c}_2)}{q_2}\right)^2 \frac{1}{d_1(\mathbf{x}_i,\mathbf{c}_1)}}{\left(\frac{d_1(\mathbf{x}_i,\mathbf{c}_1)}{q_1} + \frac{d_2(\mathbf{x}_i,\mathbf{c}_2)}{q_2}\right)^2}\ , \quad u_2(\mathbf{x}_i) = \frac{\left(\frac{d_1(\mathbf{x}_i,\mathbf{c}_1)}{q_1}\right)^2 \frac{1}{d_2(\mathbf{x}_i,\mathbf{c}_2)}}{\left(\frac{d_1(\mathbf{x}_i,\mathbf{c}_1)}{q_1} + \frac{d_2(\mathbf{x}_i,\mathbf{c}_2)}{q_2}\right)^2}\ , \tag{4.20}$$

or equivalently, in terms of the probabilities (4.4),

$$u_1(\mathbf{x}_i) = \frac{p_1(\mathbf{x}_i)^2}{d_1(\mathbf{x}_i,\mathbf{c}_1)}\ , \quad u_2(\mathbf{x}_i) = \frac{p_2(\mathbf{x}_i)^2}{d_2(\mathbf{x}_i,\mathbf{c}_2)}\ . \tag{4.21}$$

Proof. The gradient of $d(\mathbf{x},\mathbf{c}) = \big\langle(\mathbf{x}-\mathbf{c}), Q(\mathbf{x}-\mathbf{c})\big\rangle^{1/2}$ with respect to $\mathbf{c}$ is

$$\nabla_{\mathbf{c}} \big\langle(\mathbf{x}-\mathbf{c}), Q(\mathbf{x}-\mathbf{c})\big\rangle^{1/2} = -\frac{Q(\mathbf{x}-\mathbf{c})}{\big\langle(\mathbf{x}-\mathbf{c}), Q(\mathbf{x}-\mathbf{c})\big\rangle^{1/2}} = -\frac{Q(\mathbf{x}-\mathbf{c})}{d(\mathbf{x},\mathbf{c})}\ , \tag{4.22}$$

assuming $\mathbf{x} \neq \mathbf{c}$. Therefore if $\mathbf{c}_1, \mathbf{c}_2$ do not coincide with any of the data points $\mathbf{x}_i$, we have

$$\nabla_{\mathbf{c}_k} f(\mathbf{c}_1,\mathbf{c}_2) = -Q_k \sum_{i=1}^{N} \frac{(\mathbf{x}_i - \mathbf{c}_k)}{d_k(\mathbf{x}_i,\mathbf{c}_k)} \frac{p_k(\mathbf{x}_i)^2}{q_k}\ . \tag{4.23}$$

Setting the gradient equal to zero, "cancelling" the matrix Q_k and the common factor q_k, and summing like terms, we get

$$\sum_{i=1}^{N} \left(\frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i,\mathbf{c}_k)}\right) \mathbf{x}_i = \left(\sum_{i=1}^{N} \frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i,\mathbf{c}_k)}\right) \mathbf{c}_k\ ,$$

proving (4.19) and (4.21). Substituting (4.4) in (4.21) then gives (4.20). □

Note: The theorem holds also if a center coincides with a data point, if we interpret ∞/∞ as 1 in (4.19).

Theorem 4.2 applies, in particular, to the Mahalanobis distance (4.1)

$$d(\mathbf{x},\mathbf{c}_k) = \sqrt{(\mathbf{x}-\mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{x}-\mathbf{c}_k)}\ ,$$

where Σ_k is the (given or computed) covariance matrix of the cluster $\mathcal{C}_k$.

For the general case of K clusters it is convenient to use the probabilistic form (4.21).

Corollary 4.1. Consider a function of K centers

$$f(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K) = \sum_{k=1}^{K} \sum_{i=1}^{N} \Big(\frac{d_k(\mathbf{x}_i, \mathbf{c}_k)\, p_k(\mathbf{x}_i)^2}{q_k} \Big) , \tag{4.24}$$

an analog of (4.16). Then, under the hypotheses of Theorem 4.2, the minimizers of f are

$$\mathbf{c}_k = \sum_{i=1}^{N} \Big(\frac{u_k(\mathbf{x}_i)}{\sum\limits_{t=1}^{N} u_k(\mathbf{x}_t)} \Big) \mathbf{x}_i \ , \quad \text{with } u_k(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)^2}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \ , \tag{4.25}$$

for $k = 1, \ldots, K$.

Proof. Same as the proof of Theorem 4.2. □

Note: Formula (4.25) is an optimality condition for the centers $\mathbf{c}_k$, expressing them as convex combinations of the data points $\mathbf{x}_i$, with weights $u_k(\mathbf{x}_i)$ depending on the centers $\mathbf{c}_k$. It is used iteratively in Step 3 of Algorithm 4.1 below to update the centers, and is an extension to several facilities of the well–known Weiszfeld iteration for facility location, see [60], [82]. This formula, and the corresponding formulas (4.15) for the cluster sizes, are applied in [46] for solving multi–facility location problems, subject to capacity constraints.

4.2.6 The Centers and the Joint Distance Function

The centers obtained in Theorem 4.2 are stationary points for the joint distance function (4.13), written as a function of the cluster centers $\mathbf{c}_1, \mathbf{c}_2$,

$$F(\mathbf{c}_1, \mathbf{c}_2) = \sum_{i=1}^{N} \frac{\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)\, d_2(\mathbf{x}_i, \mathbf{c}_2)}{q_1 q_2}}{\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)}{q_1} + \frac{d_2(\mathbf{x}_i, \mathbf{c}_2)}{q_2}} \ . \tag{4.26}$$

Theorem 4.3. Let the distances $d_k(\mathbf{x}_i, \mathbf{c}_k)$ in (4.26) be elliptic. Then the stationary points of the function F are $\mathbf{c}_1, \mathbf{c}_2$ given by (4.19)–(4.21).

Proof. Using (4.22) we derive,

$$\nabla_{\mathbf{c}_1} F(\mathbf{c}_1, \mathbf{c}_2) =$$
$$= \frac{1}{q_1 q_2} \sum_{i=1}^{N} \frac{\left(\frac{d_1(\mathbf{x}_i)}{q_1} + \frac{d_2(\mathbf{x}_i)}{q_2}\right) d_2(\mathbf{x}_i) \left(-\frac{Q_1(\mathbf{x}_i - \mathbf{c}_1)}{d_1(\mathbf{x}_i)}\right) + d_1(\mathbf{x}_i)\, d_2(\mathbf{x}_i) \frac{1}{q_1} \left(\frac{Q_1(\mathbf{x}_i - \mathbf{c}_1)}{d_1(\mathbf{x}_i)}\right)}{\left(\frac{d_1(\mathbf{x}_i)}{q_1} + \frac{d_2(\mathbf{x}_i)}{q_2}\right)^2}$$
$$= \sum_{i=1}^{N} \frac{\frac{d_2(\mathbf{x}_i)^2}{q_2} \left(-\frac{Q_1(\mathbf{x}_i - c_1)}{d_1(\mathbf{x}_i)}\right)}{\left(\frac{d_1(\mathbf{x}_i)}{q_1} + \frac{d_2(\mathbf{x}_i)}{q_2}\right)^2} \tag{4.27}$$

Setting $\nabla_{\mathbf{c}_1} F(\mathbf{c}_1, \mathbf{c}_2)$ equal zero, and summing like terms, we obtain the center $\mathbf{c}_1$ as in (4.19)–(4.21). The statements about $\mathbf{c}_2$ are proved similarly. □

4.3 The PDQ Algorithm

The above results are used in an algorithm for unsupervised clustering of data, called the **PDQ Algorithm** (**P** for probability, **D** for distance and **Q** for the cluster sizes).

For simplicity, we describe the algorithm for the case of 2 clusters.

Algorithm 4.1. The PDQ Algorithm.

Initialization:	given data set $\mathcal{D}$ with N points,
	any two centers $\mathbf{c}_1, \mathbf{c}_2$,
	any two cluster sizes $q_1, q_2,\ q_1 + q_2 = N$,
	$\epsilon > 0$
Iteration:	
Step 1	**compute** distances from $\mathbf{c}_1, \mathbf{c}_2$ for all $\mathbf{x} \in \mathcal{D}$
Step 2	**update** the cluster sizes $\mathbf{q}_1^+, \mathbf{q}_2^+$ (using (4.15))
Step 3	**update** the centers $\mathbf{c}_1^+, \mathbf{c}_2^+$ (using (4.19)–(4.20))
Step 4	**if** $\|\mathbf{c}_1^+ - \mathbf{c}_1\| + \|\mathbf{c}_2^+ - \mathbf{c}_2\| < \epsilon$ **stop**
	return to Step 1

The algorithm iterates between the **cluster size estimates** (4.15), the cluster **centers** (4.19) expressed as minimizers of the objective function (4.18), and the **distances**

of the data points to these centers.

Notes:

(a) The distances used in Step 1 are elliptic, and may be different functions, depending on the cluster.

(b) In particular, if the Mahalanobis distance (4.1)

$$d(\mathbf{x}, \mathbf{c}_k) = \sqrt{(\mathbf{x} - \mathbf{c}_k)^T \Sigma_k^{-1} (\mathbf{x} - \mathbf{c}_k)}$$

is used, the covariance matrix Σ_k of the k^{th}–cluster can be estimated at each iteration by

$$\Sigma_k = \frac{\sum_{i=1}^{N} u_k(\mathbf{x}_i)(\mathbf{x}_i - \mathbf{c}_k)(\mathbf{x}_i - \mathbf{c}_k)^T}{\sum_{i=1}^{N} u_k(\mathbf{x}_i)} , \tag{4.28}$$

with $u_k(\mathbf{x}_i)$ given by (4.20).

(c) If the cluster sizes q_1, q_2 are known, they are used as the initial estimates and are not updated thereafter, in other words Step 2 is absent.

(d) The computations stop (in Step 4) when the centers stop moving, at which point the cluster membership probabilities may be computed by (4.4). These probabilities are not needed by the algorithm, and may be used afterwards for classifying the data.

(e) Having the probabilities corresponding to the final centers, rigid clusters can be determined, and used to refine the estimates of the covariance matrices.

(f) Step 3 of the algorithm is a generalization of the Weiszfeld iteration, [82], to several centers. As in the classical case, to establish convergence it is necessary to modify the gradient in question, if a center coincides with one of the data points, see [59], [54]. However, the set of initial centers for which such a modification ever becomes necessary is denumerable, and this issue can be safely ignored in practice.

Example 4.3. Figure 4.2(b) shows probability level sets for the data of Example 4.1 as determined by (4.4), using the centers and covariances computed by Algorithm 4.1.

4.4 Conclusions

The PDQ Algorithm is a probabilistic clustering method based on distances (of data points from cluster centers) and on the cluster sizes. At each iteration the method updates the cluster centers, and the cluster sizes (if unknown.) The method uses cheap iterations, and converges fast.

We present two different applications of PDQ Method in the following chapters. An important application is estimating the parameters of a mixture of distributions. In this problem, the PDQ Method may serve as an alternative to the EM Method, or as a preprocessor giving the EM Method a good start. In section 6.2 we apply the algorithm to the estimation of the parameters of Gaussian mixtures, and compare it to the EM method. Some numerical results are given in section 6.3. The reader find the details in Chapter 6.

Another application of PDQ Method introduced in chapter 7 is the multi-facility location problems where the cluster sizes are known. The method is a generalization to several facilities of the classical Weiszfeld Method.

Chapter 5

Clustering Validity and Joint Distance Function

5.1 Introduction

Clustering is perceived as an unsupervised process(see chapter 2) since there are no predefined classes and no examples that would show what kind of desirable relations should be valid among the data. As a consequence, the final partitions of a data set require some sort of evaluation in most applications [71]. For instance questions like "how many clusters are there in the data set?", "does the resulting clustering scheme fits our data set?", "is there a better partitioning for our data set?" call for clustering results validation and are the subjects of a number of methods discussed in the literature. They aim at the quantitative evaluation of the results of the clustering algorithms and are known under the general term *cluster validity* methods.

It is obvious that a problem we face in clustering is to decide the optimal number of clusters that fits a data set. In most algorithms' experimental evaluations 2D-data sets are used in order that the reader is able to visually verify the validity of the results (i.e., how well the clustering algorithm discovered the clusters of the data set). It is clear that visualization of the data set is a crucial verification of the clustering results. In the case of large multidimensional data sets (e.g. more than three dimensions) effective visualization of the data set would be difficult. Moreover the perception of clusters using available visualization tools is a difficult task for humans that are not accustomed to higher dimensional spaces.

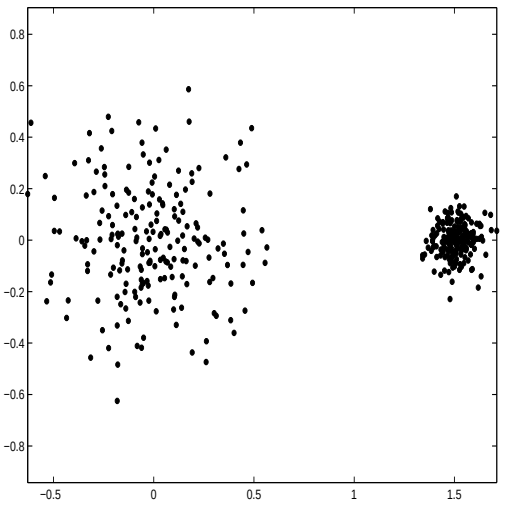
5.2 JDF as a Validity Criterion

The joint distance function (see 3.2.2) helps resolve the issue of cluster validity. Indeed, the value of the JDF decreases monotonically with K, the number of clusters, and the decrease is precipitous (which appears as "knee") until the "right" number is reached, and after that the rate of decrease is small. This is illustrated in Example 5.1 and Figures 5.1–5.3 below. The synthetically generated 2D data sets are used in order that the results can be verified visually.

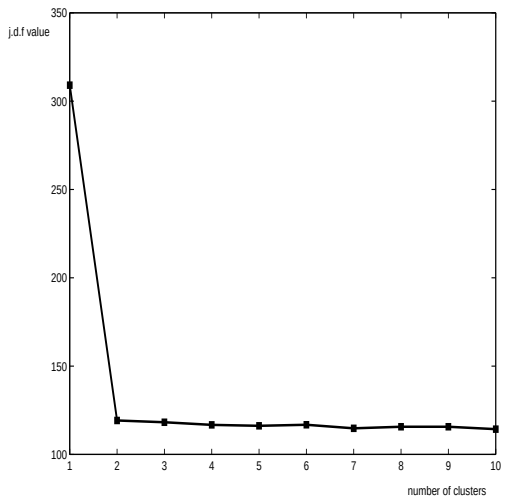
This approach is useful because the PDQ algorithm is fast, and clustering for several values of K is feasible if finding the correct K is important.

Example 5.1. Figure 5.1(a) shows a data set with 2 clusters. The PDQ algorithm was applied to this data set, and the values of the JDF are computed for values of K from 1 to 10, the results are plotted in Figure 5.1(b). Note the change of slope of the JDF at $K = 2$, the correct number of clusters.

Figures 5.2(a) and 5.3(a) show similarly data sets with $K = 3$ and $K = 4$ clusters, respectively. The JDF, computed by the PDQ algorithm, shown in Figures 5.2(b) and 5.3(b), reveal the correct number of clusters.

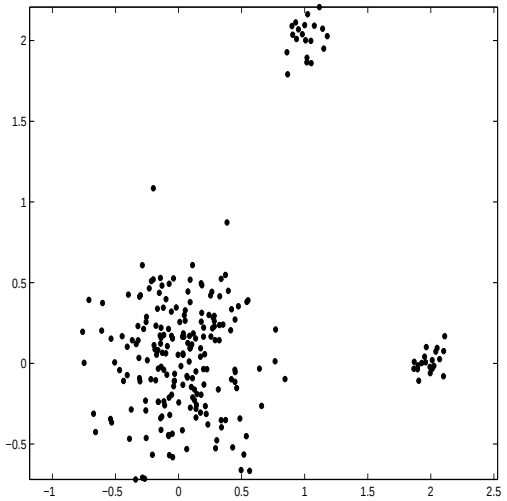


(a) A data set with $K = 2$ clusters

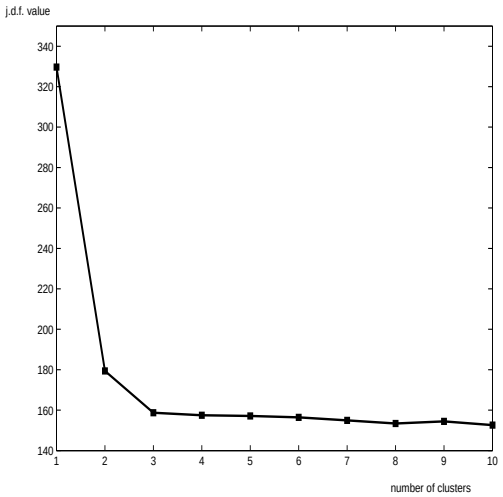


(b) The JDF as a function of K

Figure 5.1: Results of Example 5.1 for 2 clusters.

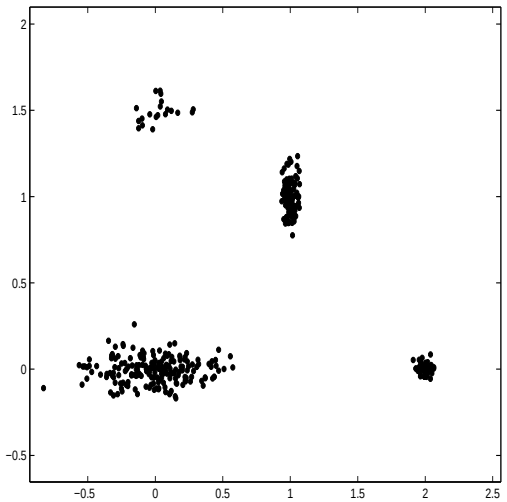


(a) A data set with $K = 3$ clusters

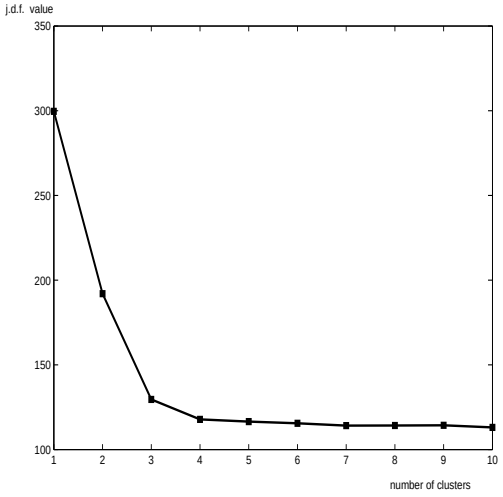


(b) The JDF as a function of K

Figure 5.2: Results of Example 5.1 for 3 clusters



(a) A data set with $K = 4$ clusters

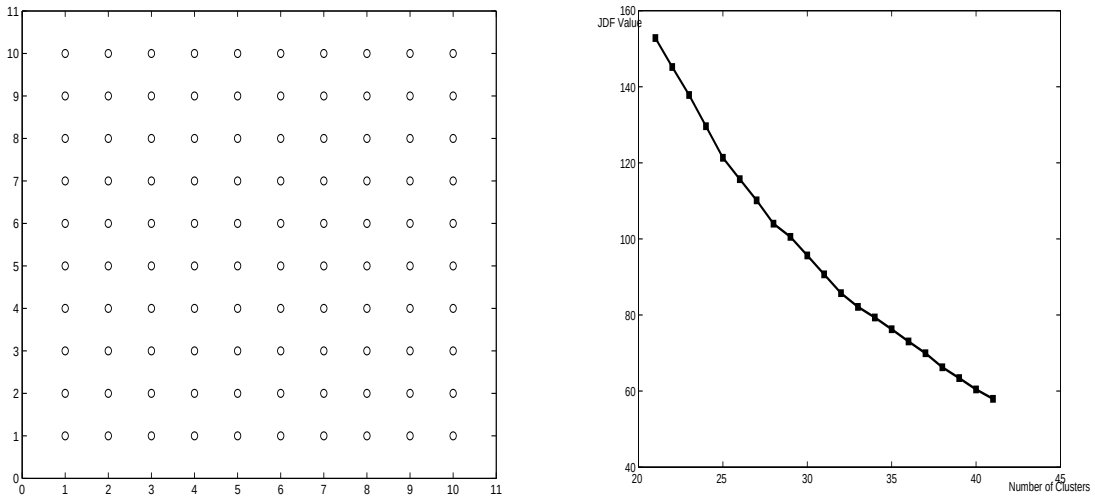


(b) The JDF as a function of K

Figure 5.3: Results of Example 5.1 for 4 clusters

The following examples illustrate that the JDF decreases monotonically and there is no significant change in its value (which appears as a "knee") if the data set don't have a cluster structure.

Example 5.2. Figure 5.4(a) shows a data set without a cluster structure. The PDQ algorithm was applied to this data set, and the values of the JDF are computed for values of K from 1 to 42, the results are plotted in Figure 5.4(b). Note there is no significant change of slope of the JDF.



(a) A data set with no cluster structure

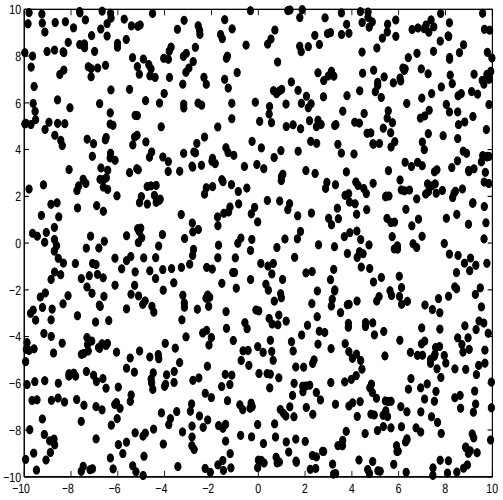
(b) The JDF as a function of K

Figure 5.4: The change of slope of the JDF in example 5.2

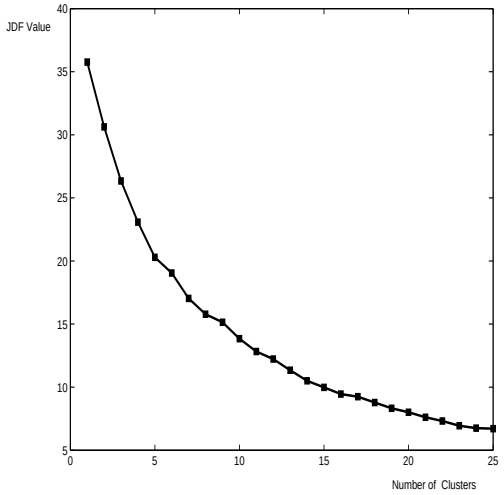
Example 5.3. Figure 5.5(a) shows the data set of Example 7.2 with $N = 1000$ random points in $[-10, 10]^2$ points without a cluster structure. The values of the JDF are computed for different values of K and the results are plotted in Figure 5.5(b). Note there is no significant change of slope of the JDF.

5.3 Other Approaches to Cluster Validity Problem

The general approach, called as *relative criteria*, to clustering validity is the evaluation of a clustering structure by comparing it to other clustering schemes, resulting by the same algorithm but with different input parameter values.



(a) The data set of Example 5.3



(b) The JDF as a function of K

Figure 5.5: The change of slope of the JDF in example 5.3

The fundamental idea of this approach is to choose the best clustering scheme of a set of defined schemes according to a pre–specified criterion. More specifically, the problem can be stated as follows in [37]:

> "*Let* $\mathbf{P}_{alg}$ *be the set of parameters associated with a specific clustering algorithm (e.g. the number of clusters* K*). Among the clustering schemes* $\mathcal{S}_i$*,* $i = 1, \ldots, K$*, defined by a specific algorithm, for different values of the parameters in* $\mathbf{P}_{alg}$*, choose the one that best fits the data set.*"

Then, we can consider the following cases of the problem:

I) $\mathbf{P}_{alg}$ **does not contain the number of clusters,** K**, as a parameter.** In this case, the choice of the optimal parameter values are described as follows: We run the algorithm for a wide range of its parameters values and we choose the largest range for which K remains constant (usually $K << N$ (number of data points)). Then we choose as appropriate values of the $\mathbf{P}_{alg}$ parameters the values that correspond to the middle of this range. Also, this procedure identifies the number of clusters that underlie our data set.

II) $\mathbf{P}_{alg}$ **contains** K **as a parameter.** The procedure of identifying the best clustering scheme is based on a validity index. Selecting a suitable performance index,

Δ, we proceed with the following steps:

- the clustering algorithm is run for all values of K between a minimum K_{min} and a maximum K_{max}. The minimum and maximum values have been defined a-priori by user.
- For each of the values of K, the algorithm is run r times, using different set of values for the other parameters of the algorithm (e.g. different initial conditions).
- The best values of the index Δ obtained by each K is plotted as the function of K.

Based on this plot we may identify the best clustering scheme. We have to stress that there are two approaches for defining the best clustering depending on the behavior of Δ with respect to K. Thus, if the validity index does not exhibit an increasing or decreasing trend as K increases we seek the maximum (minimum) of the plot. On the other hand, for indices that increase (or decrease) as the number of clusters increase we search for the values of K at which a significant local change in value of the index occurs. This change appears as a "knee" in the plot and it is an indication of the number of clusters underlying the data set. Moreover, the absence of a knee may be an indication that the data set possesses no clustering structure.

In the following subsections, some representative validity indices for crisp (hard) and soft (fuzzy) clustering (see section 2.2.3) are presented.

5.4 Crisp Clustering Indices

Crisp(hard) clustering, considers non–overlapping partitions meaning that a data point either belongs to a cluster or not. In this section we introduce validity indices suitable for crisp clustering.

5.4.1 The Modified Hubert Γ Statistic

The definition of the modified Hubert Γ [80] statistic is given by the equation

$$\Gamma = (1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} P(i,j)Q(i,j) \tag{5.1}$$

where N is the number of data points in a dataset, $M = N(N-1)/2$, P is the proximity matrix of the data set, whose (i,j) element is the distance between the data points $\mathbf{x}_i$ and $\mathbf{x}_j$, and Q is an $N \times N$ matrix whose (i,j) element is equal to the distance between the centers of the clusters where the data points $\mathbf{x}_i$ and $\mathbf{x}_j$ belong respectively.

The modified Hubert Γ statistic describes the degree of a partition fitting the data set. We note, only when two data points lie in different clusters, they have an effect on the value of Γ, otherwise, they do not contribute to the Γ because $Q(i,j) = 0$. When all data lie in a cluster, Γ is equal to 0, and with the partition number increasing, the more non–zero elements are in the matrix Q, the higher is the value of Γ

Similarly, we can define the normalized Hubert Γ statistic, given by the equation

$$\hat{\Gamma} = \frac{[(1/M) \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (P(i,j) - \mu_P)(Q(i,j) - \mu_Q)]}{\sigma_P \sigma_Q} \tag{5.2}$$

where $P(i,j)$ and $Q(i,j)$ are the (i,j) element of the matrices P and Q respectively that we have to compare. Also $\mu_P, \mu_Q, \sigma_P, \sigma_Q$ are the respective means and variances of P, Q matrices. This index takes values between -1 and 1.

For two data points $\mathbf{x}_i$ and $\mathbf{x}_j$, $i, j = 1, \ldots, N$, belonging different clusters, if the distance between them is close to that between the centers of clusters which they belong to respectively, it is indicated that the data points in a cluster are close to their center the values of Γ and $\hat{\Gamma}$(normalized $\hat{\Gamma}$) will be high. A high value of Γ (and $\hat{\Gamma}$) indicates the existence of compact clusters. Thus, in the plot of normalized Γ versus K , we seek a significant "knee" that corresponds to a significant increase of normalized Γ. The number of clusters at which the knee occurs is an indication of the number of clusters that occurs in the data. We note that for $K = 1$ and $K = N$, the index is not defined.

5.4.2 Dunn Family of Indices

A cluster validity index for crisp clustering proposed in [28], aims at the identification of "compact and well separated clusters". The index is defined in equation (5.3) for a specific number of clusters

$$D_K = \min_{k=1,\ldots,K} \left\{ \min_{t=k+1,\ldots,K} \left(\frac{d(\mathcal{C}_k, \mathcal{C}_t)}{\max_{k=1,\ldots,K} \text{diam}(\mathcal{C}_k)} \right) \right\} \tag{5.3}$$

where $d(\mathcal{C}_k, \mathcal{C}_t)$ is the dissimilarity function between two clusters $\mathcal{C}_k$ and $\mathcal{C}_t$ defined as $d(\mathcal{C}_k, \mathcal{C}_t) = \min_{\mathbf{x}\in\mathcal{C}_k, \mathbf{y}\in\mathcal{C}_t} d(\mathbf{x}, \mathbf{y})$, and $\text{diam}(\mathcal{C}_k)$ is the diameter of a cluster, which may be considered as a measure of clusters' dispersion. The diameter of a cluster $\mathcal{C}_k$ can be defined as follows:

$$\text{diam}(\mathcal{C}_k) = \max_{\mathbf{x},\mathbf{y}\in\mathcal{C}} d(\mathbf{x}, \mathbf{y}) \tag{5.4}$$

If the dataset contains compact and well–separated clusters, the distance between the clusters is expected to be large and the diameter of the clusters is expected to be small. Thus, based on the Dunn's index definition, we may conclude that large values of the index indicate the presence of compact and well-separated clusters.

Index D_K does not exhibit any trend with respect to number of clusters. Thus, the maximum in the plot of D_K versus the number of clusters can be an indication of the number of clusters that fits the data.

However, it is very difficult to evaluate the clustering validity by the Dunn index directly because of its considerable time complexity and its sensitivity to the presence of noise in data sets.

In the literature, three indices, are proposed in [70] that are known as Dunn–like indices since they are based on Dunn index. Moreover, these three indices use, for their definition, the concepts of Minimum Spanning Tree (MST), the relative neighbourhood graph (RNG) and the Gabriel graph(GG) respectively [80].

5.4.3 The Davies–Bouldin(DB) Index.

A similarity measure R_{kt} between the clusters $\mathcal{C}_k$ and $\mathcal{C}_t$ is defined based on a *measure of dispersion*, s_k of a cluster $\mathcal{C}_k$ and a *dissimilarity measure*, d_{kt} between between the clusters $\mathcal{C}_k$ and $\mathcal{C}_t$. The R_{kt} index is defined to satisfy the following conditions:

1. $R_{kt} \geq 0$
2. $R_{kt} = R_{tk}$
3. if $s_k = 0$ and $s_t = 0$ then $R_{kt} = 0$
4. if $s_k > s_t$ and d_{lk}=d_{lt} then $R_{lk} > R_{lt}$
5. $s_k = s_t$ and $d_{lk} < d_{lt}$ then $R_{lk} > R_{lt}$.

These conditions state that R_{kt} is non–negative and symmetric.

A simple choice for R_{kt} that satisfies the above conditions is

$$R_{kt} = (s_k + s_t)/d_{kt}. \tag{5.5}$$

Then the DB index is defined as

$$\begin{aligned} DB_K &= \frac{1}{K}\sum_{k=1}^{K} R_k \\ R_k &= \max_{\substack{t=1,\ldots,K \\ t\neq k}} R_{kt}, \qquad k = 1,\ldots,K \end{aligned} \tag{5.6}$$

It is clear for the above definition that DB_K is the average similarity between each cluster $\mathcal{C}_k$, $k = 1,\ldots,K$ and its most similar one. It is desirable for the clusters to have the minimum possible similarity to each other; therefore we seek clusterings that minimize DB. The DB_K index exhibits no trends with respect to the number of clusters and thus we seek the minimum value of DB_K in its plot versus the number of clusters.

Some alternative definitions of the dissimilarity between two clusters as well as the dispersion of a cluster, $\mathcal{C}_k$ is defined in [23].

Three variants of the DB_K index are proposed in [70]. They are also based on MST, RNG and GG concepts similarly to the cases of the Dunn–like indices.

5.4.4 RMSSDT, SPR, RS,CD

This family of validity indices is applicable in the cases that hierarchical clustering algorithms are used to cluster the data sets. Here, we introduce the definitions of four validity indices, which have to be used simultaneously to determine the number of clusters existing in the data set. These four indices are applied to each step of a hierarchical clustering algorithm and they are known as [74]:

- *Root–mean–square standard deviation* (RMSSTD) of the new cluster
- *Semi-partial R-squared* (SPR)
- *R-squared* (RS)
- *Distance between two clusters* (CD).

The *Root–mean–square standard deviation*(RMSSTD) of a new clustering scheme defined at a level of a clustering hierarchy is the square root of the variance of all the variables (attributes used in the clustering process). This index measures the homogeneity of the formed clusters at each step of the hierarchical algorithm. Since the objective of cluster analysis is to form homogeneous groups the RMSSTD of a cluster should be as small as possible. In case that the values of RMSSTD are higher than the ones of the previous step, we have an indication that the new clustering scheme is worse.

In the following definitions we shall use the term SS, which means *sum of squares* and refers to the equation:

$$SS = \sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}})^2 \tag{5.7}$$

Along with this we shall use some additional notation like:

i) SS_w referring to the sum of squares within group,

ii) SS_b referring to the sum of squares between groups,

iii) SS_t referring to the total sum of squares, of the whole data set.

Semi-partial R-squared (SPR) for a the new cluster is the difference between SS_w of the new cluster and the sum of the SS_w values of the clusters joined to obtain the new cluster (*loss of homogeneity*), divided by the SS_t for the whole data set. This index measures the loss of homogeneity after merging the two clusters of a single algorithm step. If the index value is zero then the new cluster is obtained by merging two perfectly homogeneous clusters. If its value is high then the new cluster is obtained by merging two heterogeneous clusters.

R-squared(RS) of the new cluster is the ratio of SS_b over SS_t. SS_b is a measure of difference between groups. Since $SS_t = SS_b + SS_w$, the greater the SS_b the smaller the SS_w and vise versa. As a result, the greater the differences between groups are the more homogenous each group is and vise versa. Thus, RS may be considered as a measure of dissimilarity between clusters. Furthermore, it measures the degree of homogeneity between groups. The values of RS range between 0 and 1. In the case that the value of RS is zero indicates that no difference exists among groups. On the other hand, when RS equals 1 there is an indication of significant difference among groups.

The *Distance between two clusters* (CD) index measures the distance between the two clusters that are merged in a given step of the hierarchical clustering. This distance depends on the selected representatives for the hierarchical clustering we perform. For instance, in case of centroid hierarchical clustering the representatives of the formed clusters are the centers of each cluster, so CD is the distance between the centers of the clusters. In the case that we use *single linkage* CD measures the minimum Euclidean distance between all possible pairs of points, whereas in *complete linkage* CD is the maximum Euclidean distance between all pairs of data points.

Using these four indices we determine the number of clusters that exist in a data set, plotting a graph of all these indices values for a number of different stages of the clustering algorithm. In this graph we search for the steepest knee, or in other words, the greater jump of these indices values from higher to smaller number of clusters.

In the case of nonhierarchical clustering (i.e. k-means) it is also possible to use some of these indices in order to evaluate the resulting clustering. The indices that are more meaningful to use in this case are RMSSTD and RS. The idea, here, is to run

the algorithm a number of times for different number of clusters each time. Then the respective graphs of the validity indices is plotted for these clusterings and we search for the significant "knee" in these graphs. The number of clusters at which the "knee" is observed indicates the optimal clustering for the data set.

5.4.5 The SD Validity Index

The *SD validity index* [37] definition is based on the concepts of *average scattering* for clusters and *total separation* between clusters. Below, we give the fundamental definition for this index.

The *average scattering* for clusters is defined as

$$Scatt(K) = \frac{1}{K}\sum_{k=1}^{K} \|\sigma(\mathbf{c}_k)\| / \|\sigma(\mathcal{D})\| \tag{5.8}$$

where $\sigma(\mathbf{c}_k) = \frac{1}{N_k}\sum_{i=1}^{N_k}(\mathbf{x}_i - \mathbf{c}_k)^2$ is the variance of cluster k and $\sigma(\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})^2$ is the variance of the data set.

The definition of *total separation (scattering)* between clusters is given by the following equation

$$Dis(K) = \frac{D_{max}}{D_{min}}\sum_{k=1}^{K}(\sum_{t=1}^{K}\|\mathbf{c}_k - \mathbf{c}_t\|)^{-1} \tag{5.9}$$

where $D_{max} = \max(\|\mathbf{c}_k - \mathbf{c}_t\|),\ \forall k,t \in \{1,\ldots,K\}$ is the maximum distance between cluster centers and $D_{min} = \min(\|\mathbf{c}_k - \mathbf{c}_t\|),\ \forall k,t \in \{1,\ldots,K\}$ is the minimum distance between cluster centers.

Now, we can define a validity index based on equations (5.8) and (5.9) as follows

$$SD(K) = a\, Scatt(K) + Dis(K) \tag{5.10}$$

where a is a weighting factor equal to $Dis(K_{max})$ where K_{max} is the maximum number of input clusters.

The first term in equation (5.10) indicates the average compactness of the clusters (i.e., intra–cluster distances). A small value for this term indicates compact clusters

and as the scattering within clusters increases (i.e., they become less compact) the value of $Scatt(K)$ also increases. The second term, $Dis(K)$, indicates the total separation between the K clusters (i.e., an indication of inter–cluster distances). Contrary to the first term, the second term, $Dis(K)$, is influenced by the geometry of the clusters and increase with the number of clusters. The two terms of SD are of the different range, thus a weighting factor is needed in order to incorporate both terms in a balanced way. The number of clusters, K, that minimizes the above index is an optimal value.

5.5 Soft Clustering Indices

In this section, we present validity indices suitable for soft clustering. The objective is to seek clustering schemes where most of the vectors of the dataset exhibit high degree of membership in one cluster. As it is presented in chapter 2.4, soft(fuzzy) clustering is defined by a matrix $\mathcal{U} = [u_{ik}]$, where u_{ik} denotes the degree of membership of the vector $\mathbf{x}_i$ in cluster k. Similarly to crisp(hard) clustering case a validity index, Δ, is defined and we search for the minimum or maximum in the plot of Δ versus K. Also, in case that Δ exhibits a trend with respect to the number of clusters, we seek a significant knee of decrease (or increase) in the plot of Δ.

We will discuss two categories of soft validity indices. The first category uses only the memberships values, u_{ij}, of a soft partition of data. The second involves both the $\mathcal{U}$ matrix and the dataset itself.

5.5.1 Validity Indices Involving the Membership Values

Bezdek proposed in [13] the *partition coefficient*, which is defined as

$$PC = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ik}^2 \tag{5.11}$$

where N is the number of data points and K is the number of clusters.

The PC index values range in $[1/K, 1]$. The closer to unity the index the "crisper" the clustering is. In case that all membership values to a soft partition are equal, that is, $u_{ik} = 1/K$, the PC obtains its lower value. Thus, the closer the value of PC is to

$1/K$, the fuzzier the clustering is. Furthermore, a value close to $1/K$ indicates that there is no clustering tendency in the considered data set or the clustering algorithm failed to reveal it.

The partition entropy coefficient is another index of this category. It is defined as follows

$$PE = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} u_{ik}\, log_a(u_{ik}) \tag{5.12}$$

where a is the base of the logarithm. The index is computed for values of K greater than 1 and its values ranges in $[0, log_a K]$. The closer the value of PE to 0, the crisper the clustering is. As in the previous case, index values close to the upper bound (i.e., $log_a K$), indicate absence of any clustering structure in the data set or inability of the algorithm to extract it.

The drawbacks of these indices are [37]:

(i) their monotonous dependency on the number of clusters. Thus, we seek significant knees of increase (for PC) or decrease (for PE) in the plots of the indices versus the number of clusters,

(ii) their sensitivity to the fuzzifier, m in fuzzy clustering. The fuzzifier is a parameter of the fuzzy clustering algorithm and indicates the fuzziness of clustering results. Then, as $m \to 1$ the indices give the same values for all values of K. On the other hand when $m \to \infty$, both PC and PE exhibit significant knee at $K = 2$.

(iii) the lack of direct connection to the geometry of the data [22], since they do not use the data itself.

5.5.2 Indices Involving the Membership Values and the Dataset

In this section, we introduce three indices; *Xie–Beni* index, *Fuguyama-Sugeno* index and indices based on concept of *hypervolume* and *density*.

5.5.3 Xie–Beni Index

The Xie–Beni index , *XB index* [83] is also called the compactness and separation validity function, uses the membership values and the data set.

Consider a fuzzy partition of the data set $\mathcal{D} = \{\mathbf{x}_i; i = 1, \ldots, N\}$ with $\mathbf{c}_k\,(k = 1, \ldots, K)$ the centers of each cluster and u_{ik} the membership of data point i with regards to cluster k. The fuzzy deviation of $\mathbf{x}_i$ from cluster k is defined as the distance, d_{ik}, between $\mathbf{x}_i$ and the center of cluster k, weighted by the fuzzy membership of data point i belonging to cluster k.

$$d_{ik} = u_{ik}\|\mathbf{x}_i - \mathbf{c}_k\|$$

Also, for a cluster k, the sum of the squares of fuzzy deviation of the data point in $\mathcal{D}$, $\sigma_k = \sum_{i=1}^{N} d_{ik}$, is called variation of cluster k. The sum of the variations of all clusters, $\sigma = \sum_{k=1}^{K} \sigma_k$, is called *total variation of the data set.*

The term $\phi = (\sigma_k/N_k)$, is called compactness of data set $\mathcal{D}$. The less its value, the more compact clusters are.

The separation of the fuzzy partitions is defined as the minimum distance between cluster centers, that is

$$d_{min} = \min_{\substack{1\le k,t\le K \\ k\neq t}} \|\mathbf{c}_k - \mathbf{c}_t\|^2$$

Then *XB index* is defined as

$$XB = \phi/(N\, d_{min}) = \frac{\sum_{i=1}^{N}\sum_{k=1}^{K} u_{ik}^2\|\mathbf{x}_i - \mathbf{c}_k\|^2}{N \min\limits_{\substack{1\le k,t\le K \\ k\neq t}} \|\mathbf{c}_k - \mathbf{c}_t\|^2} \tag{5.13}$$

where N is the number of data points in the data set.

It is clear that small values of XB are expected for compact and well–separated clusters. We note, however, that XB is monotonically decreasing when the number of clusters K gets very large and close to N.

5.5.4 Fukuyama–Sugeno Index

. Another index of this category is the *Fukuyama–Sugeno index*, which is defined as

$$FS_m = \sum_{i=1}^{N}\sum_{k=1}^{K} u_{ik}^m \left(\|\mathbf{x}_i - \mathbf{c}_k\|_A^2 - \|\mathbf{c}_k - \bar{\mathbf{x}}\|_A^2 \right) \tag{5.14}$$

where $\bar{\mathbf{x}}$ is the mean vector of $\mathcal{D}$ and A is a positive definite, symmetric matrix. When $A = I$, the above distance becomes the Euclidean distance. It is clear that for compact and well–separated clusters we expect small values for FS_m. The first term in brackets measures the compactness of the clusters while the second one measures the distances of the clusters representatives.

5.5.5 Indices Based on Hypervolume and Density

. Other soft validity indices are proposed in [34], which are based on the concepts of *hypervolume* and *density*. Let Σ_k the fuzzy covariance matrix of the k^{th} cluster defined as

$$\Sigma_k = \frac{\sum_{i=1}^{N} u_{ik}^m (\mathbf{x}_i - \mathbf{c}_k)(\mathbf{x}_i - \mathbf{c}_k)^T}{\sum_{i=1}^{N} u_{ik}^m} \tag{5.15}$$

The *fuzzy hyper volume* of k^{th} cluster is given by equation:

$$V_k = \|\Sigma_k\|^{1/2} \tag{5.16}$$

where $\|\Sigma_k\|$ is the determinant of the Σ_k and is the measure of cluster compactness.

Then the total *fuzzy hyper volume* (FH) is given by the equation

$$FH = \sum_{k=1}^{K} V_k \tag{5.17}$$

Small values of FH indicate the existence of compact clusters.

The *average partition density* (PA) can also used as an index of this category which is defined as follows:

$$PA = \frac{1}{K}\sum_{k=1}^{K} \frac{S_k}{V_k} \tag{5.18}$$

Then $S_k = \sum_{\mathbf{x} \in \mathcal{D}_k} u_{ik}$, where $\mathcal{D}_k$ is the set of data points that are within a pre–specified region around $\mathbf{c}_k$, is called the sum of the central members of the cluster k.

A few other indices are proposed and discussed in [58, 65].

Chapter 6

Mixtures of Distributions and PDQ Algorithm

6.1 Introduction

Given observations from a density $\phi(\mathbf{x})$, that is itself a mixture of two densities,

$$\phi(\mathbf{x}) = \pi\,\phi_1(\mathbf{x}) + (1-\pi)\,\phi_2(\mathbf{x})\ , \tag{6.1}$$

it is required to estimate the weight π, and the relevant parameters of the distributions ϕ_1 and ϕ_2.

A common situation is when the distribution ϕ is a mixture of normal distributions ϕ_k, each with its mean $\mathbf{c}_k$ and covariance Σ_k that need to be estimated,

$$\phi_k(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n|\Sigma_k|}}\,\exp\left\{-\tfrac{1}{2}\,(\mathbf{x}-\mathbf{c}_k)^T\Sigma_k^{-1}(\mathbf{x}-\mathbf{c}_k)\right\}\ ,\ k=1,2. \tag{6.2}$$

A well–known method for de–mixing distributions is the EM Method, [40]. The PDQ Algorithm is a viable alternative to that method.

6.2 Estimation of Parameters in Mixtures of Distributions

For the purpose of comparison with the PDQ Algorithm, we present here in schematic form the EM Method for a Gaussian mixture (6.1)–(6.2).

Algorithm 6.1. The EM Method.

Initialization: given data set $\mathcal{D}$ with N points,

initial guesses for the parameters $\hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2$, $\hat{\Sigma}_1, \hat{\Sigma}_2, \hat{\pi}$

Iteration:

Step 1: For all $\mathbf{x}_i \in \mathcal{D}$ **compute** the "responsibilities" :

$$p_1(\mathbf{x}_i) = \frac{\hat{\pi}\phi_1(\mathbf{x}_i)}{\hat{\pi}\phi_1(\mathbf{x}_i) + (1-\hat{\pi})\phi_2(\mathbf{x}_i)} ,$$

$$p_2(\mathbf{x}_i) = 1 - p_1(\mathbf{x}_i) .$$

Step 2 **update** the centers and covariances:

$$\hat{\mathbf{c}}_k = \sum_{i=1}^{N} \left(\frac{p_k(\mathbf{x}_i)}{\sum_{j=1}^{N} p_k(\mathbf{x}_j)} \right) \mathbf{x}_i,$$

$$\hat{\Sigma}_k = \sum_{i=1}^{N} \left(\frac{p_k(\mathbf{x}_i)}{\sum_{j=1}^{N} p_k(\mathbf{x}_j)} \right) (\mathbf{x}_i - \hat{\mathbf{c}}_k)(\mathbf{x}_i - \hat{\mathbf{c}}_k)^T, \; k = 1, 2$$

Step 3 **update** the mixing probabilities (weights):

$$\hat{\pi} = \frac{\sum_{i=1}^{N} p_1(\mathbf{x}_i)}{N}$$

Step 4 **stop** or **return** to Step 1

Notes:

(a) The "responsibilities" in Step 1 correspond to the cluster membership probabilities in Algorithm 4.1.

(b) Step 1 requires both the Mahalanobis distance (4.1) and the evaluation of the density (6.2).

(c) Step 2 is computationally similar to Step 3 of Algorithm 4.1.

(d) The stopping rule (Step 4) is again the convergence of centers as in Algorithm 4.1. For further details see, e.g., Hastie et al [40].

6.2.1 A Comparison of the PDQ Algorithm and the EM Method

(a) The EM Algorithm is based on maximum likelihood, and therefore depends on the

density functions in the mix, requiring different computations for different densities. The PDQ Algorithm is parameter free, making no assumptions about the densities, and using the same formulas in all cases.

(b) In each EM iteration the density functions must be evaluated, requiring (in Step 1) KN function evaluations, where K is the number of densities in the mixture. In comparison, the PDQ iterations are cheaper, requiring no function evaluations.

(c) Because the EM iterations are costly, it is common to use another method, e.g., the K–means method, as a preprocessor, to get closer to the centers before starting EM. The PDQ Algorithm need no preprocessing, and works well from a cold start.

(d) If correct assumptions are made about the mixing distributions, then the EM method has an advantage over the PDQ method, as illustrated in Example 6.3 below.

(e) While the numerical comparison of the two algorithms should best be done by others, our preliminary tests show the two algorithms to be roughly equivalent in terms of the returned results, with the PDQ Algorithm somewhat faster.

6.3 Numerical Examples

In Examples 6.3–6.3 below the PDQ and EM Algorithms were applied to the same data, in order to compare their performance. The results are reported in Tables 6.1–6.4. These examples are typical representatives of many numerical tests we did.

Both programs used here were written in MATLAB, the EM code by Tsui [81], and the PDQ code by the first author.

The comparison is subject to the following limitations:

(a) The EM program code [81] uses the K–means method (Hartigan [39]) as a preprocessor to get a good start. The number of iterations, and running time, reported for this program (in Table 6.4) is just for the EM part, not including the preprocessing by the K–means part.

	True Parameters	The PDQ Algorithm (Algorithm 4.1)	The EM Method (Algorithm 6.1)
Centers	$\boldsymbol{\mu}_1$=(2 , 0) $\boldsymbol{\mu}_2$=(3 , 0)	$\hat{\mathbf{c}}_1$=(2.0036 , -0.0542) $\hat{\mathbf{c}}_2$=(2.9993 , -0.0010)	$\hat{\mathbf{c}}_1$=(2.0011 , -0.0284) $\hat{\mathbf{c}}_2$=(3.0033 , -0.0018)
Covariance Matrices	Σ_1=$\begin{pmatrix} 0.0005 & 0 \\ 0 & 0.5 \end{pmatrix}$ Σ_2=$\begin{pmatrix} 0.0402 & 0.0014 \\ 0.0014 & 0.0430 \end{pmatrix}$	$\hat{\Sigma}_1$=$\begin{pmatrix} 0.0004 & -0.0001 \\ -0.0001 & 0.0446 \end{pmatrix}$ $\hat{\Sigma}_2$=$\begin{pmatrix} 0.0399 & -0.0020 \\ -0.0020 & 0.0432 \end{pmatrix}$	$\hat{\Sigma}_1$=$\begin{pmatrix} 0.0004 & -0.0001 \\ -0.0001 & 0.0442 \end{pmatrix}$ $\hat{\Sigma}_2$=$\begin{pmatrix} 0.0398 & -0.0020 \\ -0.0020 & 0.0431 \end{pmatrix}$
Weights	(0.0909 , 0.9090)	(0.0932 , 0.9068)	(0.0909 , 0.9091)

Table 6.1: A comparison of methods for the data of Example 4.1

(b) Our PDQ code is the first, un–finessed version, a verbatim implementation of Algorithm 4.1.

(c) The number of iterations depends on the stopping rule. In the PDQ Algorithm, the stopping rule is Step 4 of Algorithm 4.1, and the number of iterations will increase the smaller is ϵ. In the EM Algorithm the stopping rule does involve also the convergence of the likelihood function, and the effect of the tolerance ϵ is less pronounced.

(d) The number of iterations depends also on the initial estimates, the better the estimates – the fewer iterations will be required. In our PDQ code the initial solutions can be specified, or are randomly chosen. The EM program gets its initial solution from its K–means preprocessor.

Example 6.1. Algorithms 4.1 and 6.1 were applied to the data of Example 4.1. Both algorithms give good estimates of the true parameters, see Table 6.1. The comparison of running time and iterations is inconclusive, see Table 6.4.

Example 6.2. Consider the data set shown in Figure 6.1. The points of the right cluster were generated in a circle of diameter 1.5 centered at $\boldsymbol{\mu}_1 = (1, 0)$, using a radially symmetric distribution function, $\text{Prob}\{\|\mathbf{x} - \boldsymbol{\mu}_1\| \leq r\} = (4/3)\, r$, and the smaller cluster on the left was similarly generated in a circle of diameter 0.1 centered at $\boldsymbol{\mu}_2 = (0, 0)$. The ratio of sizes is 1:20.

The EM Method gives bad estimates of the left center, and of the weights, see Table 6.2 and the right panel of Figure 6.2. The estimates provided by the PDQ Algorithm are better, see Figure 6.2, left panel.

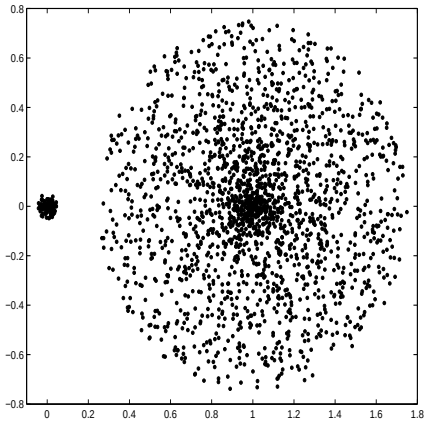


Figure 6.1: Data set of Example 6.2

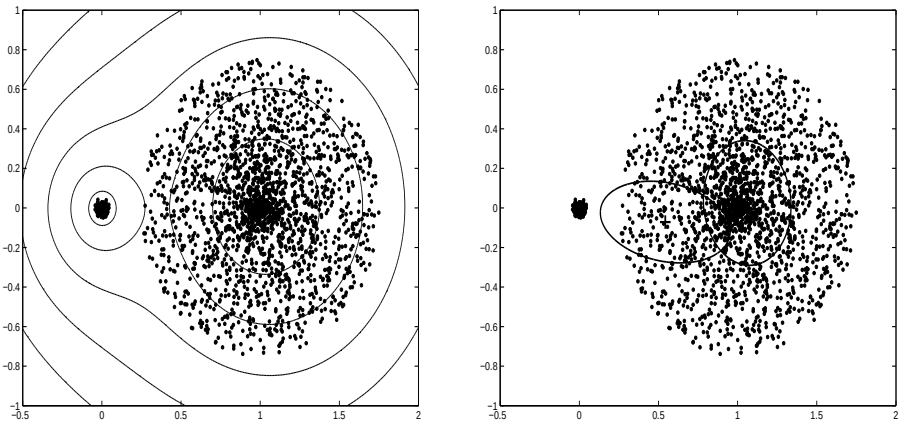


Figure 6.2: A comparison of the **PDQ Algorithm** (left), and the **EM Method** (right)

The EM Method also took longer, see Table 6.4. In repeated trials, it did not work for $\epsilon = 0.1$, and sometimes for $\epsilon = 0.01$.

Example 6.3. Consider the data set shown in Figure 6.3, left. It consists of three clusters of equal size, 200 points each, generated from Normal distributions $N(\boldsymbol{\mu}_i, \Sigma_i)$, with parameters $\boldsymbol{\mu}_i, \Sigma_i$ given in the left column of Table 6.3. A similar example appears as Fig. 9.6 in Tan et al, [78, p. 593].

As noted in section 6.2.1(d), if the assumptions on the mixing distributions are justified, the EM Method gives good estimates of the relevant parameters. The PDQ Algorithm, does not require or depend on such assumptions, and still gives decent estimates. This is illustrated in Table 6.3.

	True Parameters	The PDQ Algorithm (Algorithm 4.1)	The EM Method (Algorithm 6.1)
Centers	$\boldsymbol{\mu}_1$=(0,0) $\boldsymbol{\mu}_2$=(1,0)	$\hat{\mathbf{c}}_1$=(0.0023 ,-0.0022) $\hat{\mathbf{c}}_2$=(1.0080 , 0.0063)	$\hat{\mathbf{c}}_1$=(0.5429 ,-0.0714) $\hat{\mathbf{c}}_2$=(1.0603 , 0.02451)
Weights	(0.0476 , 0.9524)	(0.0534 , 0.9466)	(0.1851 , 0.8149)

Table 6.2: A comparison of methods for the data of Example 6.2

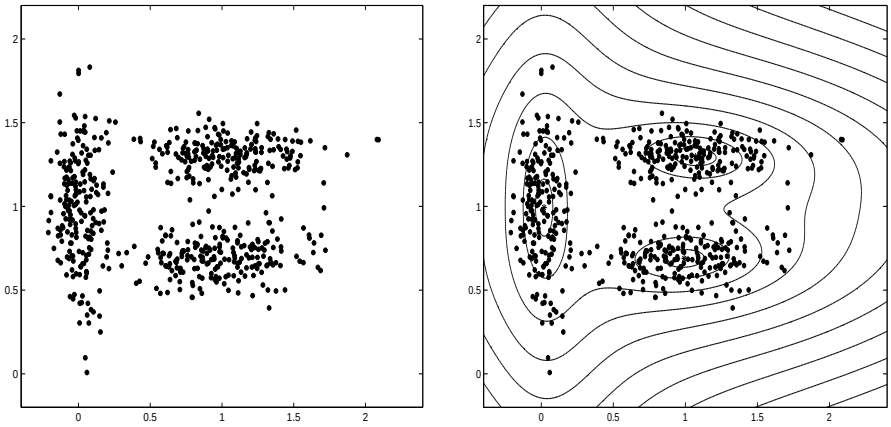


Figure 6.3: The data of Example 6.3 (left) and level sets of the joint distance function (right)

	True Parameters	The PDQ Algorithm (Algorithm 4.1)	The EM Method (Algorithm 6.1)
Centers	$\boldsymbol{\mu}_1$=(0 , 1) $\boldsymbol{\mu}_2$=(1 , 0.7) $\boldsymbol{\mu}_3$=(1 , 1.3)	$\hat{\mathbf{c}}_1$=(0.0053 , 1.0239) $\hat{\mathbf{c}}_2$=(0.9604 , 0.7146) $\hat{\mathbf{c}}_3$=(1.0735 , 1.2748)	$\hat{\mathbf{c}}_1$=(0.0049 , 0.9916) $\hat{\mathbf{c}}_2$=(0.9855 , 0.6939) $\hat{\mathbf{c}}_3$=(1.0376 , 1.3083)
Covariance Matrices	$\Sigma_1=\begin{pmatrix}0.01 & 0\\ 0 & 0.1\end{pmatrix}$ $\Sigma_2=\begin{pmatrix}0.1 & 0\\ 0 & 0.01\end{pmatrix}$ $\Sigma_3=\begin{pmatrix}0.1 & 0\\ 0 & 0.01\end{pmatrix}$	$\hat{\Sigma}_1=\begin{pmatrix}0.0134 & -0.0006\\ -0.0006 & 0.1074\end{pmatrix}$ $\hat{\Sigma}_2=\begin{pmatrix}0.0828 & 0.0023\\ 0.0023 & 0.0117\end{pmatrix}$ $\hat{\Sigma}_3=\begin{pmatrix}0.0907 & -0.0040\\ -0.0040 & 0.0123\end{pmatrix}$	$\hat{\Sigma}_1=\begin{pmatrix}0.0091 & -0.0018\\ -0.0018 & 0.1059\end{pmatrix}$ $\hat{\Sigma}_2=\begin{pmatrix}0.1012 & 0.0053\\ 0.0053 & 0.0122\end{pmatrix}$ $\hat{\Sigma}_3=\begin{pmatrix}0.0981 & -0.0005\\ -0.0005 & 0.0090\end{pmatrix}$
Weights	(0.333 , 0.333 , 0.333)	(0.3297 , 0.3345 , 0.3358)	(0.3318 , 0.3351 , 0.3331)

Table 6.3: A comparison of methods for the data of Example 6.3

Example	ϵ	PDQ Algorithm		EM Algorithm	
		Iterations	Time (sec.)	Iterations	Time (sec.)
Example 4	0.01	5	3.32	1	1.783
	0.1	2	1.42	1	1.682
Example 5	0.01	8	3.89	55	37.73
	0.1	2	1.02	9	7.28
Example 6	0.01	11	2.29	7	3.28

Table 6.4: Summary of computation results for 3 examples. See section 6.3(a) for explanation of the EM running time and iterations count.

Chapter 7

Multi–facility Location Problems

7.1 Introduction

A location problem is to locate a facility, or facilities, to serve optimally a given set of customers.

The customers are given by their coordinates and demands. The coordinates are points $\mathbf{a}$ in $\mathbb{R}^p$ (usually $n = 2$), and the demands are positive numbers w.

Assuming N customers, the data of the problem is a set of points (coordinates) $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ in $\mathbb{R}^p$ and a corresponding set of positive weights (demands) $\{w_1, w_2, \ldots, w_N\}$.

We use the Euclidean norm in $\mathbb{R}^p$

$$\|\mathbf{u}\| = \langle \mathbf{u}, \mathbf{u} \rangle^{1/2}, \tag{7.1}$$

with $\langle \cdot, \cdot \rangle$ the standard inner product, and the Euclidean distance

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| , \tag{7.2}$$

between any two points $\mathbf{x}, \mathbf{y}$ in $\mathbb{R}^p$.

If the customers are served by one facility located at $\mathbf{c}$, then the weighted sum of distances travelled by all the customers is

$$\sum_{i=1}^{N} w_i \|\mathbf{c} - \mathbf{x}_i\| .$$

The **Fermat–Weber location problem** is to find the point $\mathbf{c}$ that minimizes the

above expression, i.e.,

$$\min_{\mathbf{c}\in\mathbb{R}^p} \sum_{i=1}^{N} w_i \|\mathbf{c} - \mathbf{x}_i\| , \tag{7.3}$$

see the survey in [26].

If the customers are served by K facilities, for given K, we denote by $\mathcal{X}_k$ be the set of customers allocated (or assigned) to the k^{th}– facility. Then the weighted sum of distances travelled by these customers is

$$\sum_{\mathbf{x}_i\in\mathcal{X}_k} w_i \|\mathbf{c}_k - \mathbf{x}_i\|$$

where $\mathbf{c}_k$ is the location of the k^{th}– facility.

Given the customers $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, their demands $\{w_1, \ldots, w_N\}$ and an integer $1 < K < N$, the **Location–Allocation Problem** (LAP) (also **Multi–Facility Location Problem**) is to determine the **locations** $\{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$ of the facilities, and the **allocations** $\mathcal{X}_1, \ldots, \mathcal{X}_K$ of customers to these facilities, so as to minimize the weighted sum of distances travelled by all the customers,

$$\min_{\mathbf{c}_1,\ldots,\mathbf{c}_K} \min_{\mathcal{X}_1,\ldots,\mathcal{X}_K} \sum_{k=1}^{K} \sum_{\mathbf{x}_i\in\mathcal{X}_k} w_i \|\mathbf{c}_k - \mathbf{x}_i\| . \tag{7.4}$$

The allocation sets $\mathcal{X}_k$ are a disjoint partition of $\mathcal{X}$,

$$\mathcal{X} = \mathcal{X}_1 \cup \cdots \cup \mathcal{X}_K , \ \mathcal{X}_k \cap \mathcal{X}_t = \emptyset \text{ if } k \neq t . \tag{7.5}$$

Since the points in $\mathcal{X}_k$ are served by the same facility $\mathbf{c}_k$, we expect them to be in the proximity of that facility, and therefore close to each other. Similarly, points served by different facilities need not to be neighbors and in general are not. Using the terminology of Clustering Theory, the allocation sets $\{\mathcal{X}_k : k = 1, \ldots, K\}$ are **clusters** in $\mathcal{X}$, i.e., a disjoint partitions of $\mathcal{X}$, where each set consists of nearby points.

The Location–Allocation Problem (7.4) is therefore closely related to the **Clustering Problem**, of partitioning the set $\mathcal{X}$ into K clusters, where the locations of the facilities are at the centers of the clusters.

In some situations there are upper bounds (capacities) on the demands that a facility can handle. If the k^{th} facility has capacity Q_k, then the sum of demands allocated to it cannot exceed it,

$$\sum_{\mathbf{x}_i \in \mathcal{X}_k} w_i \leq Q_k \ . \tag{7.6}$$

The **Capacitated LAP** (CLAP) is a problem (7.4) with some capacity constraints like (7.6). In CLAP it may be necessary to split the demand of a customer between two or more facilities, so it is no longer the case that each customer takes all his business to the nearest facility.

7.2 The Fermat–Weber location problem

The problem is to find a point $\mathbf{c}$ in $\mathbb{R}^n$ that minimizes

$$f(\mathbf{c}) = \sum_{i=1}^{N} w_i \left\| \mathbf{c} - \mathbf{x}_i \right\| \ , \tag{7.7}$$

the sum of the weighted Euclidean distances between the customers $\mathbf{x}_i$ and the facility $\mathbf{c}$. The gradient of f

$$\nabla f(\mathbf{c}) = \sum_{i=1}^{N} w_i \, \frac{\mathbf{c} - \mathbf{x}_i}{\left\| \mathbf{c} - \mathbf{x}_i \right\|} \tag{7.8}$$

exists for all $\mathbf{c} \notin \mathcal{X}$. A point $\mathbf{c}^*$ is optimal iff $\mathbf{0} \in \partial f(\mathbf{c}^*)$, which reduces to $\nabla f(\mathbf{c}^*) = \mathbf{0}$ if f is differentiable at $\mathbf{c}^*$. It follows then from (7.8) that $\mathbf{c}^*$ is a convex combination of the points of $\mathcal{X}$,

$$\mathbf{c}^* = \sum_{i=1}^{N} \lambda_i(\mathbf{c}) \, \mathbf{x}_i \ , \tag{7.9}$$

with weights

$$\lambda_i(\mathbf{c}) = \frac{w_i \left\| \mathbf{c} - \mathbf{x}_i \right\|^{-1}}{\sum\limits_{j=1}^{N} w_j \left\| \mathbf{c} - \mathbf{x}_j \right\|^{-1}} \ . \tag{7.10}$$

The **Weiszfeld Method** [82] for solving this problem is an iterative method with updates

$$\mathbf{c}_+ := \sum_{i=1}^{N} \lambda_i(\mathbf{c}) \, \mathbf{x}_i \ , \tag{7.11}$$

giving the next iterate $\mathbf{c}_+$ as a convex combination, with weights $\lambda_i(\mathbf{c})$ computed by (7.10) for the current iterate $\mathbf{c}$. Note that $\lambda_i(\mathbf{c})$ is undefined if $\mathbf{c} = \mathbf{x}_i$. If the Weiszfeld iterates converge to a point $\mathbf{c}^*$, then $\mathbf{x}^*$ is optimal by (7.9).

The Weiszfeld method is the best–known method for solving the Fermat–Weber location problem, see the history in [60, section 1.3] and [26].

7.3 The Probabilistic Location-Allocation Problem and a Weiszfeld Method for the Approximate Solution of LAP

The Weiszfeld method, (7.11), expresses the facility location as a convex combination of the customers' coordinates.

The extremal principle (3.15) (see, chapter 3) is given for K clusters as,

$$\begin{aligned} \min \quad & \sum_{i=1}^{N} \left(d_1(\mathbf{x}_i)\, p_1(\mathbf{x}_i)^2 + d_2(\mathbf{x}_i)\, p_2(\mathbf{x}_i)^2 + \cdots + d_K(\mathbf{x}_i)\, p_K(\mathbf{x}_i)^2 \right) \\ \text{s.t.} \quad & p_1(\mathbf{x}_i) + p_2(\mathbf{x}_i) = 1 \\ & p_1(\mathbf{x}_i),\, p_2(\mathbf{x}_i) \geq 0 \end{aligned} \tag{7.12}$$

When $K = 1$, it reduces to

$$\min \quad \sum_{i=1}^{N} \|\mathbf{c} - \mathbf{x}_i\| \,,$$

where the probabilities are all 1 and therefore of no interest, and the centers coincides with the Weiszfeld center (7.9),

$$\mathbf{c} = \sum_{i=1}^{N} \left(\frac{u(\mathbf{x}_i)}{\sum_{j=1}^{N} u(\mathbf{x}_j)} \right) \mathbf{x}_i \,, \text{where} \quad u(\mathbf{x}_i) = \frac{1}{\|\mathbf{c} - \mathbf{x}_i\|} \,. \tag{7.13}$$

For $K > 1$ the center formulas (7.14) represent each facility as a convex combination of the customers' coordinates, which is a generalization of the Weiszfeld formula for

several facilities.

$$\mathbf{c}_k = \sum_{i=1}^{N} \left(\frac{u_k(\mathbf{x}_i)}{\sum_{j=1}^{N} u_k(\mathbf{x}_j)} \right) \mathbf{x}_i \ , \quad \text{where } u_k(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)^2}{\|\mathbf{c} - \mathbf{x}_i\|} \ . \tag{7.14}$$

e.g., for $K = 2$,

$$\mathbf{c}_1 = \sum_{i=1}^{N} \left(\frac{u_1(\mathbf{x}_i)}{\sum_{j=1}^{N} u_1(\mathbf{x}_j)} \right) \mathbf{x}_i \ , \quad \mathbf{c}_2 = \sum_{i=1}^{N} \left(\frac{u_2(\mathbf{x}_i)}{\sum_{j=1}^{N} u_2(\mathbf{x}_j)} \right) \mathbf{x}_i \ ,$$

where

$$u_1(\mathbf{x}_i) = \frac{d_2(\mathbf{x}_i)^2 / d_1(\mathbf{x}_i)}{(d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2} \ , \quad u_2(\mathbf{x}_i) = \frac{d_1(\mathbf{x}_i)^2 / d_2(\mathbf{x}_i)}{(d_1(\mathbf{x}_i) + d_2(\mathbf{x}_i))^2} \ .$$

Thus the D-Clustering Algorithm (see chapter 3) is an extension of Weiszfeld's Method when it is applied to solve LAP.

7.3.1 The Capacitated Location Allocation Problem

In PDQ method presented in chapter 4, the cluster size q_k serves as the facility capacity. Similar to (7.14), the center formulas are the convex combination of the customer locations with weights including not only the distance but also the capacity of the facility.

$$\mathbf{c}_k = \sum_{i=1}^{N} \Big(\frac{u_k(\mathbf{x}_i)}{\sum_{j=1}^{N} u_k(\mathbf{x}_j)} \Big) \mathbf{x}_i \ , \quad \text{where } u_k(\mathbf{x}_i) = \frac{p_k(\mathbf{x}_i)^2 \, q_k}{d_k(\mathbf{x}_i, \mathbf{c}_k)} \ ,$$

e.g., for $K = 2$,

$$u_1(\mathbf{x}_i) = \frac{\left(\frac{d_2(\mathbf{x}_i, \mathbf{c}_2)}{q_2} \right)^2 \frac{q_1}{d_1(\mathbf{x}_i, \mathbf{c}_1)}}{\left(\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)}{q_1} + \frac{d_2(\mathbf{x}_i, \mathbf{c}_2)}{q_2} \right)^2} \ , \quad u_2(\mathbf{x}_i) = \frac{\left(\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)}{q_1} \right)^2 \frac{q_2}{d_2(\mathbf{x}_i, \mathbf{c}_2)}}{\left(\frac{d_1(\mathbf{x}_i, \mathbf{c}_1)}{q_1} + \frac{d_2(\mathbf{x}_i, \mathbf{c}_2)}{q_2} \right)^2} \ ,$$

Thus PDQ method is an extension of the Weizsfeld Method where the cluster sizes resemble the facility capacities and normalize the distances. The PDQ algorithm presented in chapter 4 solves the **Capacitated LAP**'s and gives the approximate solutions. When capacity values are given, the PDQ Algorithm simplifies further, see section 4.3, note (c). This is illustrated in Example 7.3 and Figure 7.3 below.

7.4 Numerical Examples

Examples 7.1 and 7.2 illustrate the D-Clustering Algorithm for solving LAP's.

Example 7.1. (Cooper, [21] p. 47) It is required to locate 3 facilities to serve the following 15 customers and there is no capacity constraints for the facilities.

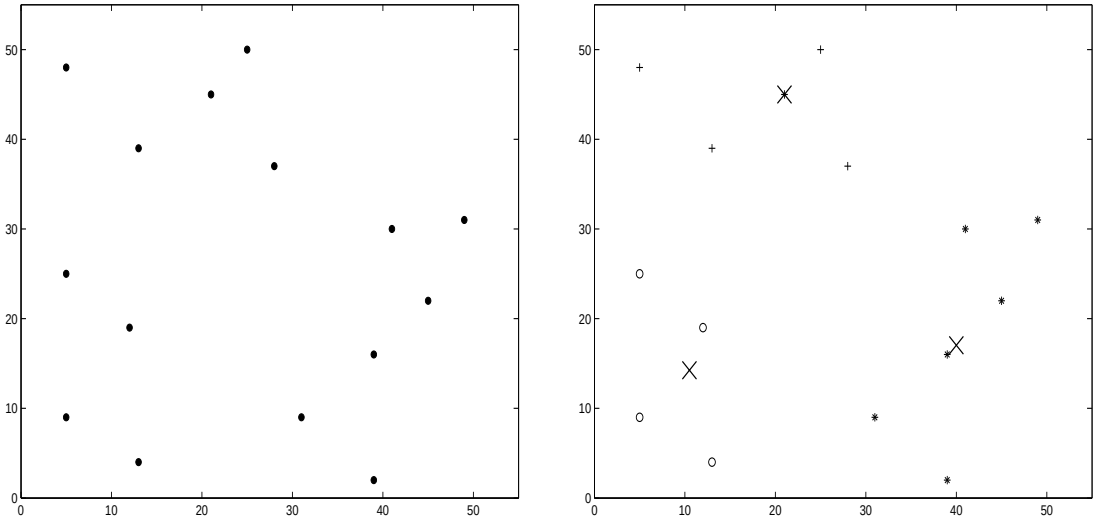
Customer	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x–coordinate	5	5	5	13	12	13	28	21	25	31	39	39	45	41	49
y–coordinate	9	25	48	4	19	39	37	45	50	9	2	16	22	30	31

Table 7.1: Data for Example 7.1

These data points are shown in Fig. 7.1(a). The PDQ algorithm, with $\epsilon = 0.001$ (in Step 4), required 14 iterations to determine the three clusters, with approximate centers. The final centers, computed after the clusters were determined (see Remark 3.4(e)), are shown in Fig. 7.1(b). In the top left cluster, the facility practically coincides with one of the customers.

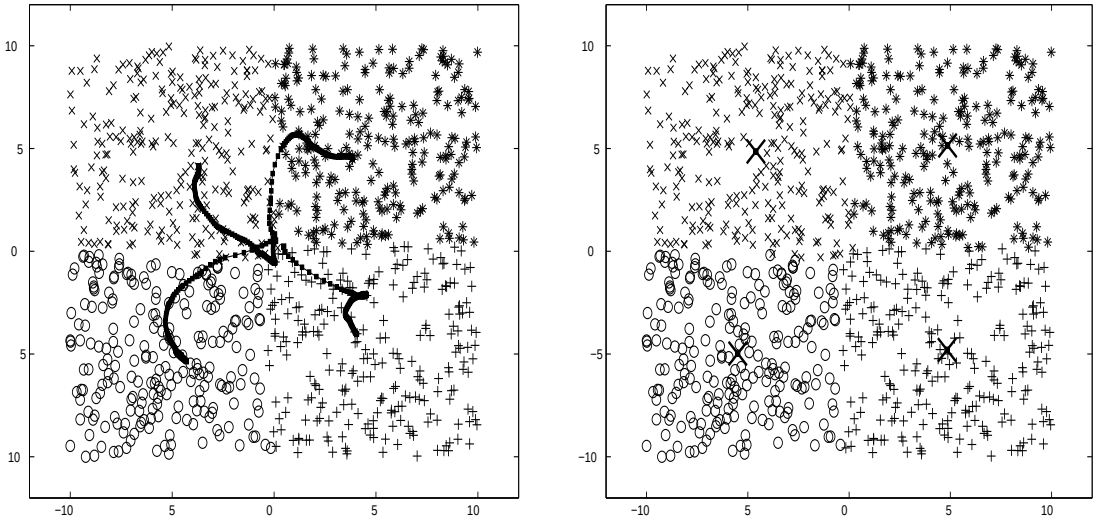
Example 7.2. Fig. 7.2 shows a data set with $N = 1000$ random points in $[-10, 10]^2$, representing the customers. It is required to locate $K = 4$ facilities to serve the customers. The algorithm starts with 4 random initial locations (centers.) Using different symbols: o, x, +, * for 4 clusters, Figure 7.2(a) illustrates the convergence from arbitrary initial points. The final clusters, obtained by truncating the cluster probabilities, allow better estimates of the facilities locations (centers), see Remark 3.4(e). Figure 7.2(b) shows the final clusters and facilities.

Example 7.3. Consider the same 1000 random data points of Example 7.2, and 4 facilities with capacities given in percentages as 35%, 25%, 15%, and 25% of the total



(a) The 15 customers (b) The final facilities and clusters

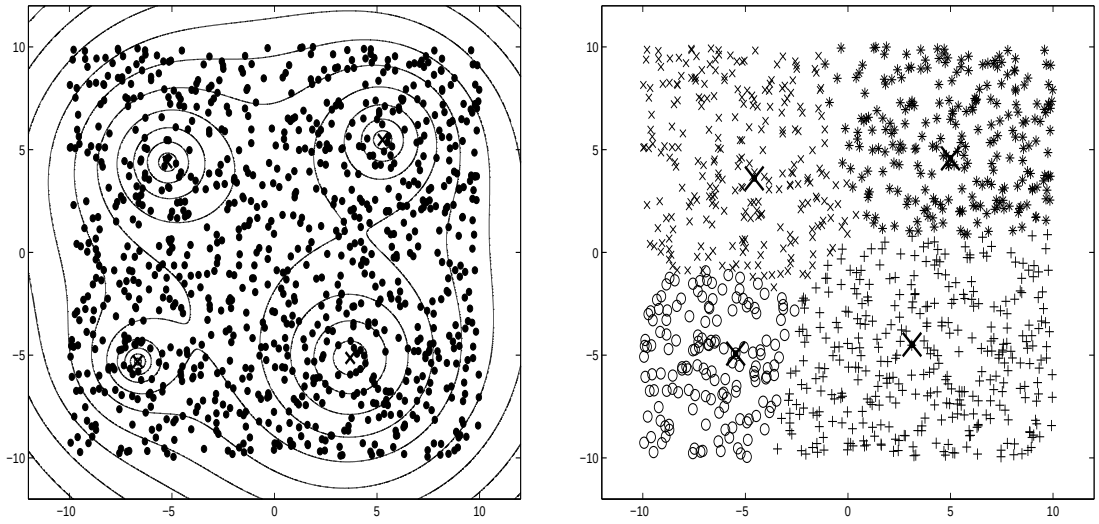
Figure 7.1: Illustration of Example 7.1



(a) Convergence to centers (b) The final centers

Figure 7.2: Results for Example 7.2

demand. The PDQ Algorithm starts with 4 random initial facilities (centers). Figure 7.3(a) shows the level sets of the JDF computed by the PDQ algorithm, and Figure 7.3(b) shows the final facilities and their clusters.



(a) Level sets of the JDF (b) Final clusters and centers

Figure 7.3: Results for Example 7.3

Chapter 8

Clustering with Similarity Data

8.1 Introduction

Many applications use similarity data, see section 2.2.6. Two examples of this type are considered below.

8.2 The Liberal-Conservative Divide of the Rehnquist Court

The **Rehnquist Supreme Court** was analyzed by Hubert and Steinley in [45], where the justices were ranked as follows, from most liberal to most conservative.

Liberals	**Conservatives**
1. John Paul Stevens (St)	5. Sandra Day O'Connor (Oc)
2. Stephen G.Breyer (Br)	6. Anthony M. Kennendy (Ke)
3. Ruth Bader Ginsberg (Gi)	7. William H. Rehnquist (Re)
4. David Souter (So)	8. Antonin Scalia (Sc)
	9. Clarence Thomas (Th)

The data used in the analysis is a 9×9 **similarity matrix**, giving the percentages of non-unanimous cases in which justices *agreed*, see Table 8.1 (a mirror image of 8.1 in [45], listing the disagreements.)

Hubert and Steinley used two methods, **unidimensional scaling** (mapping the data from $\mathbb{R}^9$ to $\mathbb{R}$), and **hierarchical classification**, see [45] for details.

We applied our method to the Rehnquist Court, with Justices represented by points $\mathbf{x}$ in $\mathbb{R}^9$ (the columns of Table 8.1), using the Euclidean distance in $\mathbb{R}^9$. Our results are given in the following table, listing the clusters and their membership probabilities.

	St	Br	Gi	So	Oc	Ke	Re	Sc	Th
1 St	1.00	.62	.66	.63	.33	.36	.25	.14	.15
2 Br	.62	1.00	.72	.71	.55	.47	.43	.25	.24
3 Gi	.66	.72	1.00	.78	.47	.49	.43	.28	.26
4 So	.63	.71	.78	1.00	.55	.50	.44	.31	.29
5 Oc	.33	.55	.47	.55	1.00	.67	.71	.54	.54
6 Ke	.36	.47	.49	.50	.67	1.00	.77	.58	.59
7 Re	.25	.43	.43	.44	.71	.77	1.00	.66	.68
8 Sc	.14	.25	.28	.31	.54	.58	.66	1.00	.79
9 Th	.15	.24	.26	.29	.54	.59	.68	.79	1.00

Table 8.1: Similarities among the nine Supreme Court justices

Cluster	Justice	Membership Probability
Liberal	Ruth Bader Ginsburg	0.8685
	David Souter	0.8390
	Stephen Breyer	0.7922
	John Paul Stevens	0.7144
Conservative	William Rehnquist	0.8966
	Anthony Kennedy	0.7540
	Clarence Thomas	0.7220
	Antonin Scalia	0.7173
	Sandra Day O'Connor	0.6740

Table 8.2: The liberal–conservative divide of the Rehnquist Court

The membership probability of a Justice in a cluster is, by equation (3.6), proportional to the proximity to the cluster center, and is thus a measure of the agreement of the Justice with others in the cluster.

Since not all non–unanimous cases were equally important, or equally revealing of ideology, we should not read into these probabilities more than is supported by the data. For example, Justice Kennedy (probability 0.7540) is not "more conservative" than Justice Scalia (probability 0.7173), but perhaps "more conformist" with the "conservative center".

Similarly, Justice Stevens, ranked "most liberal" in [45], is in our analysis the "least conformist" in the liberal cluster.

Overall, the liberal cluster is tighter, and more conformist, than the conservative cluster.

8.3 Country Dissimilarities

This example is presented in [40], page 469. The data (taken from [56]) comes from a study in which political science students were asked to provide pairwise dissimilarity measures (1–10) for 12 countries. The average dissimilarity scores are given in the following table. The abbreviations used here are

BEL: Belgium

BRA: Brazil

CHI: China

CUB: Cuba

EGY: Egypt

FRA: France

IND: India

ISR: Israel

USA: United States of America

USS: The Soviet Union, now Russian Federation

YUG: Yugoslavia, now Serbia

ZAI: Zaire, now Democratic Republic of Congo.

We construct the corresponding similarity matrix by subtracting each entry from 10. We run the D-clustering algorithm for 3 clusters. The membership function values for each data point are listed in table 8.4. Final clusters are formed based on the highest membership function values of the data points. Cluster-1 is { BEL, FRA, ISR, USA}, cluster-2 is {BRA, EGY, IND, ZAI} and cluster-3 is {CHI, CUB, USS, YUG}.

Although EGY is in the second cluster with BRA,IND, ZAI, it is also close to the first cluster. It almost falls about halfway between two clusters.

	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG	ZAI
BEL	0	5.58	7	7.08	4.83	2.17	6.42	3.42	2.5	6.08	5.25	4.75
BRA	5.58	0	6.50	7	5.08	5.75	5	5.5	4.92	6.67	6.83	3
CHI	7	6.5	0	3.83	8.17	6.67	5.58	6.42	6.25	4.25	4.5	6.08
CUB	7.08	7	3.83	0	5.83	6.92	6	6.42	7.33	2.67	3.75	6.67
EGY	4.83	5.08	8.17	5.83	0	4.92	4.67	5	4.5	6	5.75	5
FRA	2.17	5.75	6.67	6.92	4.92	0	6.42	3.92	2.25	6.17	5.42	5.58
IND	6.42	5	5.58	6	4.67	6.42	0	6.17	6.33	6.17	6.08	4.83
ISR	3.42	5.5	6.42	6.42	5	3.92	6.17	0	2.75	6.92	5.83	6.17
USA	2.5	4.92	6.25	7.33	4.5	2.25	6.33	2.75	0	6.17	6.67	5.67
USS	6.08	6.67	4.25	2.67	6	6.17	6.17	6.92	6.17	0	3.67	6.5
YUG	5.25	6.83	4.5	3.75	5.75	5.42	6.08	5.83	6.67	3.67	0	6.92
ZAI	4.75	3	6.08	6.67	5	5.58	4.83	6.17	5.67	6.5	6.92	0

Table 8.3: Dissimilarity matrix for countries

	Probability-1	Probability-2	Probability-3
BEL	**0.69398**	0.17810	0.12793
FRA	**0.71699**	0.16032	0.12270
ISR	**0.52237**	0.27169	0.20594
USA	**0.66277**	0.19733	0.13990
BRA	0.20590	**0.62920**	0.16490
EGY	0.35136	**0.39239**	0.25625
IND	0.26331	**0.43734**	0.29935
ZAI	0.17042	**0.68736**	0.14222
CHI	0.22799	0.27278	**0.49923**
CUB	0.15209	0.17664	**0.67127**
USS	0.11936	0.13206	**0.74857**
YUG	0.23153	0.23433	**0.53414**

Table 8.4: The membership function values and the final clusters of the countries

Chapter 9

Determining The Spatial Clusters Of Accidents

9.1 Introduction

This chapter deals with determining the spatial clusters of accidents along a continuous highway using different objectives. Identifying such spatial clusters of accidents according to different objectives can provide useful insights to various operational and safety issues.

The knowledge of the spatial clusters of accidents can be advantageous in the following applications:

(1) Incident management: Incidents are random events such as vehicle crashes, spilled loads and hazardous materials, vehicle disablement and other random activities that disrupt traffic flow. Timely detection, verification and clearance of incidents are of utmost importance, not only for minimizing congestion, but also for reducing the number of fatalities [69].

Within the incident management context, location of emergency service depots, the number of patrolling units and their patrolling area, and the optimal locations of tow-truck facilities are important factors that affect the incident clearance time [68]. Incident clearance is more efficient, when the responders are located closer to the incident locations. Furthermore, the location and the number of traffic surveillance units such as roadside detectors, closed-circuit cameras and call boxes also affect the incident detection time. Thus, it is beneficial to deploy this kind of equipment at locations where concentration of future incidents is expected to be highest.

(2) Accident prevention and mitigation: Traffic agencies can undertake different safety measures to reduce or eliminate incidents by identifying the features that makes

roadway segments hazardous [20, 64]. The important step is to determine which road segments require safety treatments the most. As the network size increases, the process of identifying such hot spots and prioritizing them can be an infeasible task. It is therefore useful for the agencies to automatically generate incident "hot spot" and identify their characteristics for more timely and effective safety considerations.

(3) Travel time variability: Travel time variability has emerged as a new performance measure in traffic networks. Knowing approximately how long it would take to travel between specific points is very important information for almost all drivers. Empirical evidence shows that the major cause of travel time variability is traffic incidents, including major accidents that block traffic lanes [19]. However, estimating travel time variability is not a simple task. There have been studies that have investigated which segments of the network and how many to select to measure travel time variability [6, 75, 84]. The proposed methodology can give useful guidelines as to where to measure travel time variability in a large-scale network.

Although vehicle crashes are random and non-recurrent events, the analysis of historical data shows that the frequency of vehicle crashes in space show high spatial correlation from one year to another.

In this chapter, we propose that given the network and traffic characteristics there exists an optimal spatial distribution of accident clusters along a continuous highway. Historical crash datasets can be analyzed to determine the location of these accident clusters and thus to gain valuable insight to various traffic management and safety issues describe above.

New Jersey Turnpike (NJTPK) is selected as the study network due to the availability of extensive vehicle crash dataset and the authors familiarity with the facility. However, similar datasets are readily available for most of the highways in many States, which makes the proposed approach readily applicable to almost all types of highways in the US.

9.2 Determining Accident Clusters For Different Objectives

The proposed formulation allows the variation of cluster configurations with different weights of each data point. Through the use of weight, it is possible to build clusters for different objectives of interest. For example, for accident mitigation and reduction purposes, analysts would be more interested in finding clusters of severe accidents and implement measures to alleviate the number of accidents. The proposed algorithm will be capable of build such clusters by changing the associated weight or assigning more weight to this objective. This is a unique feature of our proposed clustering algorithm that goes beyond well-known, but more limited "hot spot" identification methods mainly based on the frequencies.

9.3 Numerical Analysis

This section undertakes the problem of identifying accident clusters discussed before. The proposed clustering approach is tested using historical crash data available for NJTPK. The feasibility of the clustering approach is quantified based on the objective function of the clustering algorithm (refer:chapter-3).

9.3.1 Study Network and Data Description

NJTPK is a 148-mile toll facility. Toll collection is performed using a closed-ticked system. Each interchange in the facility has entry and exit toll plazas. Vehicles enter the facility at an interchanges entry toll plaza, and when they leave facility at another interchange they pay the toll, which is based on their entry interchange. There exist 29 operational interchanges in NJTPK with average daily traffic exceeding 500,000 vehicles. It is one of the principal north-south highway corridors in New Jersey. It is a direct connection between Delaware Memorial Bridge in the south and the George Washington Bridge, Lincoln Tunnel and Holland Tunnel to the New York City in the north. Figure 9.1 shows the NJTPK map.

The available database includes vehicle crash records between 2003 and 2005. These records are based on the reports filled out by police officers at the accident scene. Each

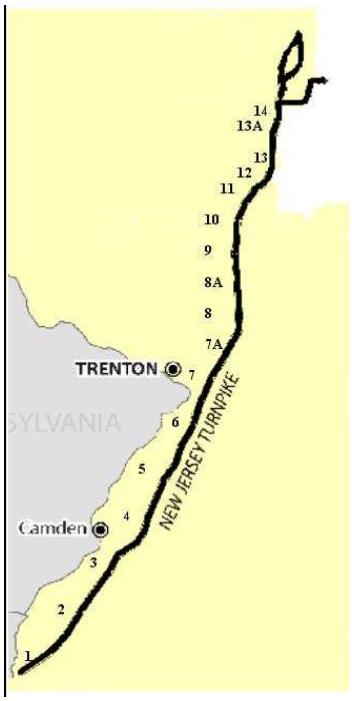


Figure 9.1: New Jersey Turnpike

record involves accident specific information such as time, day and location of the accident, how many vehicles involved, vehicle type, the degree of severity and property damage, crash type, etc.

For the brevity of the analysis, only the crashes that occurred on the mainline between interchange 1 and interchange 14 are considered. At the north of interchange 14 traffic splits up to easterly and westerly roadways, and at the east the traffic extends to Easter spur which leads to New York City. The same procedure as shown here can be repeated for the sections at the north and east of interchange 14.

Table 1 shows the summary of accidents that occurred between interchange 1 and 14.

It should be mentioned the number of accidents in NJTPK becomes higher towards the northern part of the network. For example, the total number of accidents between interchange 11 and 14 comprise of 30% of all accidents in the network. This is directly related to the higher traffic flow at the northern links. It is reasonable to assume that accident occurrence is more probable where there is higher number of vehicles traveling on a link.

	2003	2004	2005
Accidents	3,377	3,375	3,366
Number of Vehicles	5,884	5,890	5,844
Passenger Cars	4,872	4,754	4,699
Trucks	953	1,085	1,091
Buses	48	43	46
Motorcycles	11	8	8
Fatality Accidents	17	13	10
Injury Accidents	865	756	740
Property Damage Accidents	2,495	2,606	2,616
Number of Fatalities	20	12	15
Number of Injuries	1,448	1,300	1,152

Table 9.1: Summary of NJTPK accident database between interchange 1–14

9.3.2 Results

State agencies are always interested in a number of incident related decisions such as prioritizing highest locations of accident concentration, or determination of the optimal number and location of depot or the emergency response team or determining optimal number and locations of traffic surveillance units for incident detection. Let us take the first problem of the prioritization of 5 of the most serious hot spots (clusters) with the ultimate goal of implementing engineering improvements. Figure 9.2(a) shows the clustering results of this prioritization for years 2003, 2004 and 2005, where each accident data point is regarded as identical (has the same weight), i.e. $w_i = 1$ where w_i is the weight of accident-i. It can be observed from the results that the cluster centers do not fluctuate substantially for different years. The figure also shows with shaded area the sections within which accidents have more than 70% probability of being in the selected cluster, i.e. $p_k(\mathbf{x}) >= 0.70$. These areas can be named as the "hot-spots" or "points of interest" on the network.

The result that cluster centers do not vary substantially can be attributed to the similar spatial distribution of accidents for different years. A simple analysis of accident locations for three years shows that the correlations between the accidents frequencies by location are 0.976 between years 2005 and 2004, 0.975 between years 2005 and 2003, and 0.983 between years 2004 and 2003.

The probability of a crash at a given traffic facility is directly related to the number of conflict points. While conflict events differ for each traffic facility, the most common conflict events for freeways are merging flows, following flows, adjacent flows and evasive maneuvers. It is reasonable to expect accident occurrence rates obtained from historical crash data over years follow the accident occurrence probabilities at these locations. For example, accidents on US Route 1 in New Jersey are analyzed between 2001 and 2004. The data were extracted from the online accident database obtained from the New Jersey Department of Transportation [67]. The analysis shows that the frequencies of accidents by location show a similar pattern for different years. The correlation coefficients of these frequency data for different years fall between 0.97 and 0.99. These similarities in the frequency plots suggest that although accidents are random events, accident frequencies follow a pattern for a given traffic facility.

The fact that cluster centers do not vary much eliminates the question if the configuration of segments based on one year of data would fail for subsequent years.

9.3.3 Weighing Accidents

Each accident has different characteristics. One would expect that certain type of accidents have bigger impacts on traffic flow. This impact depends on several factors, of which are the number of vehicles and the type of vehicles involved in the accident. On the other hand, certain accidents are more severe than others due to the number of fatalities and injuries involved. Clearly, if one would like to weigh accidents, the purpose of weighing should be clearly defined. Here, two different objectives are considered. One is the impact of an accident on traffic flow, and the latter is the severity of accident.

Based on our correspondence with the NJTPK traffic operations group, the following accident weighting functions are formulated.

$$W_I = 10N_T + 10N_B + N_{PC} + 0.5N_M \tag{9.1}$$

$$W_{II} = 7N_F + 3N_{MajInj} + 2N_{MinInj} + N_{SI} \tag{9.2}$$

Where, N_T, N_B, N_{PC}, N_M are the number of trucks, buses, passenger cars and motorcycles involved in the accident, respectively; and $N_F, N_{MajInj}, N_{MinInj}, N_{SI}$ are the number of fatalities, major injuries, minor injuries and slight injuries, respectively.

Trucks and buses have higher parameter values because of (1) the possible load spills from trucks, (2) the special towing required to remove these vehicles, and (3) the higher number of emergency units that might be required for bus passengers.

It should be noted that these functions are by no means based on real–data. They would not completely reflect the seriousness or severity of an accident, or its impact on traffic flow. Nevertheless, they are based on the expertise and intuitiveness of the authors, and are used for the sake of the analysis here.

Probabilistic Distance Clustering algorithm is performed for the accident dataset, but this time with different weights given in equations 9.1 and 9.2. Figure 9.2(b) shows the clustering results based on weight I. Although centers of cluster 1 and 2 are in different locations compared to equal weight clustering, centers of cluster 3, 4 and 5 stay in approximately same locations (cluster numbers are in increasing order from left to right).

Figure 9.2(c) shows the results based on weight II. The center locations, in particular of cluster 1 and 2, appear in different locations compared to the locations in equal weight and weight I results.

9.3.4 Discussion

As mentioned earlier, the configuration of clusters depends on "the objective" of the analysis. The variation of cluster configurations with different weights has useful insights for various applications of interest. For example, for accident mitigation and reduction purposes, analysts would be more interested in finding clusters of severe accidents (weight II), and implement measures to alleviate the number of accidents. In some cases, low–cost design implementations, such as proper signage, markings to channelize traffic or variable speed limits can reduce certain type of accidents. Similarly, for more efficient incident management strategies, the agency would be interested in finding clusters of higher number of vehicles involved (weight I) for optimally locating tow

truck depots to reduce incident clearance times. Furthermore, clusters based on equal weight can be utilized for locating traffic surveillance units, such as roadside detectors or cameras for faster incident detection.

Similar analysis can be repeated with different variables to identify certain accident characteristics. For example, if the planners analysts are specifically concerned with reducing accidents that involve sideswiped or overturned vehicles, due to their high effect on congestion then clusters of that type of accidents can be determined by adjusting the weights of accident data points accordingly. Similarly, traffic operation center might be only interested in accidents during peak–time periods since their impact on traffic flow during those times is higher. Then higher weight $w_i$can be associated with peak–time accidents. This is very unique and useful feature of our proposed algorithm and will be further studied in the future.

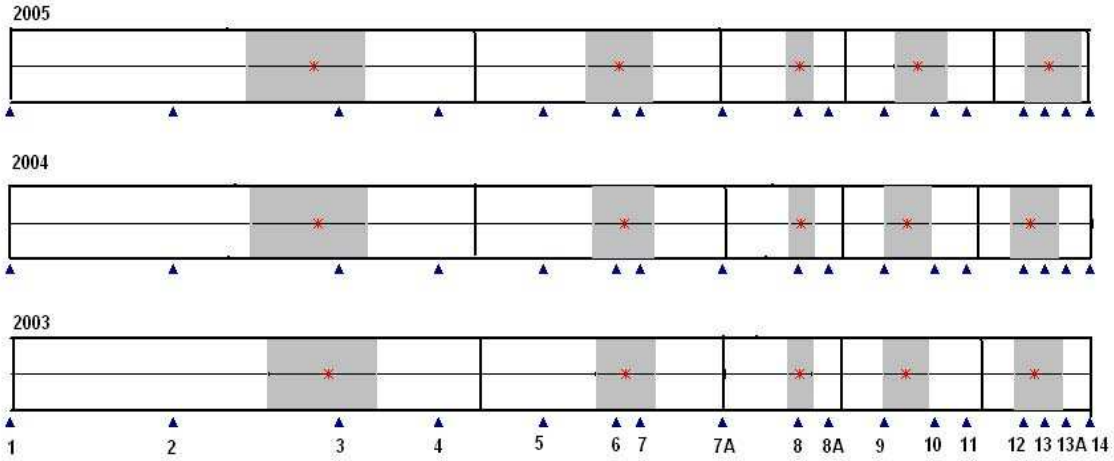
9.3.5 Determining the Optimum Number of Segments

In the analysis results presented above, only 5 roadway segments were considered. However, it is highly desirable to determine the optimum number of segments (clusters) that needs to be considered.

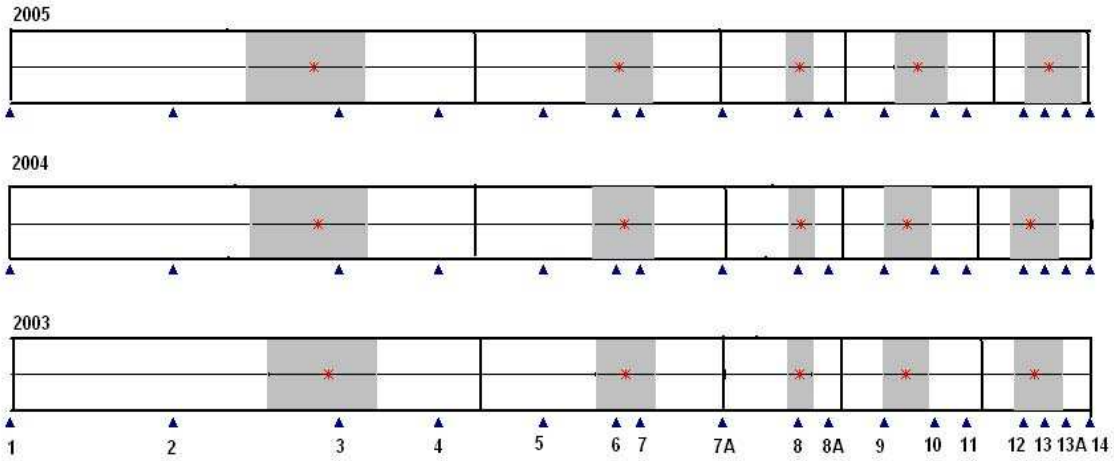
The clustering approach is also well suited for this purpose. Clustering procedure can be terminated when the cumulative gain from clustering becomes minimal. Here, we associate the marginal gain with the percent decrease in the value of the objection function by an additional cluster (see Chapter-7).

Figure 9.3 demonstrates the marginal gain versus the number of segments for each three cases, namely accident points with no weight, weight I and weight II. It can be observed that as the number of clusters increases, the marginal gain of adding a cluster decreases. It should be reasonable to terminate the process of adding new clusters, where the marginal gain is below a pre–determined threshold.

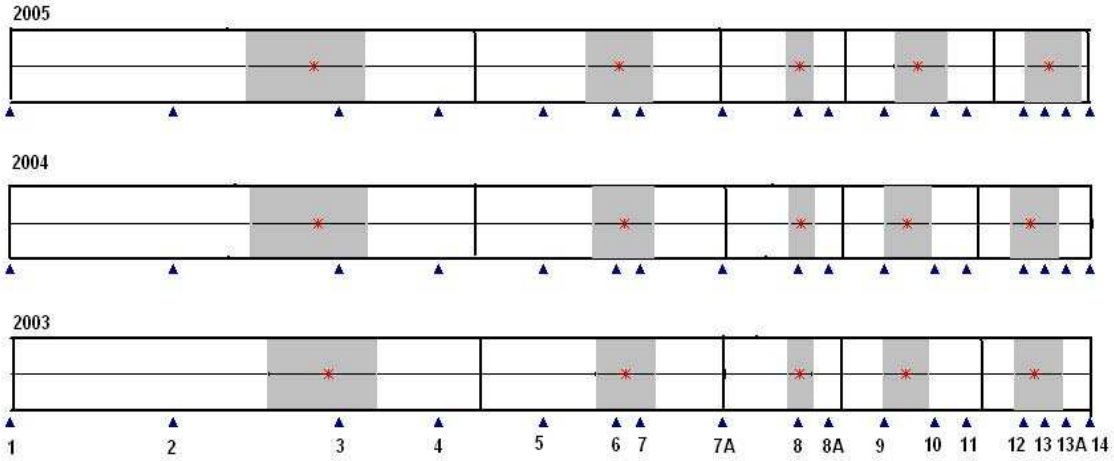
A flattening at each curve begins at 8–cluster solution (14%), and the curves become essentially flat after 18–cluster solution ($< 4\%$). It should be clear that the marginal gain would converge to zero for each case, as the number of clusters approach the number of data points.



(a) Data points with equal weights



(b) Data points with weight I



(c) Data points with weight II

Figure 9.2: Clustering results for $K = 5$ with different weights of data points

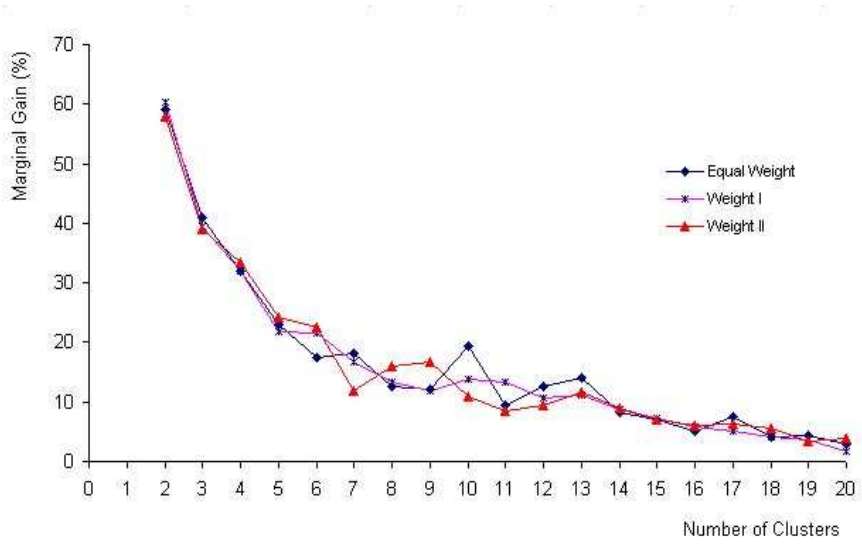


Figure 9.3: Marginal gain of clustering

Here the optimal number of cluster depends on the analyst. If we determine that 10% is the threshold, then for each of the three cases, 13–cluster solution becomes the optimal number, since threshold for the 14–cluster solutions for each case is lower than 10% (8.3%, 8.7%, 8.9% for equal weight, weight I and weight II, respectively).

Figure 9.4 shows the configuration of cluster center and boundaries for 13–cluster solution of equal weights, weight I and weight II. It can be seen that the discrepancy between different weights are clearer in this figure. In particular, within the northern sections of the mainline i.e. between interchanges 7A and 14, there appears higher number of clusters as compared to the southern sections. This can be related to the higher volumes and therefore more number of accidents occurring in the northern sections of the network.

An interesting observation in Figure 9.4 is the concentration of accidents around interchange 8 and 8A for each three cases. Although the average annual daily traffic of this section is not as high as the section at its north, there is more number of clusters within this region. Indeed, this portion has a 2,400 ft mainline merging section that drops from 5–lanes to 3–lanes. Not only does 11 this segment experience congestion due to merging, but it also has high number of accidents per length. In fact, recently, the NJTPK authority has proposed the widening of the Turnpike between interchange

6 and 8A.

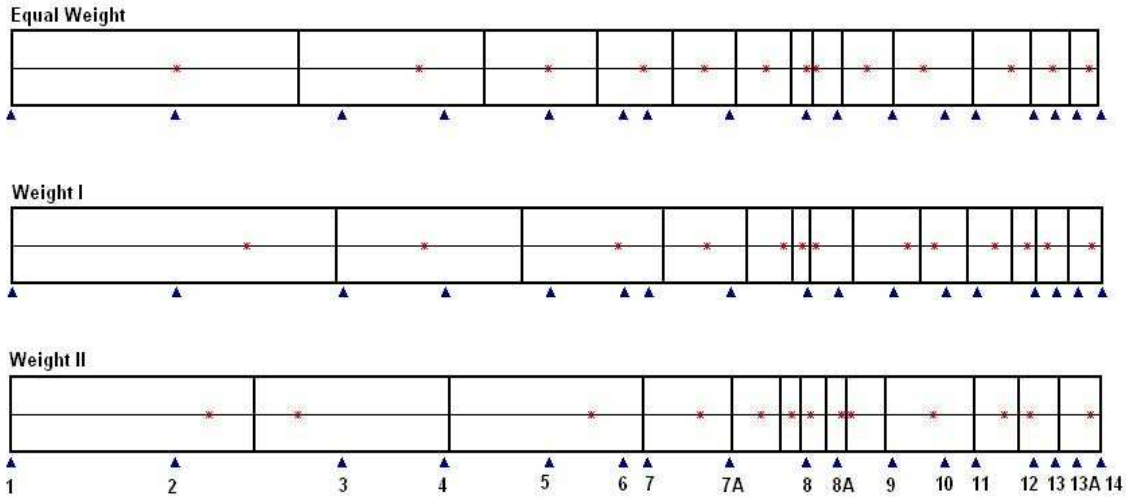


Figure 9.4: Optimal configuration of clusters for $K = 13$

Similarly, other locations with high density of clusters in the northern sections can be further investigated to understand the characteristics of the roadway and alleviate accidents by various means.

An interesting comparison is the comparison of homogeneous segments as determined by the clustering approach and the predefined segments on the mainline. By predefined segments, we mean links between each interchanges, such as 1–2, 2–3, 8–8A, etc. There are 16 segments on the selected mainline of the NJTPK. Figure 9.5 shows the configuration of homogeneous segments for 16 cluster solutions of each three cases. It can be seen that the segments configuration based on clustering approach is substantially different from the predefined segments.

9.4 Conclusion

In this chapter, probabilistic distance clustering algorithm is applied to determine the spatial clusters of accidents along a continuous highway using different objectives and is tested using historical accident data.

The most important advantage of the proposed clustering approach is the ease with

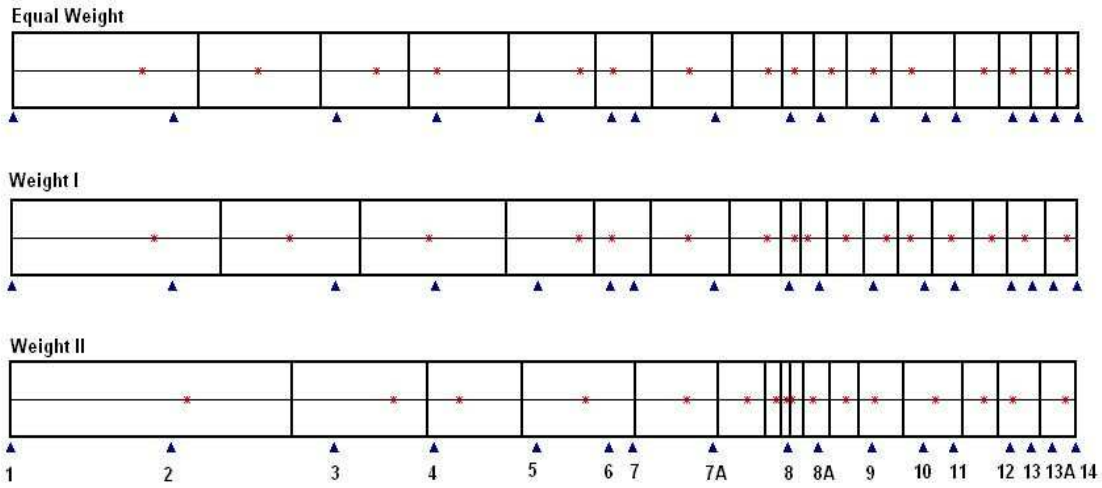


Figure 9.5: Clustering results for $K = 16$

which various characteristics of incidents can be incorporated. It has been shown that for different objectives of applications, different accident characteristics play important roles in determining the clusters. In particular, three different cases have been analyzed: (1) incidents of equal weight, (2) number of vehicles types involved, and (3) severity of incident. The analysis have shown that for each case, the cluster configurations change along the roadway; but for different years, the cluster configurations remain approximately the same. Also, within each clusters,"hot-spot", which correspond to concentration of accident that have high probability of belonging to each cluster, have been identified. Finally, a simple method for determining the optimal number of clusters have been presented using the marginal gain of the objective function.

Similar clustering analysis can be carried out for a more general and complex highway network. For policy makers, state agencies and planners, a map that shows "hot-spots" of accidents according to various factors of interest can be useful for determining which highways need to be treated to reduce accidents, or where to deploy emergency assistance centers for better incident response. Many State Departments of Transportation, including NJDOT have historical accident data. See for instance NJDOT online crash database [67]. These dataset are based on police accident reports and have detailed information as the database used in our analysis. Using this extensive set of

information, a network–wide map of "hot-spots" can be generated based on any desired objective using clustering algorithm.

Another advantage of the clustering approach is that it does not rely on predetermined highway segments, such as intersections, weaving areas, between interchanges, curvatures, etc. Analysis based on such segments already impose a space discretization error in the results. In the context of prioritization of highways based on safety, Miaou and Song [64] describe such methods as nave methods; and show that predetermined highway segments are bound to prioritization errors. Thus, the method described here can help reduce such errors. Future work will be towards the use of the proposed clustering algorithm to better understand the impact of reducing these prioritization errors in terms of the effectiveness of statewide safety decisions.

Chapter 10

Semi–Supervised Distance Clustering

10.1 Introduction

In this chapter, we introduce a new method for semi–supervised clustering, combining probabilistic distance clustering (see chapter 3) for the unlabelled data points and a least squares criterion for the labelled ones.

10.2 Semi–Supervised Clustering

Given a dataset S with N points, $\mathcal{S} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\} \subset \mathbb{R}^n$, we look at two extreme ways of clustering the data set.

In **supervised clustering**, a subset $\mathcal{T} \subset \mathcal{S}$ called the **training set** (or **labelled data**) is given, already partitioned into L disjoint clusters, say $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \cdots \cup \mathcal{T}_L$.

At a labelled point $\mathbf{x} \in \mathcal{T}$, the **prior information** (or **label**) is given as the cluster membership functions,

$$r_i(\mathbf{x}) := \begin{cases} 1, & \text{if } \mathbf{x} \in \mathcal{T}_i \ ; \\ 0, & \text{otherwise,} \end{cases} \quad (i = 1, \cdots, L) \ , \tag{10.1}$$

or as cluster membership probabilities, i.e.,

$$r_i(\mathbf{x}) := \text{Prob}\left\{\mathbf{x} \in \mathcal{T}_i\right\} \ , \ (i = 1, \cdots, L) \ , \tag{10.2}$$

where,

$$\sum_{i=1}^{L} r_i(\mathbf{x}) = 1 \ , \ r_i(\mathbf{x}) \geq 0 \ . \tag{10.3}$$

This information is used to design a clustering rule, which is then applied to the remaining data, $\mathcal{S} \setminus \mathcal{T}$, called the **testing set**.

In **unsupervised clustering** no prior information is given, and the data set is clustered to disjoint clusters, $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_K$, using intrinsic properties of the data. The number of clusters K, possibly different than L, is either given or is determined by the clustering algorithm.

In the unsupervised case we use probabilistic distance clustering, where the cluster membership probability as at point $\mathbf{x}$,

$$p_k(\mathbf{x}) := \text{Prob}\left\{\mathbf{x} \in \mathcal{S}_k\right\} \ , \ (k = 1, \cdots, K) \ , \tag{10.4}$$

depends on the distance of $\mathbf{x}$ from the center of the k^{th}–cluster, see chapter 3. These probabilities are determined by the clustering algorithm, and are in general different than the prior probabilities (10.2).

We propose a way to combine supervised and unsupervised clustering in a parametric model, using a parameter $0 \leq \theta \leq 1$ that measures the reliability of the prior information.

To simplify notation we consider here the case $K = L = 2$.

10.3 An Extremal Principle for Semi–Supervised Clustering

With the i^{th}–cluster we associate a center $\mathbf{c}_i$, and a distance function

$$d_i(\mathbf{x}) := \|\mathbf{x} - \mathbf{c}_i\| \ , \tag{10.5}$$

using an elliptical norm $\|\cdot\|$,

$$\|\mathbf{u}\| := \langle \mathbf{u}, Q\mathbf{u} \rangle \ , \tag{10.6}$$

where Q is a positive definite matrix, in particular, the Euclidean norm for $Q = I$, and the Mahalanobis norm for $Q = \Sigma_i^{-1}$, where Σ_i is the covariance of the cluster.

Let $0 \leq \theta \leq 1$ be a parameter measuring the importance of prior information, with $\theta = 1$ or $\theta = 0$ corresponding to supervised or unsupervised clustering, respectively.

For any point $\mathbf{x}$ in the training set $\mathcal{T}$ consider the problem,

$$\begin{aligned} \min_{p_1,p_2} \quad & (1-\theta)\left(d_1\, p_1^2 + d_2\, p_2^2\right) + \theta\left((p_1 - r_1)^2 d_1 + (p_2 - r_2)^2 d_2\right) \\ \text{s.t.} \quad & p_1 + p_2 = 1 \\ & p_1,\, p_2 \geq 0 \end{aligned} \tag{10.7}$$

where $p_i = p_i(\mathbf{x})$ and $d_i = d_i(\mathbf{x})$. The numbers $r_i = r_i(\mathbf{x})$ are the given labels. The role of the second half of (10.7) is to reconcile the labels (prior probabilities) r_i and the computed probabilities p_i.

The Lagrangian of this problem is

$$\begin{aligned} L(p_1, p_2, \lambda) = {} & (1-\theta)\left(d_1\, p_1^2 + d_2\, p_2^2\right) + \theta\left((p_1 - r_1)^2 d_1 + (p_2 - r_2)^2 d_2\right) \\ & + \lambda\,(1 - p_1 - p_2) \end{aligned} \tag{10.8}$$

and zeroing the gradient (with respect to p_1, p_2) we get

$$\begin{aligned} 2\,(1-\theta)\, p_1 d_1 + 2\,\theta(p_1 - r_1) &= \lambda \ , \\ 2\,(1-\theta)\, p_2 d_2 + 2\,\theta(p_2 - r_2) &= \lambda \ , \end{aligned}$$

and the probabilities,

$$p_1 = \frac{\lambda + 2\,\theta\, r_1 d_1}{2 d_1} \ , \ p_2 = \frac{\lambda + 2\,\theta\, r_2 d_2}{2 d_2} \ .$$

The probabilities add to 1,

$$\frac{\lambda + 2\,\theta\, r_1 d_1}{2d_1} + \frac{\lambda + \theta\, r_2 d_2}{2d_2} = 1\ ,$$

and therefore,

$$\lambda = 2\,(1-\theta)\frac{d_1 d_2}{d_1 + d_2}\ . \tag{10.9}$$

Substituting (10.9) in the probabilities,

$$p_1 = (1-\theta)\frac{d_2}{d_1 + d_2} + \theta\, r_1\ , \tag{10.10}$$

$$p_2 = (1-\theta)\frac{d_1}{d_1 + d_2} + \theta\, r_2\ . \tag{10.11}$$

10.4 Cluster Centers

For simplicity we identify the data set $\mathcal{D}$ with the training set $\mathcal{T}$, i.e. we assume labels for the whole data set. Then the extremal problem is

$$\min_{p_1, p_2} \quad (1-\theta)\sum_{\mathbf{x}\in\mathcal{T}} \left(d_1\, p_1^2 + d_2\, p_2^2\right) + \theta \sum_{\mathbf{x}\in\mathcal{T}} \left((p_1 - r_1)^2 d_1 + (p_2 - r_2)^2 d_2\right) \tag{10.12}$$

The gradient of the objective function in (10.12) w.r.t. $\mathbf{c}_1$ is

$$-\nabla_{\mathbf{c}_1} = (1-\theta)\sum_{\mathbf{x}} p_1^2\, \frac{\mathbf{x} - \mathbf{c}_1}{d_1} + \theta \sum_{\mathbf{x}} (p_1 - r_1)^2\, \frac{\mathbf{x} - \mathbf{c}_1}{d_1}$$

Zeroing the gradient, we get

$$\sum_{\mathbf{x}} \left[(1-\theta)\frac{p_1^2}{d_1} + \theta\frac{(p_1 - r_1)^2}{d_1}\right] \mathbf{x} = \mathbf{c}_1 \sum_{\mathbf{x}} \left[(1-\theta)\frac{p_1^2}{d_1} + \theta\frac{(p_1 - r_1)^2}{d_1}\right]\ .$$

An analogous expression can be written for the gradient with respect to $\mathbf{c}_2$, and therefore

$$\mathbf{c}_k = \sum_{i=1}^{N} \left(u_k(\mathbf{x}_i) \Big/ \sum_{j=1}^{N} u_k(\mathbf{x}_j) \right) \mathbf{x}_i \ , \tag{10.13}$$

where

$$u_k(\mathbf{x}_i) = (1-\theta)\frac{p_k^2}{d_k} + \theta\frac{(p_k - r_k)^2}{d_k} \ , \ k = 1, 2 \ . \tag{10.14}$$

The coefficients $u_k(\mathbf{x}_i)$ in (10.14) depend on the parameter θ. We study the behavior of the coefficient $u_1(\mathbf{x}_i)$ in the extreme cases $\theta = 0$ and 1. For this we calculate first, using (10.10),

$$\begin{aligned} p_1^2 &= (1-\theta)^2 \left(\frac{d_2}{d_1+d_2}\right)^2 + 2\,\theta(1-\theta)\frac{d_2}{d_1+d_2}\, r_1 + \theta^2\, r_1^2 \\ (p_1 - r_1)^2 &= (1-\theta)^2 \left[\left(\frac{d_2}{d_1+d_2}\right) - r_1\right]^2 \end{aligned}$$

Therefore,

$$\begin{aligned} d_1\, u_1(\mathbf{x}_i) = {} & (1-\theta)^3 \left(\frac{d_2}{d_1+d_2}\right)^2 + 2\,\theta(1-\theta)^2\frac{d_2}{d_1+d_2}\, r_1 \\ & + \theta^2(1-\theta) r_1^2 + \theta\,(1-\theta)^2 \left(\frac{d_2}{d_1+d_2}\right)^2 - 2\,\theta(1-\theta)^2 r_1 \frac{d_2}{d_1+d_2} \\ & + \theta(1-\theta)^2\, r_1^2 \ . \end{aligned} \tag{10.15}$$

The value for $\theta = 0$, and the limit as $\theta \to 1$, are respectively,

$$u_1(\mathbf{x}_i) = \begin{cases} \left(\dfrac{d_2}{d_1+d_2}\right)^2 / d_1 & , \ \theta = 0 \ , \\ \left(\dfrac{r_1^2}{d_1}\right) & , \ \theta \to 1 \ . \end{cases} \tag{10.16}$$

Analogous results apply to the coefficient $u_2(\mathbf{x}_i)$.

10.5 Semi–supervised Distance Clustering Algorithm

The above ideas are implemented in Algorithm 10.1 for semi–supervised distance clustering of data. A schematic description, presented – for simplicity – for the case of 2 clusters, follows.

Algorithm 10.1. Semi-supervised Distance Clustering

Initialization:	given data $\mathcal{T}$, any two points $\mathbf{c}_1, \mathbf{c}_2$, a value θ, and $\epsilon > 0$
Iteration:	
Step 1	**compute** distances $d_1(\mathbf{x})$, $d_2(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{T}$
Step 2	**compute** probabilities $p_1(\mathbf{x})$, $p_2(\mathbf{x})$, using (10.10)–(10.11) for all $\mathbf{x} \in \mathcal{T}$
Step 3	**update** the centers $\mathbf{c}_1^+, \mathbf{c}_2^+$, using (10.13)–(10.14)
Step 4	**if** $\|\mathbf{c}_1^+ - \mathbf{c}_1\| + \|\mathbf{c}_2^+ - \mathbf{c}_2\| < \epsilon$ **stop**
	return to step 1

The following two examples, illustrate Algorithm 10.1 in simulated datasets.

Example 10.1. A data set in $\mathbb{R}^2$ with $N = 200$ data points in each of two clusters is shown in Figure 10.1, where the labels are represented by different colors. The labels are clearly in conflict with the intrinsic clusters.

Figure 10.2 shows the clusters obtained for diffenet values of θ. In particular, for $\theta = 0.1$ or 0.3 (Figures 10.2(a)–10.2(b)) the prior labels are ignored. As θ increases, the prior information becomes more important, and for $\theta = 0.85$ (Figure 10.2(f)) the clusters agree with the given labels.

Example 10.2. In this example, a data set in the shape of **Yin–Yang** symbol with $N = 500$ data points is simulated (shown in Figure 10.3). Clustering results using different θ values are presented in Figure 10.4

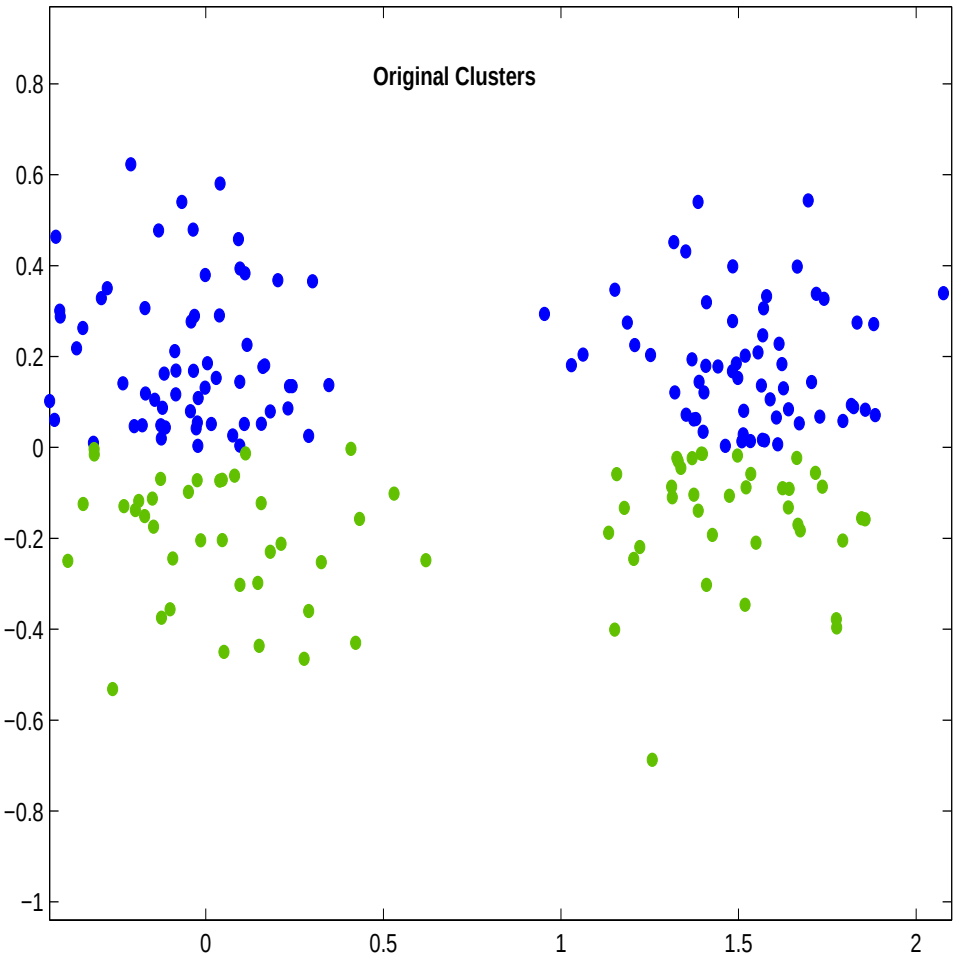
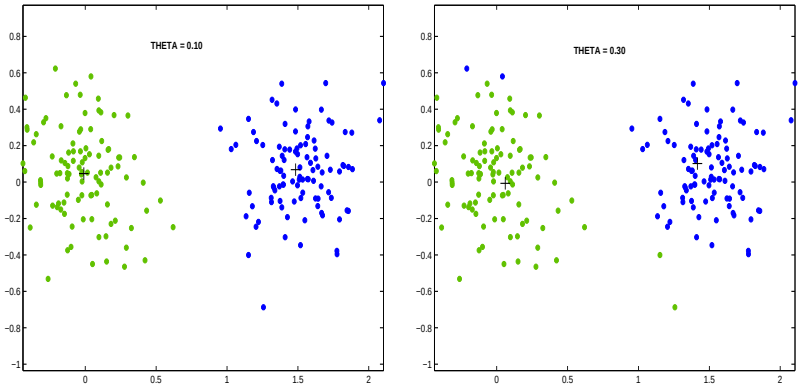
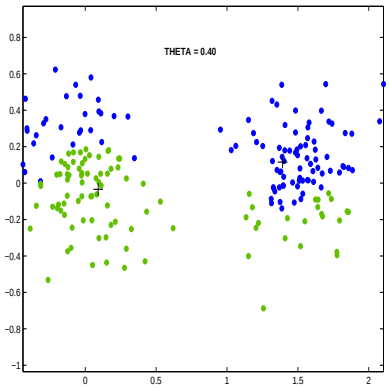


Figure 10.1: Original clusters in Example 10.1

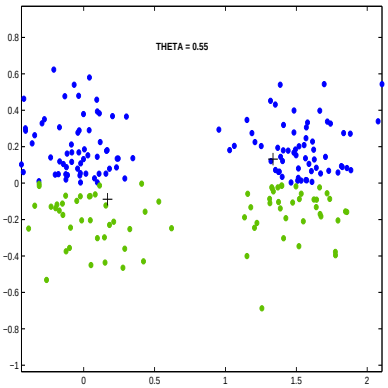


(a) Clusters with $\theta = 0.10$

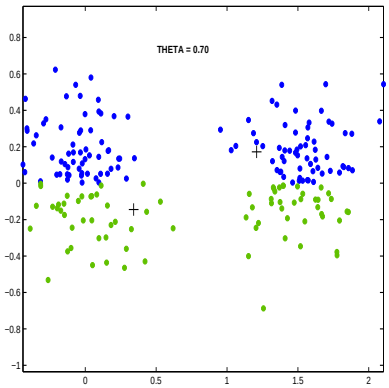
(b) Clusters with $\theta = 0.30$



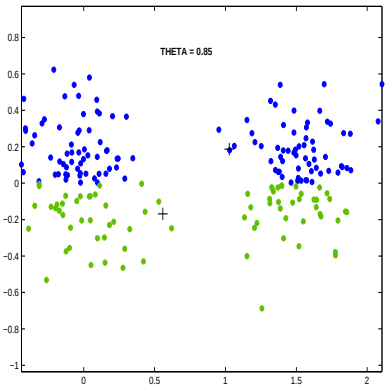
(c) Clusters with $\theta = 0.40$



(d) Clusters with $\theta = 0.50$



(e) Clusters with $\theta = 0.70$



(f) Clusters with $\theta = 0.85$

Figure 10.2: Clusters in Example 10.1 for different θ values

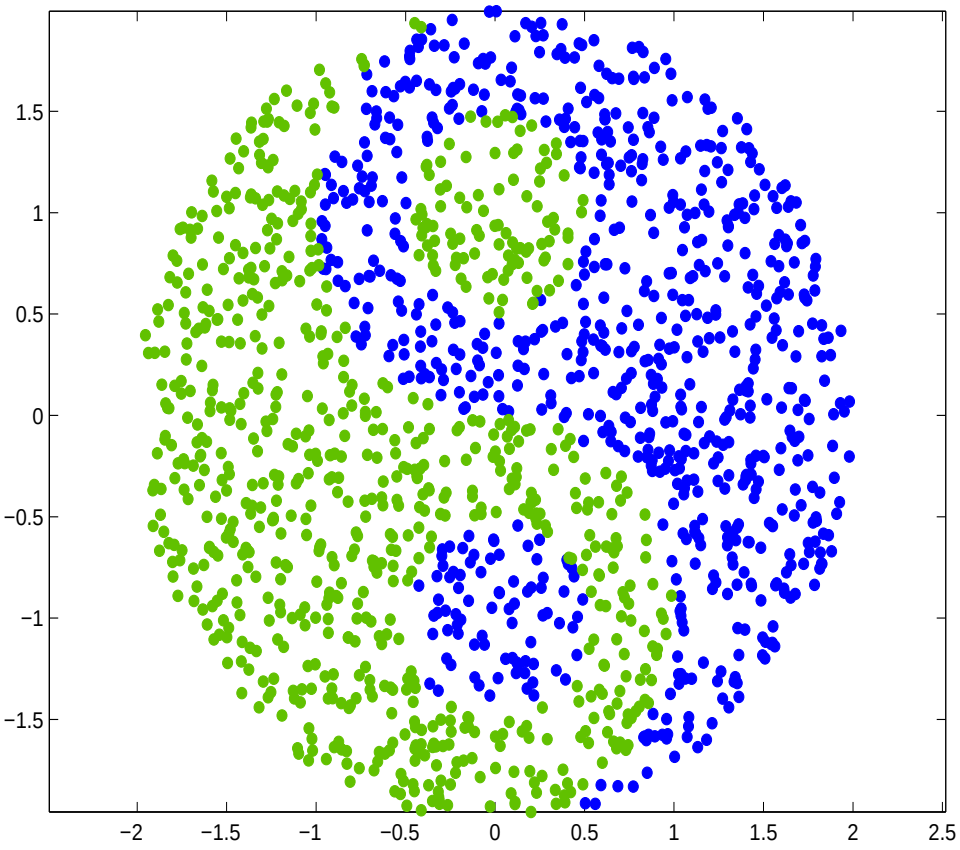
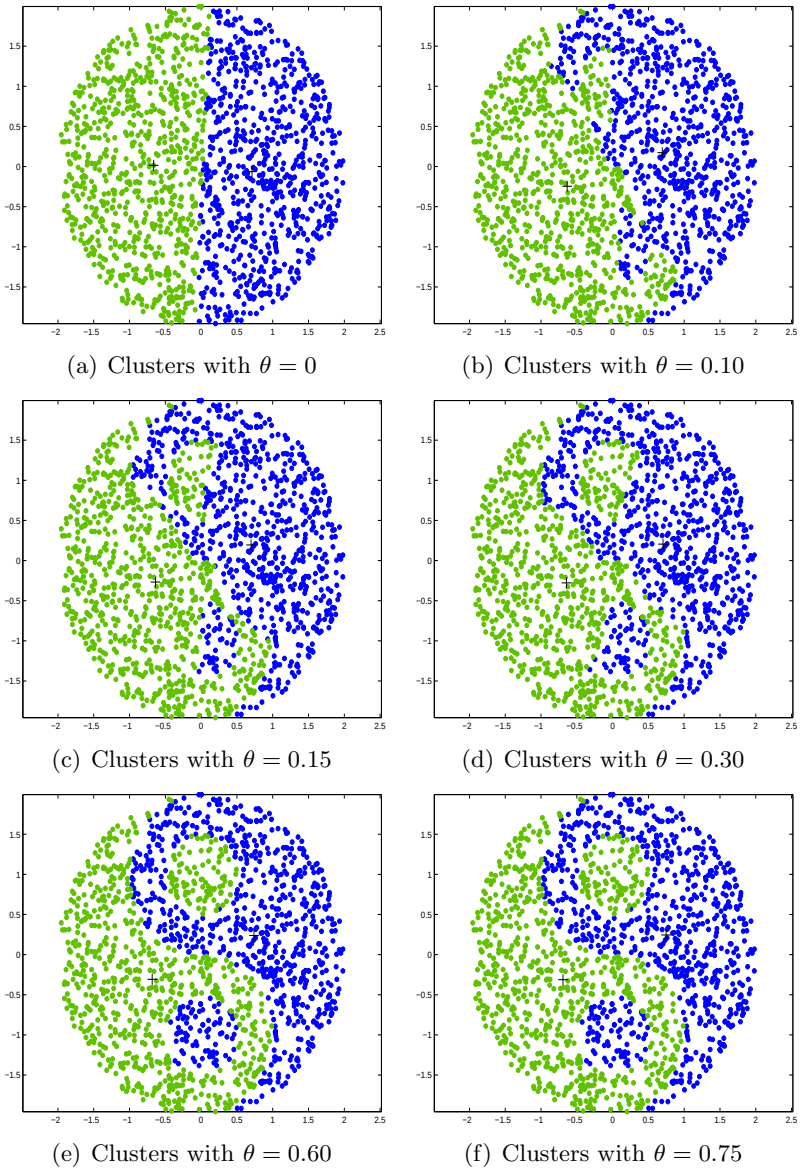


Figure 10.3: Original clusters in Example 10.2



(a) Clusters with $\theta = 0$

(b) Clusters with $\theta = 0.10$

(c) Clusters with $\theta = 0.15$

(d) Clusters with $\theta = 0.30$

(e) Clusters with $\theta = 0.60$

(f) Clusters with $\theta = 0.75$

Figure 10.4: Clusters in Example 10.2 for different θ values

References

[1] Cluster analysis. `http://en.wikipedia.org/wiki/Cluster_analysis`, 2007.

[2] Support vector machines. `http://en.wikipedia.org/wiki/Support_vector_machines`, 2007.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM-SIGMOD Conference On the Management of Data*, pages 94–105, 1998.

[4] M. R. Anderberg. *Cluster Analysis for Cluster Applications.* Acedemic Press, Inc., New York, NY, 1973.

[5] M. Arav. Contour approximation of data and the harmonic mean. *Mathematical Inequalities and Applications.* (to appear).

[6] B. Bartin B. and K. Ozbay. Evaluation of travel time variability in new jersey turnpike–a case study. *IEEE Transactions on Intelligent Transportation Systems*, 2007.

[7] G. H. Ball and D. J. Hall. Isodata, a novel method of data analysis and classification. Technical report, Standford University, Stanford, CA, 1965.

[8] G. H. Ball and D. J. Hall. A clustering tehnique for summarizing multivariate data. *Behavioural Science*, 12:153–155, 1967.

[9] J. D. Banfield and A. E. Raftery. Model–based gaussian and non–gaussian clustering. *Biometrics*, 49:803–821, 1993.

[10] A. Ben-Israel and C. Iyigun. Probabilistic distance clustering. *Journal of Classification*, 2007. (to appear).

[11] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, Inc., 2003.

[12] D. Bertsimas, A. J. Mersearau, and N. R. Patel. Dynamic classification of online customers. In *Proceedings of SIAM International Conference on Data Mining*, 2003.

[13] J. C. Bezdeck, R. Ehrlich, and W. Full. Fcm:fuzzy cmeans algorithm. *Computers and Geoscience*, 1984.

[14] J. C. Bezdek. *Fuzzy Mathematics in Pattern Classification.* PhD thesis, Applied Mathematics, Cornell University, Ithaca, New York, 1973.

[15] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York, Plenum, 1981.

[16] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Clustering via concave minimization. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, number 9, pages 368–374. MIT Press, Cambridge, 1997.

[17] C-Y. Chen, S-C. Hwang, and Y-J. Oyang. An incremental hierarchical data clustering algorithm based on gravity theory. In *Advances in Knowledge Discovery and Data Mining:6th Pacific-Asia Conference, PAKDD 2002*. Springer-Verlag LNCS 2336, 2002.

[18] Y-M. Cheung. k*-means: A new generalized k-means clustering algorithm. *Pattern Recognition Letters*, 24:2883–2893, 2003.

[19] H. Cohen and F. Southworth. On the measurement and valuation of travel time variability due to incidents on freeways. *Journal of Transportation and Statistics*, 2(2):123–131, 1999.

[20] W. D. Cook, A. Kazakov, and B. N. Persaud. Prioritising highway accident sites: A data envelopment analysis model. *Journal of the Operational Research Society*, 52:303–309, 2001.

[21] L. Cooper. Heuristic methods for location–allocation problems. *SIAM Review*, 6:37–53, 1964.

[22] R. N. Dave. Validating fuzzy partitions obtained through c-shells clustering. *Pattern Recognition Letters*, 17:613–623, 1996.

[23] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), 1979.

[24] E. Diday. The dynamic clustering method in non-hierarcial clustering. *Journal of Computational Information Sciences*, 2:61–88, 1973.

[25] K. R. Dixon and J. A. Chapman. Harmonic mean measure of animal activity areas. *Ecology*, 61:1040–1044, 1980.

[26] Z. Drezner, K. Klamroth, A. Schöbel, and G. O. Wesolowsky. The weber problem. In Z. Drezner and H. W. Hamacher, editors, *Facility Location: Applications and Theory*. Springer, New York, 2001.

[27] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

[28] J. C. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95–104, 1974.

[29] M. Ester, H-P.Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial data sets with noise. In *Proceedings 2nd Int. Conf. on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[30] B. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. Oxford University Press Inc., 2001.

[31] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth P., and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. MIT Press, Cambridge, Mass., 1996.

[32] D. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[33] E. W. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768769, 1965.

[34] I. Gath and A. B. Geva. Unsupervised optimal fuzzy clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11:773–781, 1989.

[35] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani. Mining the stock market: Which measure is best? In *Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 487–496, 2000.

[36] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large data sets. In *Proceedings of the ACM SIGMOD Conference*, 1998.

[37] M. Halkidi, M. Vazirgiannis, and I. Batistakis. Quality scheme assessment in the clustering process. In *Proceedings of PKDD, Lyon, France*, 2000.

[38] P. Hansen and B. Jaumard. Cluster analysis and mathematical programming. *Mathematical Programming*, 79:191–215, 1997.

[39] J. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., New York, N.Y., 1975.

[40] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, 2003.

[41] W. J. Heiser. Geometric representation of association between categories. *Psychometrika*, 69:513–545, 2004.

[42] A. Hinneburg and D. Keim. An efficient approach to clustering in large multimedia data sets with noise. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[43] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis*. John Wiley & Sons, Inc., New York, N.Y., 1999.

[44] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2:283–304, 1998.

[45] L. Hubert and D. Steinley. Agreement among supreme court justices: Categorical vs. continuous representation. *SIAM News*, 38(7), 2005.

[46] C. Iyigun and A. Ben-Israel. A distance clustering method for multifacility location problems. (to appear).

[47] C. Iyigun and A. Ben-Israel. A new criterion for clustering validity via contour approximation of data. (to appear).

[48] C. Iyigun and A. Ben-Israel. Probabilistic semi–supervised clustering. (to appear).

[49] C. Iyigun and A. Ben-Israel. Probabilistic clustering adjusted for cluster size. *Probability in the Engineering and Informational Sciences*, 2007. (to appear).

[50] C. Iyigun and A. Ben-Israel. Probabilistic distance clustering: Theory and applications. In W. Chaovalitwongse and P. M. Pardalos, editors, *Clustering Challenges in Biological Networks*. World Scientific, 2007.

[51] A. K. Jain. Cluster analysis. In Y. T. Young and K-S. Fu, editors, *Handbook of Pattern Recognition and Image Processing*. Academic Press, 1986.

[52] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[53] N. Jardine and R. Sibson. *Mathematical Taxonomy*. John Wiley & Sons, New York, 1971.

[54] L. M. Ostresh Jr. On the convergence of a class of iterative methods for solving the weber location problem. *Operations Research*, 26:597–609, 1978.

[55] G. Karypis, E. H. Han, and V. Kumar. Chameleon: A hierarchical clustering algorithm using dynamic modeling. *Computer*, 32(7):68–75, 1999.

[56] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, NY, 1990.

[57] S. B. Kotsiantis and P. E. Pintelas. Recent advances in clustering: A brief survey. Technical report, Department of Mathematics, University of Patras, 2003.

[58] R. Krishnapuram, H. Frigui, and O. Nasraoui. Quadratic shell clustering algorithms and the detection of second–degree curves. *Pattern Recognition Letters*, 14(7), 1993.

[59] H. W. Kuhn. A note on fermat's problem. *Mathematical Programming*, 4:98–107, 1973.

[60] R. Love, J. Morris, and G. Wesolowsky. *Facilities Location: Models and Methods*. North-Holland, 1988.

[61] J. Mao and A. K. Jain. A self–organizing network for hyperellipsoidal clustering. *IEEE Transactions Neural Networks*, 7:16–29, 1996.

[62] G. J. McLachlan and T. Krishnan. *The EM algorithm and Extensions*. John Wiley & Sons, New York, NY, 1997.

[63] J. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[64] S-P. Miaou and J. J. Song. Bayesian ranking of sites for engineering safety improvements: Decision parameter, treatability concept, statistical criterion, and spatial dependence. *Accident Analysis and Prevention*, 37:699–720, 2005.

[65] G. W. Milligan and M. C. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50:159–179, 1985.

[66] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of 14th Advances in Neural Information Processing Systems*, 2002.

[67] NJDOT. New jersey department of transportation (njdot) web site, accident database. Technical report, PAMI Research Group, University of Waterloo, 2005.

[68] K. Ozbay and B. Bartin B. Incident management simulation. *SCS Simulation Journal*, 79(2):69–82, 2003.

[69] K. Ozbay and P. Kachroo. *Incident Management for Intelligent Transportation Systems (ITS)*. Artech House, Massachusetts., 1999.

[70] N. R. Pal and J. Biswas. Cluster validation using graph theoretic concepts. *Pattern Recognition*, 30(6), 1997.

[71] R. Rezaee, B. P. F. Lelieveldt, and J. H. C. Reiber. A new cluster validity index for the fuzzy c-mean. *Pattern Recognition Letters*, 19:237–246, 1998.

[72] M. Sato, Y. Sato, and L. Jain. Fuzzy clustering models and applications. *Studies in Fuzziness and Soft Computing*, 9, 1997.

[73] B. Sch"olkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, Mass., 2002.

[74] S. C. Sharma. *Applied Multivariate Techniques*. John Willey & Sons, 1996.

[75] H. D. Sherali, J. Desai, H. Rakha, and I. El-Shawarby. A discrete optimization approach for locating automatic vehicle identification readers for the provision of roadway travel times. In *82nd Annual Meeting of the Transportation Research Board*, 2002.

[76] R. Shioda and L. Tuncel. Clustering via minumum volume ellipsoids. *Computational Optimization and Applications*, 2006. (to appear).

[77] P. Sun and R.M. Freund. Computation of minimum volume covering ellipsoids. *Operations Research*, 52(5):690–706, 2004.

[78] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 2006.

[79] M. Teboulle. A unified continuous optimization framework for center-based clustering methods. *Journal of Machine Learning*, 8:65–102, 2007.

[80] S. Theodoridis and K. Koutroubas. *Pattern Recognition*. Academic Press, 1999.

[81] P. Tsui. Em–gm algorithm matlab code. Technical report, PAMI Research Group, University of Waterloo, 2006.

[82] E. Weiszfeld. Sur le point par lequel la somme des distances de n points donnés est minimum. *Tohoku Math. J.*, 43:355–386, 1937.

[83] X. L. Xie and G. Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4), 1991.

[84] B. Yang and E. Miller-Hooks. Determining critical arcs for collecting real–time travel information. In *81st Annual Meeting of the Transportation Research Board*, 2001.

[85] C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, C-20(1):6886, 1971.

[86] T. Zhang, R. Ramakrishnan, and M. Linvy. Birch: An efficient data clustering method for very large data sets. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.

Vita

Cem Iyigun

Education

2007 **PhD in Operations Research**
Rutgers, The State University of New Jersey, Piscataway, NJ

2002 **MS in Operations Research**
Rutgers, The State University of New Jersey, Piscataway, NJ

1999 **BS in Industrial Engineering**
Middle East Technical University, Ankara, Turkey

Academic & Teaching Activities

2003–2007 **Instructor, Rutgers Business School, Piscataway, NJ**

2001–03/ Summer **Lecturer, Mathematics Department**
Rutgers University, Piscataway, NJ

2000–01 **Research Assistant, Center for Advanced Infrastructure and Transportation**
Rutgers University, Piscataway, NJ

2000–03 **Teaching Assistant, Center for Operations Research**
Rutgers University, Piscataway, NJ

Awards & Honors

2003–2007 **DIMACS Summer/Winter Graduate Student Research Award**
Center for Discrete Mathematics and Theoretical Computer Science, NJ

2000–2003 **Research / Teaching Assistantship**
Rutgers University, Piscataway, NJ

1999–2000 **Walter C. Russell Graduate Fellowship**
Rutgers University, Piscataway, NJ

1999 **Graduate NATO Science Fellowship**
The Scientific and Educational Research Council of Turkey (TUBITAK)

Publications

Journal Papers:

• A. Ben-Israel and **C. Iyigun**, Probabilistic Distance Clustering, *Journal of Classification*, 2007 (in print).

• **C. Iyigun** and A. Ben-Israel, Probabilistic Distance Clustering with Cluster Size, *Probability in Engineering and Informational Sciences*, 2007 (in print).

• K. Ozbay, B. Bartin, **C. Iyigun**, A Clustering Based Methodology for Determining the Optimal Roadway Configuration of Detectors for Travel Time Estimation, *Journal of the Transportation Research Board*, 2007 (in print).

Refereed Book Chapters:

• **C. Iyigun** and A. Ben-Israel, Probabilistic Distance Clustering, Theory and Applications, in P.M. Pardalos, W.A. Chaovalitwongse (Eds.), *Clustering Challenges in Biological Networks*, World Scientific, 2007 (in print).

• Y–J Fan, **C. Iyigun** and W. A. Chaovalitwongse, Recent Advances in Optimization Models for Data Mining: Clustering, Feature Selection, and Classification, in P. Hansen, P.M. Pardolos (Eds.), *CRM Proceedings & Lecture Notes*, American Mathematical Society, 2008 (in print).