ORDER SELECTION IN UNSUPERVISED LEARNING AND CLUSTERING FOR ARBITRARY AND NON-ARBITRARY SHAPED DATA

by

Mahdi Shahbaba

M.Sc. Boras University, Sweden, 2010

A dissertation

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

in the Program of

Electrical and Computer Engineering

Toronto, Ontario, Canada, 2015

© Mahdi Shahbaba 2015

Author's Declaration

I hereby declare that I am the sole author of this dissertation. This is a true copy of the dissertation, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my dissertation may be made electronically available to the public.

Abstract

Order Selection in Unsupervised Learning and Clustering for Arbitrary and Non-arbitrary Shaped Data

Mahdi Shahbaba

Doctor of Philosophy, Electrical and Computer Engineering

Ryerson University, 2015

This thesis focuses on clustering for the purpose of unsupervised learning. One topic of our interest is on estimating the correct number of clusters (CNC). In conventional clustering approaches, such as X-means, G-means, PG-means and Dip-means, estimating the CNC is a preprocessing step prior to finding the centers and clusters. In another word, the first step estimates the CNC and the second step finds the clusters. Each step having different objective function to minimize. Here, we propose minimum averaged central error (MACE)-means clustering and use one objective function to simultaneously estimate the CNC and provide the cluster centers. We have shown superiority of MACE-means over the conventional methods in term of estimating the CNC with comparable complexity. In addition, on average MACE-means results in better values for adjusted rand index (ARI) and variation of information (VI). Next topic of our interest is order selection step of the conventional methods which is usually a statistical testing method such as Kolmogrov-Smrinov test, Anderson-Darling test, and Hartigan's Dip test. We propose a new statistical test denoted by Sigtest (signature testing). The conventional statistical testing approaches rely on a particular assumption on the probability distribution of each cluster. Sigtest on the other hand can be used with any prior distribution assumption on the clusters. By replacing the statistical testing of the mentioned conventional approaches with Sigtest, we have shown that the clustering methods are improved in terms of having more accurate CNC as well as ARI and VI. Conventional clustering approaches fail in arbitrary shaped clustering. Our last contribution of the thesis is in arbitrary shaped clustering. The proposed method denoted by minimum Pathways in

Arbitrary Shaped (minPAS) clustering is proposed based on a unique minimum spanning tree structure of the data. Our simulation results show advantage of minPAS over the state-of-the-art arbitrary shaped clustering methods such as DBSCAN and Affinity Propagation in terms of accuracy, ARI and VI indexes.

Acknowledgments

First and foremost, I would like to express my special gratitude to my supervisor Professor Soosan Beheshti for sharing her knowledge and wisdom with me, encouraging my research, and supporting me during these past four years. I was tremendously fortunate to have her as my supervisor.

My sincere gratitude goes to my committee members, Professors Ebrahim Bagheri, Matthew Kyan, Alireza Sadeghian, Amirnaser Yazdani, and Shahryar Rahnamayan for their brilliant comments and suggestions. I also want to thank Professor Jean Mason for her availability and assistance during my defense.

I would also like to thank my parents, my wife, my brother, my sister, my brothers-in-law, and my sisters-in-law for their continued support and love. This accomplishment would not have been possible without their help. Special thanks to my colleagues and friends in Signal and Information Processing (SIP) lab for their insightful discussions and collaborations.

I would like to acknowledge the Natural Sciences and Engineering Research Council for providing funding for this work.

To my wife Maryam, for all of her love and support

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Clustering has wide range of applications in different disciplines of science and engineering such as bioinformatics, genetics, image segmentation [1], voice recognition, document classification and weather classification [2–4]. Even astronomers categorize galaxies and stars using cluster analysis techniques [5]. Clustering is used in monitoring spread of disease and detecting significant spatial disease clusters [6]. In life sciences, clustering is an inevitable step in most of the methods for analyzing protein interactions and grouping gene expressions [7, 8]. Customizing and categorizing Internet search results is another application of clustering on text, image and audio data, which can be used for recommending books, movies and music to users [9]. For example, a user who has searched for a specific book might be also interested in other books written by the same author. Those books are considered as a single cluster based on the selected author. Therefore, any member of the cluster can be suggested to the user. In market research, clustering is mainly used for segmenting or grouping customers and products. The result of clustering also reveals the size and capacity of market, dependencies among different segments, and it gives a better understanding about the acceptance of a product by its users [10–12]. Finding different communities in social networks is another application of clustering in the field of computer science [13].

Clustering is mainly an unsupervised learning problem, where unlike supervised learning training sets or class labels of data samples are not available. In other words, the only available information is the unlabeled data itself. The goal of a clustering algorithm is to subjectively group observed data samples based on their similarity and dissimilarity [14]. *A priori* definition of similarity plays an important role in data clustering. Assumed shape or distribution of a cluster such as Gaussian or non-Gaussian [15], the membership definition for samples of clusters, and clustering optimization criteria are fundamental elements for defining similarity in clustering [16], [17].

Challenges in Clustering: In majority of clustering methods a predefined number of clusters is required before clustering process. In real life, however, the correct number of clusters (CNC) is not known *a priori*. Consequently, one main challenge for these clustering methods is determining the CNC. Another challenge in clustering is having a predefined definition for shape of clusters. Therefore, grouping arbitrary shaped datasets that do not follow any specific known distribution seems to be a difficult task.

In the following, a brief history of existing clustering methods is provided to further elaborate on these challenges.

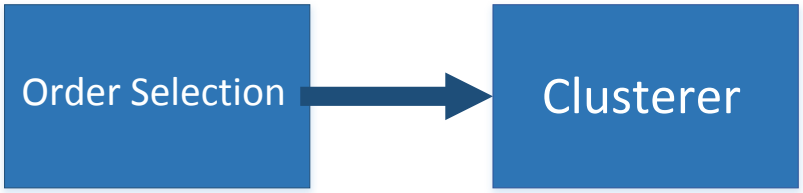


Figure 1.1: Clustering with order selection.

Non-arbitrary shaped clustering methods: K-means clustering is one of the earliest partitional clustering methods which needs to have the number of clusters (K) before clustering [18, 19]. It starts with randomly selecting K cluster centers from the samples and assigns other samples as the cluster members. K-means iteratively updates

the centers and cluster members until the algorithm converges to a stable solution. K-means algorithm is considered as a fast clustering method for partitioning datasets with spherical Gaussian clusters. However, it is very sensitive to the initial choice of the centers and can get trapped in the local optima of its objective function.

Model-based methods also have attracted a lot of attention over the non-probabilistic and heuristic clustering techniques [20]. The reason of this superiority is in providing a generative and predictive model, and measuring the uncertainty of the samples assigned to each cluster [21]. Gaussian Mixture Model (GMM), a model based method, is among the most commonly utilized models for clustering. In a GMM framework, each cluster is assumed to have a Gaussian distribution and each sample can have a shared membership with couple of clusters in the mixture model. To estimate the model parameters of a Gaussian mixture, Expectation Maximization (EM) algorithm is considered as an efficient solver which iteratively converges to a solution. EM algorithm is considered as a general form of K-means clustering with this difference that clusters can follow any distribution in Gaussian family. In addition, EM relates samples to their centers using soft assignment, while K-means is limited to hard assignments between centers and their members. Similar to K-means, EM relies on a provided CNC.

To estimate the CNC in K-means clustering (order selection in Figure 1.1) a wide range of approaches are suggested in the literature [22]. Pioneer methods of CNC calculation are proposed with K-means clustersr and are validity indexes. According to these approaches, for each *a priori* assumption for the CNC, an index value is calculated. The method chooses the CNC based on optimizing the index value. Example of these approaches are Xie-Beni index [23], Dunn index [24], Silhouette index [25], Davies-Bouldin (DB) index, Calinski-Harabasz (CH) index [26], Krzanowski-Lai index [27], and weighted inter-to intra-cluster ratio (wtertra) [28]. Most recent validity index methods are available in [29], [30]. For example, Figure 1.1 shows two general steps in clustering methods with order selection.

In general, model based clustering approaches are categorized as hierarchical or partitional clustering processes.

Hierarchical Clustering: Another approach for clustering data and estimating the CNC is using hierarchical clustering joined with a proper order selection criterion or statistical test in Figure 1.1. In this scenario, clustering algorithms such as K-means and EM can split the data hierarchically and use an order selection criterion to decide about terminating the splitting procedure. X-means [31], G-means [32] and Dip-means [33] are some of the widely used and state of the art clustering methods based on splitting criteria.

Order selection of these methods (in Figure 1.1) can be categorizes as follows: 1) Bayesian Information Criterion (BIC) in X-means; X-means is an extension to K-means which employs BIC to estimate the CNC, but it tends to over-fit the data. 2) Anderson-Darling (AD) in G-means; G-means is another wrapper around K-means, which estimates the CNC based on the AD statistical test. It examines the Gaussianity of the estimated clusters and performs better than X-means, but has difficulties for overlapped clusters. 3) Dip test in Dip-means; Dip-means is a newly proposed hierarchical clustering method that extends the choice of cluster distribution from Gaussian to a wider range of unimodal distributions, where Gaussian, log-Normal and student's t-distributions are three examples of this large family. This method employs Hartigan's Dip statistical test to evaluate whether each cluster has a unimodal distribution or not.

Partitional Clustering: Similar to hierarchical clustering methods, statistical tests can be used for estimating CNC in partitional clustering approaches. PG-means clustering is an example of a partitional clustering method with order selection. In this method, data and its assumed Gaussian mixture model are projected over the direction of maximum variance of data iteratively. Consequently, the projected versions of the model and data are compared using Kolmogorov-Smirnov (KS) test to decide about the CNC. In term of having Gaussian assumption PG-means is similar to G-means, but it

has this advantage that can find better models using EM algorithm [34]. In general, clustering methods with statistical tests have a prior assumption on the distribution of the observed data. The observed data provides an empirical distribution function (ecdf) and this ecdf is compared with the desired cdf (Dcdf) in the process of estimating the CNC.

In addition to the mentioned hierarchical and partitional clustering approaches, the CNC challenge in clustering is studied from different perspectives. For example, Gap statistic takes the output of any clustering algorithm, and then compares the change within cluster dispersion to that expected under an appropriate reference null distribution [35]. The main disadvantage of this method is its computationally expensive behavior which makes it inefficient for high dimensional data. Another example is System Evolution which clusters data using K-medois [36, 37] and defines a validity index from the viewpoint of a pseudo thermodynamics system [38] for estimating the CNC. This method is efficient for well-separated clusters but has difficulties on overlapped clusters.

Arbitrary shaped clustering methods: Unlike the above discussed approaches, a large family of clustering methods don't have any assumption on the shape and distribution of clusters. In these methods the center of cluster is a loose concept which is not defined in the related algorithms. Most of these clustering methods group data samples heuristically and don't rely on statistical tests or order selection approaches for estimating the CNC.

Spectral Clustering is one of the well-known clustering methods that can partition arbitrary shaped data [39]. This method is constructed based on the eigenvectors and eigenvalues of similarity matrix of data. The number of clusters is one of the requirements of this algorithm that should be available as a predefined value.

Normalized cuts (Ncut) for image segmentation is another clustering approach based on partitioning image graph [1]. This method measures both the total dissimilarity

among different clusters and the total similarity within clusters. Ncut optimizes its criterion based on a generalized eigenvalue approach to partition the data samples. Similar to spectral clustering, Ncut can work on arbitrary shaped data. The estimation of CNC is not efficiently addressed, and the number of clusters should be provided to the algorithm.

Data Spectroscopic (DaSpec) [40] is another extension of spectral clustering methods which is constructed based on the Gaussian kernel matrix of data. This method estimates the number of clusters by identifying the eigenvectors that have no sign changes up to a predefined threshold value. A predefined kernel bandwidth is also another requirement of this algorithm. DaSpec assumes that clusters are well-separated, therefore it cannot recognize clusters with small distances among them.

Voting-K-means is an example of clustering methods based on combination of different clustering results [41]. In this method, results of different K-means clusterings for an initial number of clusters leads to a co-association matrix which helps to extract the underlying consistent clusters.

Density based clustering approaches are another group of methods that don't have any assumption on the distribution of data. In these methods, high density regions are assumed to be clusters which are separated with low density regions or gaps. Therefore, these type of methods can cluster arbitrary shaped data better than model based algorithms. DBSCAN is an example of state of the art clustering methods for arbitrary shaped data [42].

Affinity Propagation [43] is an example of Graph theoretic clustering approaches which also doesn't have any assumption on the distribution of clusters and can independently estimate the CNC value. This method has a high time complexity and has difficulties in recognizing clusters with complex geometries.

Our Objectives: In this thesis, our main focus is on clustering methods that can estimate the CNC and find clusters in arbitrary and non-arbitrary shaped datasets.

We consider different statistical tests and order selection approaches for estimating the number of clusters and suggest our solutions for estimating CNC and clustering different shapes of data.

Thesis Outline: This thesis is organized as follows: In Chapter 2, we provide a brief background on some of the widely used clustering methods, solvers algorithms, dimension reduction and splitting criterion for estimating the number of clusters.

In Chapter 3, we estimate the CNC based on a rigorous modeling and analysis of Mean Square Error for different number of centers in clustering. We define the average central error (ACE) that is the difference between the ground truth center of the cluster and our estimation of the center (unavailable error). We present the estimate of this error based on the available compactness error. The proposed MACE-means clustering is constructed based on minimizing the ACE error (Minimum ACE-means). The art of this approach is in probabilistic validation of the unknown ACE by using available cluster compactness.

In Chapter 4, we propose a new statistical test denoted by signature test (Sigtest) for estimating the number of clusters. Unlike the existing statistical tests it can be used with *any* prior assumption on the distribution of the clusters (Dcdf). Since the Dcdf can be replaced with any other desired cdf, using the method with any other prior assumption is analogous to what is presented here. Details of using Sigtest in both hierarchical and partitional clustering is provided. Our simulation results show the superiority of using Sigtest as the statistical test in terms of estimating the number of clusters, adjusted rand index (ARI) and variation of information (VI). In addition, we propose using Sigtest in image classification using bag of visual words (BOVW) [44], [45], [46]. While majority of the BOVW based methods have the assumption of a fixed or predefined visual vocabulary size, we propose using Sigtest for estimating the optimum size of the vocabulary based on Scale Invariant Feature Transform (SIFT) features. We show that adaptive prior estimation of vocabulary size in BOVW has a significant effect on increasing the accuracy

of image classification along with decreasing the time complexity in some applications.

In Chapter 5, we propose minimum pathway arbitrary shaped clustering, denoted by minPAS, as a clustering approach that can work with both arbitrary and non-arbitrary shaped clusters. minPAS can independently estimate the number of clusters with a high level of accuracy compared with the similar methods. The proposed method benefits from the unique tree structure of data and can measures the level of similarity among samples using the minimum pathways. Unlike regular distance measures such as euclidean distance, minimum pathways in minPAS respect the geometry of data and does not impose any assumption on the distribution. Chapter 6 is our conclusion on the proposed methods.

Chapter 2

Background

In this Chapter, we briefly discuss some of the widely used clustering methods and their requirements.

In Section 2.1, we explain K-means and Mixture of Gaussians as two of the most important bases in hierarchical and partitional clustering methods. We discuss X-means, G-means, PG-means and Dip-means as some of the well known clustering examples based on K-means and Gaussian Mixture models.

Kolmogorov-Smirnov, Anderson-Darling and Hartigan's Dip test as statistical tests for splitting criterion in clustering methods are discussed in Section 2.2.

In Section 2.3 we explain Principal Component Analysis (PCA) that is one of the basic approaches for data dimension reduction. PCA can be used as a preprocessing step for estimating the number of clusters.

Spectral clustering, Normalized Cut, DBSCAN, Voting-K-menas, and Affinity Propagation as examples of clustering methods that can cluster arbitrary shaped data are discussed in Section 2.4.

2.1 Hierarchical and Partitional Clustering Methods

2.1.1 K-means

K-means is one of the most well known and widely used partitional clustering methods. The algorithm can be implemented easily and has a reasonable speed for converging to the clustering solution. However, it has the following limitations: K-means is very sensitive to the initialization error and can be trapped in local optima of its objective function. K-means also has its best performance for spherical Gaussian clusters and using that for arbitrary shaped data can lead to a poor clustering result.

Assume we have a data set $y = [x_1, ..., x_N]^T$ which is an $N \times n$ matrix consisting of N observations in an n-dimensional space and each x_i $(i = 1, \ldots, N)$ represents a data sample in the feature space. The goal is to partition the data samples into K clusters in a way that in each cluster internal distances among members of the cluster are smaller than distances to points outside of the cluster. Consider $\mu = [\mu_1, ..., \mu_K]^T$ as a $K \times n$ matrix where μ_i $(i \in \{1, \ldots, K\})$ is a row of the matrix and the center of the ith cluster. We assume that the number of clusters or K is given [47].

For the first step of clustering, μ should be initialized with K random centers in the n-dimensional space. Then each data point should be assigned to the nearest cluster C_l:

$$x_i \in C_l,\, if \parallel x_i - \mu_l \parallel < \parallel x_i - \mu_j \parallel \tag{2.1}$$

$$for\, i = 1, ..., N,\, j \neq l,\, and \;\; j, l \in \{1, ..., K\} \tag{2.2}$$

where μ_l and μ_j are the centers of clusters C_l and C_j, respectively. x_i is a member of the cluster C_l and $||\cdot||$ is l_2-norm.

In the next step, we should recalculate the center of each cluster:

$$\mu_l = \frac{1}{N_l} \sum_{x_i \in C_l} x_i, \tag{2.3}$$

where N_l is the total number of the samples assigned to the cluster C_l.

After this step all of the centers will be updated to the new values, then data points will be assigned to the nearest centers and this routine continues until either location of the centers remain unchanged or the algorithm reaches to the predefined maximum number of iterations [48].

2.1.2 Mixture of Gaussians

Suppose we are given a data set $y = [x_1, \ldots, x_N]$ where x_is are i.i.d. random vectors of length n generated from m unknown Gaussian distributions. In our assumptions, the latent variable $z_i = j$ from a multinomial distribution assigns the ith sample of our data set to the jth Gaussian distribution.

We wish to model our data set by joint distribution $p(x_i, z_i)$:

$$p(x_i, z_i) = p(x_i|z_i)\, p(z_i) \tag{2.4}$$

having the latent variable z_i, the distribution of the data sample x_i can be given as:

$$(x_i|z_i = j) \sim \mathcal{N}(\mu_j, \Sigma_j) \tag{2.5}$$

where μ_j and Σ_j are the mean and covariance matrices of the jth Gaussian distribution. We define ϕ_j as the probability of choosing the jth Gaussian. Therefore, μ, Σ and ϕ can be assumed the model parameters which are required to be estimated.

The log-likelihood of the data can be employed to estimate the parameters of this model:

$$
\begin{aligned}
l(\phi, \mu, \Sigma) &= \sum_{i=1}^{n} \log p(x_i; \phi, \mu, \Sigma) \\
&= \sum_{i=1}^{n} \log \sum_{z_i=1}^{m} p(x_i|z_i; \mu, \Sigma)\, p(z_i; \phi). \qquad (2.6)
\end{aligned}
$$

Unfortunately, the maximum likelihood estimates of the parameters does not lead to a closed form solution. Thus, finding the derivatives of $l(\phi, \mu, \Sigma)$ with respect to the model parameters and setting them to zero cannot tackle this problem. Moreover, z_is are unknown which makes it more difficult to find the maximum likelihood estimation.

The Expectation Maximization (EM) algorithm is an iterative algorithm which has two main steps (E-step and M-step). This algorithm can be applied to our problem to estimate the model parameters. In the E-step, z_is will be estimated:

$$
w_i^j = p(z_i = j|x_i; \phi, \mu, \Sigma) \qquad (2.7)
$$

where w_i^j is the probability of the ith sample being generated by the jth Gaussian component, and $\sum_{j=1}^{m} w_i^j = 1$. Then in the M-step, model parameters will be estimated as follows:

$$\phi_j = \frac{1}{n}\sum_{i=1}^{n} w_i^j \tag{2.8}$$

$$\mu_j = \frac{\sum_{i=1}^{n} w_i^j x_i}{\sum_{i=1}^{n} w_i^j} \tag{2.9}$$

$$\Sigma_j = \frac{\sum_{i=1}^{n} w_i^j (x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=1}^{n} w_i^j} \tag{2.10}$$

In the following section, the EM algorithm is explained in details, but before that Jensen's inequality which is a prerequisite to the EM is explained.

Jensen's inequality: Let f be a convex function of random variable X, then:

$$E[f(X)] \leq f(E[X]) \tag{2.11}$$

In addition, for a strictly convex f, $E[f(X)] = f(E[X])$ holds true if and only if $X = E[X]$ with probability 1 which means X must be a constant. f is a convex function and X (horizontal axis) is a random variable with a 0.5 chance for choosing a and 0.5 chance of choosing b.

If f is strictly concave then $-f$ is strictly convex and direction of Jensen's inequalities will be reversed ($E[f(X)] \geq f(E[X])$).

EM algorithm: In some cases, deriving a closed form solution for maximum likelihood may not be possible. Expectation Maximization (EM) is an iterative algorithm which is promising to find the maximum likelihood parameters [49–51].

Let consider an estimation problem in which $x = [x_1, \ldots, x_N]$ is a set of N independent training data (observed data) and $z = [z_1, \ldots, z_N]$ is a set of latent random variable (unseen data). We aim to fit the parameters of a model $p(x, z)$ to the data. In this case

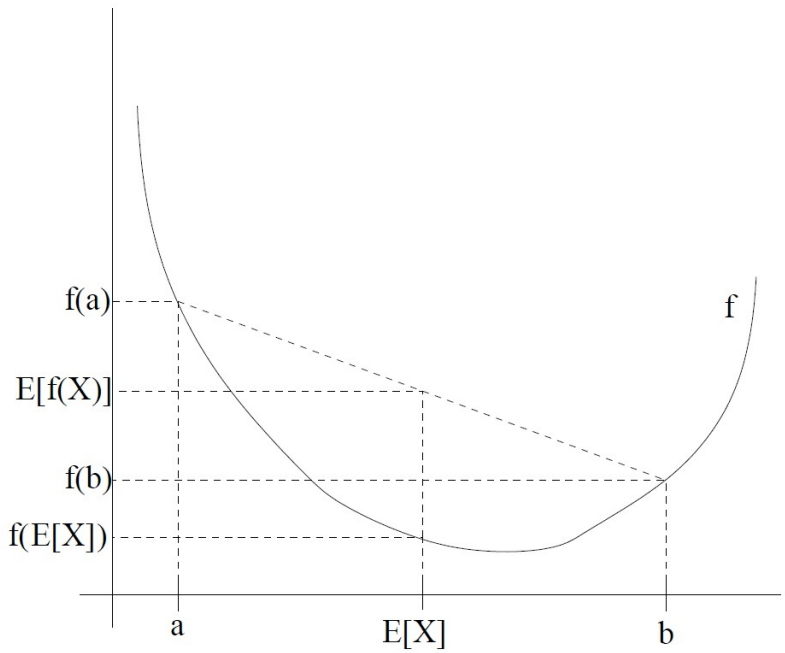


Figure 2.1: Visualization of Jensen's inequality.

likelihood of the data is given by:

$$
\begin{aligned}
l(\theta) &= \sum_{i=1}^{n} log\, p(x_i;\theta) \\
&= \sum_{i=1}^{n} log \sum_{z} p(x_i, z;\theta)
\end{aligned}
\tag{2.12}
$$

To maximize $l(\theta)$ using EM, we repeatedly construct a lower-bound on l (E-step) and then maximize that lower bound to estimate new optimal model parameters (θ).

Consider the following log-likelihood equation:

$$
\sum_{i=1} log\, p(x_i;\theta) = \sum_{i=1} log \sum_{z_i} p(x_i, z_i;\theta) \tag{2.13}
$$

Let assume $Q_i(z)$ as a probability distribution over z ($\sum_z Q_i(z) = 1, Q_i(z) \geq 0$). Then,

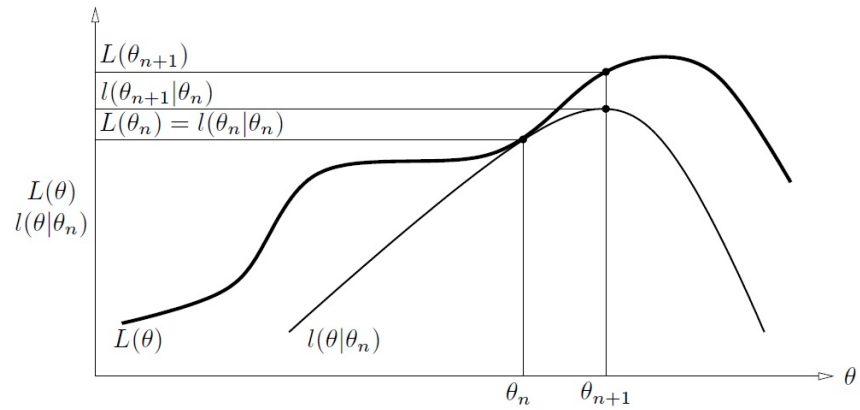


Figure 2.2: EM algorithm iterations for estimating maximum likelihood.

we can rewrite the previous equation as follows:

$$
\begin{aligned}
&= \sum_{i=1} log \sum_{z_i} Q_i(z_i) \, \frac{p(x_i, z_i; \theta)}{Q_i(z_i)} \\
&\geq \sum_{i=1} \sum_{z_i} Q_i(z_i) \, log \, \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}
\end{aligned}
\tag{2.14}
$$

Which the last step is derived based on Jensen's inequality and the term:

$$
\sum_{z_i} Q_i(z_i) \, \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}
\tag{2.15}
$$

The above equation (2.15) is expectation of $p(x_i, z_i; \theta)/Q_i(z_i)$ with respect to z_i drawn from distribution $Q_i(z_i)$. Therefore, for any set of Q_i a lower bound on $l(\theta)$ can be estimated. There are many possible choices for Q_i. If we have a current guess for θ then we will have the lower bound tight at the value of θ which means inequality will change into equality at that θ. By using Jensen's inequality, the bound will be tight at a particular θ. It means $E[f(X)] = f(E[X])$ holds true if and only if $X = E[X]$, and

this latter is only true for constant X. Therefore:

$$\frac{p(x_i, z_i; \theta)}{Q_i(z_i)} = c \tag{2.16}$$

where c is a constant and it is independent of z_i. Since we know $\sum_z Q_i(z) = 1$, then:

$$\begin{aligned} Q_i(z_i) &= \frac{p(x_i, z_i; \theta)}{\sum_z p(x_i, z_i; \theta)} \\ &= p(z_i | x_i; \theta) \end{aligned} \tag{2.17}$$

where Q_is are set to be posterior distribution of zs, given x_i and setting of the parameters θ. This was E-step of the algorithm which gives a choice of Q_i to find the lower bound on the log-likelihood $l(\theta)$ that we want to be maximized. In M-step we try to maximize this lower bound in (2.14) to estimate the next optimum θ:

$$\theta = max_\theta \sum_{i=1} \sum_{z_i} Q_i(z_i) \; log \; \frac{p(x_i, z_i; \theta)}{Q_i(z_i)} \tag{2.18}$$

The model parameters (μ, Σ, ϕ) of a Gaussian mixture can be estimated by calculating the derivatives of (2.18) with respect to the model parameters and setting them to zero. We can rewrite this equation as follows:

$$
\begin{aligned}
&\sum_{i=1}^{m}\sum_{z_i} Q_i(z_i)\ log\ \frac{p(x_i, z_i; \phi, \mu, \Sigma)}{Q_i(z_i)} \\
&= \sum_{i=1}^{m}\sum_{j=1}^{k} Q_i(z_i = j) \log \frac{p(x_i|z_i = j; \mu, \Sigma)p(z_i = j; \phi)}{Q_i(z_i = j)} \\
&= \sum_{i=1}^{m}\sum_{j=1}^{k} w_i^j \log \frac{\frac{1}{(2\pi)^{n/2}|\Sigma_j|^{1/2}} \exp(-\frac{1}{2}(x_i - \mu_j)^T \Sigma_j^{-1}(x_i - \mu_j))\phi}{w_i^j}
\end{aligned} \tag{2.19}
$$

Every E-step will be followed by a M-step and this procedure will be repeated iteratively. Since EM causes log-likelihood to converge monotonically, the stopping point for the algorithm will be reached if the increase in $l(\theta)$ between two successive iterations is smaller than some tolerance parameter.

2.1.3 X-means Clustering

One of the very first clustering methods which is able to estimate the number of clusters independently is X-means clustering [31], [52]. This method is a wrapper around K-means algorithm that hierarchically splits parent clusters into children clusters until the convergence condition is satisfied. X-means relies on Bayesian Information Criterion (BIC) to check whether a subset of data should be split or we should consider it as a single cluster. Consequently it can estimate the number of clusters from number of times that splitting happens. For example, X-means starts clustering with assumption of $K = 1$ as the number of clusters and then cluster the same data for $K = 2$. Consequently, clustering results of these two scenarios will be compared by BIC criterion to decide between $K = 1$ and $K = 2$.

In X-means clustering, there are following steps for grouping the data samples:

1. Initialize $K = 1$.

2. Run K-means algorithm on the data.

3. Calculate the BIC of each cluster with K=1 and K=2

4. If BIC for K=1 is less than BIC for K=2, check the test for other clusters.

5. If BIC for K=2 is less than BIC for K=1, increment K by one, go to the step 2.

6. Stop the algorithm if the convergence condition is satisfied, for example K is not changing.

The BIC can be calculated as follows:

$$BIC(\theta) = L(D) - \frac{1}{2}\, p \log\, N \tag{2.20}$$

where $L(D)$ is the log-likelihood of the data set D based on model θ which suggests a certain number of clusters; p is the number of free parameters in the model and N is the size of the data set. With assumption of having clusters generated from spherical Gaussian distributions, the log-likelihood $L(D)$ of the data set D will be defined as follows:

$$L(D) = \log \prod_j \prod_i pr(x_{ij})$$

$$= \sum_j \sum_i \log(\frac{n_j}{N} \frac{1}{\sqrt{2\pi}\hat{\sigma}^d} exp(-\frac{\parallel x_{ij} - \hat{C}_j \parallel^2}{2\hat{\sigma}^2})) \tag{2.21}$$

where x_{ij} is the ith member of the jth cluster with $pr(x_{ij})$ as its probability, n_j is the size of the jth cluster with $\hat{C}_j$ as its estimated center and $\hat{\sigma}^2$ is the estimated variance in the clusters. d is the dimension of the data.

Estimations of C_j and σ in (2.21) can be derived as follows:

$$\hat{C}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} x_{ij} \tag{2.22}$$

$$\hat{\sigma}^2 = \frac{1}{N-K} \sum_j \sum_i \parallel x_{ij} - \hat{C}_j \parallel^2 \tag{2.23}$$

where K is the number of clusters.

Based on this fact that log-likelihood for all the data points is sum of the log-likelihood of every single data point [53], log-likelihood $L(D_j)$ for data points belong to the jth cluster will be defined as follows:

$$L(D_j) = n_j \log n_j - n_j \log N - \frac{n_j}{2} \log(2\pi) \tag{2.24}$$

$$- \frac{n_j d}{2} \log \hat{\sigma}^2 - \frac{n_j - K}{2}$$

2.1.4 G-means Clustering

G-means is a hierarchical clustering algorithm which benefits from Anderson-Darling statistic test (AD) for evaluating the Gaussianity of clusters. Similar to X-means clustering, this algorithm is also a wrapper around the K-means and can only perform hard clustering. In contrast to X-means, G-means can deal with any distribution from the Gaussian family [32].

In G-means clustering, there are following steps for grouping the data samples:

1. Initialize $K = 1$.
2. Run K-means algorithm on the data.
3. Project the cluster members onto the direction of the maximum variance in the cluster, this direction can be found by PCA or similar approaches.
4. Use the Anderson Darling test on the projected data to test its Gaussianity.
5. If the data passed the test, check the test for other clusters.
6. If the data didn't pass the test, increment K by one, go to the step 2.

7. Stop the algorithm if the convergence condition is satisfied, for example K is not changing.

2.1.5 PG-means Clustering

PG-means clustering can learn the number of clusters in a Gaussian Mixture Model (GMM) scheme using Expectation Maximization algorithm (EM) [34]. This method is a partitional clustering approach which simultaneously projects data and its model onto the several random directions in space. PG-means uses Kolmogrove-Smirnov test (KS) to detect any mismatch between the data and the model. Unlike G-means that works with K-means, using EM algorithm gives a better ability of recognizing the overlapped clusters to PG-means.

In PG-means clustering, there are following steps for grouping the data samples:

1. Initialize $K = 1$.
2. Run EM algorithm on the data.
3. Project all of the data and the assumed model of the data onto several random direction in space.
4. Use the Kolmogrove-Smirnov test (KS) on the projected data and model to find any mismatch between them.
5. If the data and model are matched, stop the algorithm and give the final K.
6. If the data and model are not matched, increment K by one, go to step 2.

2.1.6 Dip-means Clustering

Dip-means clustering is constructed based on the Hartigan's dip test of unimodality [33]. According to this clustering method, each sample is a viewer with different distance values from other samples. Using Dip test, distribution of the distance values should be

examined for unimodality. If all viewers pass the unimodality test then null hypothesis of having a single cluster will be approved. Otherwise, a model with more than one cluster should be considered for the samples. This method is a wrapper around K-means.

In Dip-means clustering, there are following steps for grouping the data samples:

1. Initialize $K = 1$.
2. Run K-means algorithm on the data.
3. Calculate the distance matrix for each cluster.
4. Use the Hartigan's Dip test for the distances between each sample and the other cluster members.
5. If the data passed the test, check the test for other clusters.
6. If the data didn't pass the test, increment K by one, go to step 2.
7. Stop the algorithm if the convergence condition is satisfied, for example K is not changing.

2.2 Statistical Tests in Clustering

In statistical testing for clustering, empirical distribution function of data (ecdf) is compared to a desired cdf (Dcdf). The main goal of following statistical testing is to verify whether ecdf can be considered as a sample of Dcdf or not. In general there are two possible hypothesis:

- H_0: The observed data (ecdf) is a sample of the desired model (Dcdf)
- H_1: The observed data (ecdf) is not a sample of the desired model (Dcdf)

The first step of these methods usually requires transforming the observed data into a 1-dimensional data. This preprocessing step is required by majority of the statistical

tests such as AD, KS and Dip test. For example, one well-known approach is using principal component analysis (PCA) to calculate the direction of maximum variance in data (the main principal component), and projecting the data samples onto that direction. G-means and PG-means clustering are examples of clustering methods based on data projection. Dip-means clustering on the other hand works with another transformation of data by calculating the distances between data samples and a proper reference point.

Lets $x = [x_i, \cdots, x_N]^T$ be the observed data and $x_i \in R^d$, where d is the dimension of data samples. The transformed version of data is $y = [y_1, \cdots, y_N]^T$, where $y_i \in R^1$.

In the following subsections, we briefly discuss three statistical tests KS, AD and Dip which employ different approaches to measure the similarity between the distribution of transformed data y and the Dcdf.

2.2.1 Kolmogorov-Smirnov Test

This test compares the maximum point-wise distance between the ecdf with the Dcdf of the reference distribution. For example, in the case of PG-means clustering the Dcdf has Gaussian distribution. The distance is defined as [54]:

$$KS_{score} = \sup_{\substack{i \\ 1 \leq i \leq N}} |F_N(y_i) - F(y_i)| \tag{2.25}$$

where $F(y_i)$ is the Dcdf and $F_N(y_i)$ is the ecdf of observed data:

$$F_N(y_i) = \frac{1}{N} \sum_{j=1}^{N} I(y_j, y_i) \tag{2.26}$$

where

$$I(y_j, y_i) = \begin{cases} 1 & y_j < y_i \\ 0 & y_j \geq y_i \end{cases} \tag{2.27}$$

The observed samples are compared with a critical value T, to either be accepted

as a sample of a Gaussian distribution (H_0) or to be not considered as a sample of a Gaussian distribution (H_1).

- H_0: The observed data (ecdf) is a sample of a Gaussian distribution (Dcdf) $\leftrightarrow$ $KS_{score} \leq T$
- H_1: The observed data (ecdf) is not a sample of a Gaussian distribution (Dcdf) $\leftrightarrow$ $KS_{score} > T$

The critical value T is chosen adaptively by Lilliefor's test statistic which is the result of Monte Carlo calculations in [55].

2.2.2 Anderson Darling Test

Anderson-Darling test is similar to Kolmogorov-Smirnov test in comparing the ecdf and Dcdf, but it calculates a weighted difference between $F_n(y)$ and $F(y)$ over all of the N samples [56], [57]:

$$AD_{score} = N \int_{-\infty}^{\infty} A(y)(F_N(y) - F(y))^2 dF(y) \tag{2.28}$$

note that compare to KS, AD emphasizes more on the tails of the distribution:

$$A(y) = \frac{1}{F(y)(1 - F(y))} \tag{2.29}$$

The observed samples are compared with a critical value T, to either be accepted as a sample of a Gaussian distribution (H_0) or it is not considered as a sample of a Gaussian distribution (H_1).

- H_0: The observed data (ecdf) is a sample of a Gaussian distribution $\leftrightarrow$ $AD_{score} \leq T$
- H_1: The observed data (ecdf) is not a sample of a Gaussian distribution $\leftrightarrow$ $AD_{score} > T$

The critical value T is suggested to be $T = 1.8692$ for a confidence level of 0.0001 in [32].

2.2.3 Haritagn's Dip test

A more recently proposed method for the purpose of statistical testing in clustering is Hartigan's Dip test. This method generalizes the Gaussian assumption of the two above methods to a unimodal distribution. Unimodal distribution includes distribution such as Gaussian, Log-Normal, Student's t-distribution.

The probability density function (pdf), denoted by f, of unimodal distributions is monotonically non-decreasing in $(-\infty, y_L)$ and monotonically non-increasing in (y_U, ∞), where (y_L, y_U) for $y_L \leq y_U$ is the mode region of distribution. We let $\rho(F_N, G) = \sup_{\substack{i \\ 1 \leq i \leq N}} |F_N(y_i) - G(y_i)|$, therefore, Dip value of ecdf F_N, denoted by $D(F_N)$ is defined as [58]:

$$D(F_N) = \min_{G \in \mathcal{U}} \rho(F_n, G) \tag{2.30}$$

where G is a member of unimodal family $\mathcal{U}$ that represents the closest approximation for F_N[1].

It is shown that uniform distribution has the smallest Dip value among all of the unimodal distributions, to decide about the unimodality of F_N, its Dip value will be compared with the Dip values of uniform distributions $U[0, 1]$. [33] suggested that if for 1000 bootstraps of $U[0, 1]$, the probability of having $D(F_n)$ smaller than Dip values of the uniform distributions (we denote it by Dip_{score}) is larger than a critical value T, then data is unimodal (H_0), otherwise it is multimodal (H_1).

- H_0: The observed data (ecdf) is a sample of a unimodal distribution $\leftrightarrow Dip_{score} > T$
- H_1: The observed data (ecdf) is not a sample of a multimodal distribution $\leftrightarrow$

[1] For some $y_L \leq y_U$, G is the greatest convex minorant (g.c.m.) of F_n over $(-\infty, y_L)$, and for (y_U, ∞) it is the least concave minorant (l.c.m.) of F_n. G has the constant maximum slope in (y_L, y_U).

$Dip_{score} \leq T$

The critical value T is suggested to be zero ($T = 0$) for a significance level of 0, where in all of the cases Dip_{score} of a data with unimodal distribution should be larger than any other data with uniform distribution [33].

2.3 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is an algorithm for transforming data samples from a space with high dimensionality to another space with less dimensionality.

PCA is an orthogonal transformation and it can only deal with linear data. In general, we have the following assumptions for PCA:

1. Linearity of data (some extensions of PCA consider non-linear data).
2. Large variances have important structures.
3. The principal components are orthogonal.

Let $y = [x_1, \ldots, x_N]$ be an $n \times N$ matrix where each column x_i is a variable regarded as a vector belonging to N observations. Assuming these observations are obtained on a large number of variables (N is a large number), there may exist a redundancy in those variables. In other words, some of the variables are correlated with each other, possibly because they are observing the same source and not different independent constructs. The goal of PCA is to decrease this redundancy and represent the data with m ($m < N$) synthetic variables or Principal Components (PCs). These PCs are axes of new space which represent the data with less dimensionality.

If we assume each x_i is a linear combination of m independent sources $S = [s_1, \ldots, s_m]$ where s_i has the same length as x_i, then the observed data y can be given as:

$$y = SA \tag{2.31}$$

where A is an $m \times N$ mixing matrix with constant elements.

Now the question is, what is the possible transformation on the observed data which derives PCs and leads to the less dimensionality. The transformation should preserve the independent constructs of the data and PCs should be the directions in the data space which have high variances. The PC with the highest variance among other components has the maximum information about the data. Therefore, PCA is more about finding the components with high variances [59].

The covariance matrix of the data is necessary for deriving PCs. If we assume that the covariance matrix Λ_{yy} of the normalized data y is not available but the observations have stationary behavior, the estimation of covariance matrix can be given as follows:

$$\begin{aligned} \hat{\Lambda}_{yy} &= E[y^T y] \\ &= \begin{pmatrix} \hat{\sigma}_{11} & \cdots & \hat{\sigma}_{1n} \\ \vdots & \ddots & \vdots \\ \hat{\sigma}_{n1} & \cdots & \hat{\sigma}_{nn} \end{pmatrix} \end{aligned} \tag{2.32}$$

$$\hat{\sigma}_{jk} = \frac{1}{N-1} \sum_{i=1}^{N} (x_{(i,j)} - \mu_j)(x_{(i,k)} - \mu_k) \tag{2.33}$$

where $\hat{\sigma}_{jk}$ is located at the jth row and the kth column of the covariance matrix and is a measure of relation between x_j and x_k which μ_j and μ_k are means of the variables. $x_{(i,j)}$ and $x_{(i,k)}$ are the ith observation of x_j and x_k variables.

To decrease the variable redundancy and find the PCs, the first step is to find a linear function of the variables x_i $(i = 1, \ldots, N)$ which gives the maximum variance:

$$y\alpha_1 = \alpha_{11}x_1 + \alpha_{21}x_2 + \ldots + \alpha_{N1}x_N \tag{2.34}$$

where $\alpha_1 = [\alpha_{11}, \ldots, \alpha_{N1}]^T$ is a vector of N constant values. We look for another linear

function of the data samples ($y\alpha_2$) which has the second largest variance and is uncorrelated with $y\alpha_1$. The procedure of finding the linear functions can be followed until having the kth ($k \leq N$) linear function of the variables which has the kth largest variance and at the same time is uncorrelated to previously derived functions. Therefore, it is possible to calculate up to maximum n uncorrelated PCs, but we hope to represent the data with less number of PCs which have most of the variations of the data set.

Then the kth largest PC is given by $z_k = y\alpha_k$ where α_k is an eigenvector of $\hat{\Lambda}_{yy}$ corresponding to the kth largest eigenvalue λ_k. For unit length α_k ($\alpha_k^T \alpha_k = 1$) the $var(z_k) = \lambda_k$ where $var(z_k)$ is the variance of z_k. The first PC ($y\alpha_1$) is related to α_1 which maximizes $var(y\alpha_1) = \alpha_1^T \hat{\Lambda}_{yy} \alpha_1$. The constraint $\|\alpha_k\| = 1$ is used in derivation which simplifies the optimization problem and it means sum of squares of elements of α_1 is equal to one.

To maximize $\alpha_1^T \hat{\Lambda}_{yy} \alpha_1$ with respect to $\|\alpha_k\| = 1$ the Lagrangian equation $\mathcal{L}(\alpha_1)$ can be written as follows:

$$\mathcal{L}(\alpha_1) = \alpha_1^T \hat{\Lambda}_{yy} \alpha_1 - \lambda(\alpha_1^T \alpha_1 - 1) \tag{2.35}$$

where λ is a Lagrange multiplier. Differentiation subject to α_1 gives:

$$\hat{\alpha}_1 = \max_{\alpha 1} \mathcal{L}(\alpha_1) \tag{2.36}$$

$$\hat{\Lambda}_{yy} \hat{\alpha}_1 - \lambda \hat{\alpha}_1 = 0 \tag{2.37}$$

which gives:

$$(\hat{\Lambda}_{yy} - \lambda I)\hat{\alpha}_1 = 0 \tag{2.38}$$

where I is the identity matrix. Thus, λ is an eigenvalue of $\hat{\Lambda}_{yy}$ and it is corresponding

to eigenvector $\hat{\alpha}_1$. To decide which of the eigenvectors maximizes the variance of PC, we consider that following term should be maximized:

$$\begin{aligned} \hat{\alpha}_1^T \hat{\Lambda}_{yy} \hat{\alpha}_1 &= \hat{\alpha}_1^T \lambda \hat{\alpha}_1 \\ &= \lambda \hat{\alpha}_1^T \hat{\alpha}_1 \\ &= \lambda \end{aligned} \tag{2.39}$$

Therefore, λ corresponds to eigenvector $\hat{\alpha}_1$ and it should be the largest eigenvalue of $\hat{\Lambda}_{yy}$. The second PC ($y\alpha_2$) also should maximizes $\alpha_2^T \hat{\Lambda}_{yy} \alpha_2$ subject to being uncorrelated with previous PC ($y\hat{\alpha}_1$) which means $E[(y\hat{\alpha}_1)^T (y\alpha_2)] = 0$.

$$\begin{aligned} E[(y\hat{\alpha}_1)^T (y\alpha_2)] &= \hat{\alpha}_1^T \hat{\Lambda}_{yy} \alpha_2 \\ &= \alpha_2^T \hat{\Lambda}_{yy} \hat{\alpha}_1 \\ &= \alpha_2^T \lambda_1 \hat{\alpha}_1 \\ &= \lambda_1 \alpha_2^T \hat{\alpha}_1 \\ &= \lambda_1 \hat{\alpha}_1^T \alpha_2 \end{aligned} \tag{2.40}$$

Then any of the following conditions will be useful to define the PCs uncorrelated:

$$\hat{\alpha}_1^T \hat{\Lambda}_{yy} \alpha_2 = 0, \qquad \alpha_2^T \hat{\Lambda}_{yy} \hat{\alpha}_1 = 0 \tag{2.41}$$

$$\hat{\alpha}_1^T \alpha_2 = 0, \qquad \alpha_2^T \hat{\alpha}_1 = 0 \tag{2.42}$$

If we choose the later constrain and having the normalization constrain then the Lagrangian function will be:

$$\mathcal{L}(\alpha_2) = \alpha_2^T \hat{\Lambda}_{yy} \alpha_2 - \lambda(\alpha_2^T \alpha_2 - 1) - \phi\, \alpha_2^T \hat{\alpha}_1 \tag{2.43}$$

where λ and ϕ are Lagrange multipliers. Maximization with respect to α_2 gives:

$$\hat{\alpha}_2 = \max_{\alpha_2} \mathcal{L}(\alpha_2) \tag{2.44}$$

$$\hat{\Lambda}_{yy}\hat{\alpha}_2 - \lambda\hat{\alpha}_2 - \phi\hat{\alpha}_1 = 0 \tag{2.45}$$

Multiplication by $\hat{\alpha}_1^T$ gives:

$$\hat{\alpha}_1^T\hat{\Lambda}_{yy}\hat{\alpha}_2 - \lambda\hat{\alpha}_1^T\hat{\alpha}_2 - \hat{\alpha}_1^T\phi\hat{\alpha}_1 = 0 \tag{2.46}$$

Since first two terms are zero and $\hat{\alpha}_1^T\hat{\alpha}_1 = 1$ then $\phi = 0$.

$$\hat{\Lambda}_{yy}\hat{\alpha}_2 - \lambda\hat{\alpha}_2 = 0 \tag{2.47}$$

and

$$(\hat{\Lambda}_{yy} - \lambda I)\hat{\alpha}_2 = 0 \tag{2.48}$$

So λ is an eigenvalue of $\hat{\Lambda}_{yy}$, and $\hat{\alpha}_2$ the corresponding eigenvector. Similar to the first PC, $\lambda = \hat{\alpha}_2^T\hat{\Lambda}_{yy}\hat{\alpha}_2$ should be as large as possible. This procedure can be repeated to find all of the PCs as the λ_k is the kth largest eigenvalue of $\hat{\Lambda}_{yy}$ and α_k is the corresponding eigenvector:

$$var(y\alpha_k) = \lambda_k \tag{2.49}$$

We can summarize the calculation of PCs as follows:

1. Normalize data.
2. If covariance matrix of the population is not available calculate the covariance matrix of the samples.

3. Calculate eigenvectors and eigenvalues.

4. Keep m top eigenvectors which are related to the m largest eigenvalues.

5. Multiply eigenvectors by the data to calculate the PCs.

An extension of PCA for calculating principal curves is introduced in [60].

2.4 Arbitrary Shaped Clustering Methods

2.4.1 Spectral Clustering

Spectral clustering can partition arbitrary shaped data if the number of clusters K is available [61, 62]. In general spectral clustering methods, K largest eigenvectors of the Laplacian of the affinity matrix will be used for partitioning data. Following steps show the process of spectral clustering for an observed dataset $x = [x_1, x_2, \cdots, x_N]^T$ [39]:

1. Calculate the affinity matrix $A \in R^{N \times N}$, where A_{ij} is the distance between samples x_i and x_j:

$$A_{ij} = e^{\frac{-\|x_i - x_j\|^2}{2\sigma^2}} \tag{2.50}$$

   where $i \neq j$ and $A_{ii} = 0$.

2. Define D as a diagonal matrix where each D_{ii} is a summation of the elements in the ith row.

3. Calculate the Laplacian matrix L as follows:

$$L = D^{-1/2} A D^{-1/2} \tag{2.51}$$

4. Find the K largest eigenvectors of L, denoted by $\alpha_1, \alpha_2, \cdots, \alpha_K$, and form the

matrix $P \in R^{N \times K}$ as follows:

$$P = [\alpha_1, \alpha_2, \cdots, \alpha_K] \tag{2.52}$$

5. Normalize P and denote it by Q as follows:

$$Q_{ij} = \frac{P_{ij}}{(\sum_j P_{ij}^2)^{\frac{1}{2}}} \tag{2.53}$$

   where each row of Q is a data point in R^K.

6. K-means or similar algorithms can cluster rows of Q into K clusters.

7. If the ith row of Q is a member of cluster j, then assign the original point x_i to the cluster j.

2.4.2 Normalized Cut Clustering

Normalized Cuts (N-Cut) clustering is proposed for image segmentation in [1]. This method constructs a weighted graph of data, where samples are nodes and weights on edges between nodes reflect a measure of similarity between samples. More specifically, edge weights are inversely proportional to the distances between nodes. N-Cut tries to find two optimal partitions in the data which removing the edges between them will have the minimum *cut*. In this setting, a cut is the total value of the removed edges and N-cut for two partitions A and B can be given as follows:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \tag{2.54}$$

and

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v) \tag{2.55}$$

where $w(u, v)$ is the weight between nodes u and v. The $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$ is the total connections between nodes in A and all nodes in the graph. The same definition is valid for $assoc(B, V)$. The following shows required steps for partitioning an image data using N-cut:

1. Given an image dataset, set up the weighted graph of the image with nodes and weights for the connected nodes.

2. Solve the following relation for eigenvectors with the smallest eigenvalues:

$$(D - W)x = \lambda Dx \tag{2.56}$$

   where W is an $N \times N$ symmetrical matrix of weights and D is an $N \times N$ diagonal matrix with $d(i) = \sum_j w(i, j)$ as the ith element on its diagonal. $d(i)$ is the total connection from node i to all of the nodes.

3. Bipartition the graph using the eigenvector with the second smallest eigenvalue that minimizes the N-cut.

4. Decide if the partitions should be subdivided recursively based on the stability of cut and having N-cut less than a predefined value. The number of segments in the above steps can be controlled by the maximum value of N-cut, but this 2-way cut procedure has drawbacks on treating oscillatory eigenvectors and the approach is computationally wasteful [1]. Therefore, instead of 2-way cut a K-way partitioning using all of the eigenvectors based on a given K is suggested.

2.4.3 Voting-K-means

Voting-K-means algorithm combines clustering results of several K-means clusterings for an initial given number of clusters. The resulted co-association sample matrix which shows overall outcome of clusterings is then used to extract the underlying consistent

clusters [41]. Following steps show the required procedure for partitioning data using Voting-K-means for a given initial number of clusters K [41]:

1. Do R times:
    - Randomly select K cluster centers among the N data samples.
    - Organize the N samples in random order, keeping track of the initial data indexes.
    - Run the K-means algorithm with the reordered data and cluster centers and update the co-association matrix according to the partition thus obtained over the initial data indexes.
2. Detect the consistent clusters through the co-association matrix.

2.4.4 DBSCAN

Density-based spatial clustering of applications with noise (DBSCAN) is a well known and widely used clustering approach for arbitrary shape clusters [42].

In this method, clusters are regions with high density of samples which are separated by lower density areas. This general definition suggests that clusters can follow any non-convex and arbitrary shaped geometry.

DBSCAN algorithm requires to have two input parameters available before the task of clustering, ε-neighborhood, and minimum number of points (minPts).

All the members of e-neighborhood of x_i are within epsilon distance (distance usually is defined based on the euclidean distance). for any x_j, and a member of ε-neighborhood of x_i, we have the distance between x_i andx_j defined as $d_{x_i x_j}$.

The ε-neighborhood is the distance from any x_i sample in the data set that any other sample x_j within this distance will be considered as a neighborhood sample for the x_i. In other words, for any neighborhood sample x_j with the the distance d_{ij} from x_i, the following inequality is true:

$$d_{ij} \leq \varepsilon_{neighborhood} \tag{2.57}$$

The minPts defines the possible minimum number of samples for the clusters. In this method, samples with at least minPts number of samples in their neighborhood are called core-samples, and samples with less number of neighborhoods are non-core samples. Core samples are mainly located in the denser area of the data set, while non-core samples belong to regions with less density. The samples which are neither core sample nor non-core sample are considered as outliers.

Based on the above definition, any x_j sample is directly density-reachable from the sample x_i, if there is a chain of intermediate samples which are all core samples like x_i.

In general, DBSCAN starts with an arbitrary sample and check it for being a core sample. If it is a core sample then a cluster will be emerged otherwise it will be assumed as an out-lier sample which has this potential to be assigned to an undiscovered cluster.

2.4.5 Affinity Propagation Clustering

Affinity propagation or clustering by passing messages between data points is introduced in [43], and relies on the similarity matrix of data as the input of algorithm. Here, the number of clusters will be estimated simultaneously and the algorithm doesn't rely on a predefined distribution for clusters. According to this method, an iterative process of sending messages back and forth among the data samples will lead to emergence of clusters and their exemplars or centers. In this context, there are two types of messages to be sent by data samples to each other: *responsibility* and *availability*.

We consider $s(i, k)$ as the similarity value between the sample x_i and sample x_k. When the goal is to minimize the squared error, each similarity is set to a negative squared error. Therefore, we can formulate $s(i, k)$ as the negative Euclidean distance between the samples:

$$s(i, k) = -||x_i - x_k||^2 \tag{2.58}$$

Accordingly, $r(i,k)$ is the responsibility message sent from the sample i to the sample k, which shows the significance of k as an exemplar for i considering all other available exemplars for i. $a(i,k)$ is the *availability* message sent from the potential exemplar k to the sample i, which shows how appropriate is it for the sample i to choose the sample k as its exemplar knowing the vote of other samples to select the sample k as an exemplar. Consequently, the $r(i,k)$ and $a(i,k)$ can be defined as follows:

$$r(i,k) \leftarrow s(i,k) - \max_{k^{'} s.t.\ k^{'} \neq k} \Big\{ a(i,k^{'}) + s(i,k^{'}) \Big\} \tag{2.59}$$

where $s(i,k^{'})$ is the similarity between the sample i and all of the available exemplars except the exemplar k. $a(i,k^{'})$ shows the availability of all of the exemplars excluding the sample k for the sample i.

$$a(i,k) \leftarrow \min \Big\{ 0, r(k,k) + \sum_{i^{'} s.t.\ i^{'} \notin \{i,k\}} \max\{0, r(i^{'},k)\} \Big\} \tag{2.60}$$

where $r(k,k)$ is the self responsibility of the sample k which relies on the $s(k,k)$. Here, $s(k,k)$ is the self similarity or preference of the sample k that could be given as a *prior* knowledge in the beginning of the clustering. Sample k with a large predefined value of $s(k,k)$ has higher chance to be served as an exemplar, therefore, $preference$ values can dictate the final number of clusters.

The self availability of k shows the positive votes or responsibilities sent from all of the samples excluding the sample i to choose the sample k as the exemplar.

$$a(k,k) \leftarrow \sum_{i^{'} s.t.\ i^{'} \neq k} \max\{0, r(i^{'},k)\} \tag{2.61}$$

Having the availability and responsibility values, any sample k that maximizes the $a(i,k) + r(i,k)$ is an exemplar for the sample i.

In summary, availabilities will be updated based on the responsibilities and respon-

sibilities will be updated based on the availabilities. Then, exemplars will be suggested based on the combination of them. This procedure will be repeated iteratively until a predefined condition for terminating the algorithm is satisfied.

In some cases, a numerical oscillation might occur which need to be avoided by using a damping factor between 0 and 1 in the algorithm. It is suggested that a damping factor equal to 0.5 can avoid most of the oscillations. Another suggested approach is adding a tiny amount of noise to the similarity matrix.

Chapter 3

MACE-means Clustering

Majority of the existing clustering methods that can estimate the number of clusters independently need to solve two optimization problems. One optimization problem for estimating the number of clusters and one for clustering data based on the estimated number of clusters. The methods that have optimization stages such as K-means based methods are sensitive to the initial optimization parameter and local optima. These errors, in most of the algorithms will not be detected, which propagates to the results of clustering. One scenario is that number of clusters is chosen correctly, but optimization error causes a difference between the true center and the estimated center. In another scenario, the chosen number of clusters is not correct, and the error of mismatching between samples and true centers is also added to the optimization error. Inspired by [63], [64] and [65], we penalize the errors of clustering and optimization based on a probabilistic approach. This chapter is motivated by searching for a quantitative measure that can evaluate the clustering error [66].

3.1 Our Formulation and Correct Number of Clusters (CNC) Challenges

Observed data of length N, $x = [x_1, \ldots, x_N]^T$ where $x_i \in R_{1\times d}$, is available and the data is generated by m^* cluster model. Centers of these clusters are rows of matrix $c^*_{m^*}$, $c^*_{m^*} \in R_{m^*\times d}$ (with dimension d):

$$c^*_{m^*} = [c^*_1, \ldots, c^*_m]^T, \tag{3.1}$$

The observed data x is a sample of random variable X with the following statement:

$$\begin{pmatrix} X_1 \\ \vdots \\ X_N \end{pmatrix} = \begin{pmatrix} c^*_{x_1} \\ \vdots \\ c^*_{x_N} \end{pmatrix} + \begin{pmatrix} W_1 \\ \vdots \\ W_N \end{pmatrix}, \tag{3.2}$$

$$X = c^*_x + W. \tag{3.3}$$

where c^*_x is the associated centers of the data, i.e., each $c^*_{x_i}$ is an element of $c^*_{m^*}$, and W is the representative of a random variable that demonstrates the variations in the clusters. For example, if the variations are assumed to be from independent and identically distributed (iid) Gaussian distributions, we have $W_i^T \sim \mathcal{N}(0, \sigma_w^2 I_{d\times d})$. Figure 3.1 shows an example of such model in $3-dimensional$ space ($d = 3$) with three centers ($m^* = 3$).

CNC Challenges: A clustering method aims for estimating the correct centers c^*_x. In this estimation, finding the CNC (m^*) is an important task. Here, we model the problem of CNC calculation and clustering as follows:

$$c^*_x \quad \longrightarrow \quad x \longrightarrow \hat{c}_m = [\hat{c}_{m1}, \ldots, \hat{c}_{mm}]^T. \tag{3.4}$$

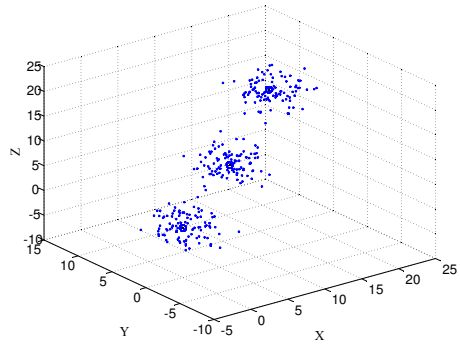


Figure 3.1: Three clusters with 100 samples each ($N = 300$). The three bold points are centers and the cluster variation is $\sigma_w^2 = 3$.

where $\hat{c}_m$ is the vector of estimated centers in m-clustering. Arrows show that from the correct centers, members of the clusters are generated and from the members of the clusters the estimated centers are calculated. The main challenge is to find an optimum m from a feasible range of values, $m \in [m_{min}, m_{max}]$. In an efficient clustering method $\hat{m}$=m^*, i.e., the CNC is found.

3.1.1 Naive K-means and Calculating $\hat{c}_m$

Naive K-means is a clustering algorithm that provides center estimates for a given number of clusters m. For the available m and randomly initialized $\hat{c}_m^0$ (where the superscript zero represents the initial step of the iterative optimization steps), K-means estimates the compactness error y_{Sm}, which is the error between the available data and estimated centers. At each iteration step, the following optimization is solved by K-means:

$$y_{Sm}^l(\hat{c}_m^l) = \|x - \hat{c}_m^l\|_2^2, \qquad \hat{c}_m^{l+1} = \arg\min_{\hat{c}_m^l} y_{Sm}^l(\hat{c}_m^l), \tag{3.5}$$

$$\hat{c}_m = \hat{c}_m^{l_{max}}, \qquad y_{Sm} = y_{Sm}^{l_{max}}. \tag{3.6}$$

where l_{max} is the step when the convergence to a desired compactness error is satisfied.

K-means can converge to the solution of clustering with a reasonable speed but it has some shortcomings that need to be considered before any clustering. One of the issues with K-means is that the objective function in (3.5) makes it limited to clusters with spherical Gaussian distributions. Also, K-means is very sensitive to initialization error for selecting centers of clusters and can be trapped in local optima of the objective function.

CNC Challenges with K-means: Naive K-means can be used when CNC is known *a priori*. However, its wide use is also for cases that CNC is not known. In this scenario, a range of number of clusters is first considered and additional processing compares these number of clusters to come up with an estimate of CNC. For example, in some of these methods, the compactness error (y_{Sm} in (3.6)) is used as a part of the criterion. Validity index methods, such as CH, DB, KL, Sil, and wtertra are examples of such methods. Note that the compactness error itself is a decreasing function of m and cannot provide any estimate of CNC by itself.

3.2 Minimum Average Central Error (MACE)

Considering (3.4) it seems desirable to have estimate of the sample of error[1]:

$$Z_{Sm} = \|C_x^* - \hat{C}_m\|_2^2. \tag{3.7}$$

This error denoted by Average Central Error (ACE), is the error between the true centers with correct number of centers and the estimated centers with m number of centers. In the following, we provide a unique method of estimating this error and will show how comparison of this error for a range of m is an efficient method for CNC estimation. We will show how the available compactness error can be used in estimating the ACE.

[1]Please note that $\hat{c}_m$ is a sample of the random variable $\hat{C}_m$ resulted by the random variable X.

3.2.1 Average Central Error

The ACE, for when the number of clusters is assumed to be m, can be formulated as follows:

$$Z_{Sm} = \sum_{i=1}^{m} Z_{Smi}, \tag{3.8}$$

where Z_{Smi} is the ACE in the ith cluster:

$$Z_{Smi} = \frac{1}{n_i} \| C^*_{x_{mi}} - \hat{C}_{mi} \|_2^2. \tag{3.9}$$

Denote members of this cluster with $X_{mi} = [X^1_{mi}, \cdots, X^{n_i}_{mi}]$. Therefore, we have :

$$\begin{pmatrix} c^*_{x^1_{mi}} \\ \vdots \\ c^*_{x^{ni}_{mi}} \end{pmatrix} \longrightarrow \begin{pmatrix} X^1_{mi} \\ \vdots \\ X^{ni}_{mi} \end{pmatrix} \longrightarrow \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \hat{C}_{mi}, \tag{3.10}$$

$$C^*_{x_{mi}} \longrightarrow X_{mi} \longrightarrow \hat{C}_{mi}, \tag{3.11}$$

$$\hat{C}_{mi} = \frac{1}{n_i} \sum_{j=1}^{n_i} X^j_{mi}, \tag{3.12}$$

Figure 3.2 shows an example in which $m^* = 3$ and $m = 2$. In this case K-means provides an estimate of centers as $\hat{C}_2 = [\hat{c}_{21}, \hat{c}_{22}]^T$. As the figure shows, the associated cluster with $\hat{c}_{21}$ has $n_2 = 9$ members that includes one member of c^*_3 denoted by x^1_{21}, and eight members of c^*_1 denoted by $[x^2_{21}, \cdots, x^9_{21}]$. For these nine members of x_{21} and based on (3.10), we have:

$$c^*_{x_{21}} = [c^*_3, c^*_1, \cdots, c^*_1]^T \longrightarrow x_{21} \longrightarrow [1, 1, \cdots, 1]^T_{1\times 10}\, \hat{c}_{21}. \tag{3.13}$$

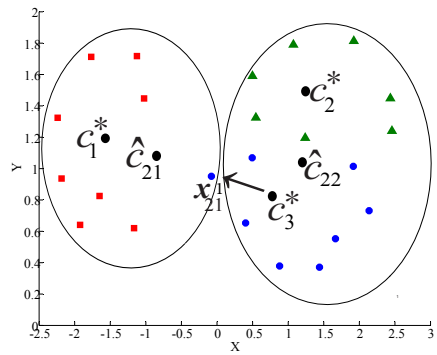


Figure 3.2: In this example, $m^* = 3$, $d = 2$, and $m = 2$. The two estimated centers are $\hat{c}_{21}$ and $\hat{c}_{22}$.

3.2.2 MACE-means criterion

Minimizing ACE over a considered range of m results an estimate of CNC. This method is denoted as MACE-means. In the following, we show required steps for deriving MACE and employing that in data clustering.

3.3 Calculating Minimum Average Central Error (MACE)

The average central error (3.9) is (for details see Appendix A):

$$Z_{Smi} = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{1}{n_i^2}\sum_{j=1}^{n_i} W_j^2 + \frac{2}{n_i^2}\sum_{j\neq k}^{n_i} W_j W_k \tag{3.14}$$

where

$$A_{mi} = \begin{pmatrix} 1-\frac{1}{n_i} & \cdots & \frac{1}{n_i} \\ \vdots & \ddots & \vdots \\ \frac{1}{n_i} & \cdots & 1-\frac{1}{n_i} \end{pmatrix},$$

The variance and mean of the ACE are as follows:

$$E[Z_{Smi}] = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{1}{n_i}\sigma_w^2, \tag{3.15}$$

$$var[Z_{Sm_i}] = \frac{2}{n_i^2}\sigma_w^4. \tag{3.16}$$

where the above relations are result of assuming the same variance in clusters. We show that in practice, the proposed clustering method based on this assumption can deal with clusters with different variances to an acceptable level. An example of the behavior of these statistics is shown in Figure 3.3. As these values show, the standard deviation is

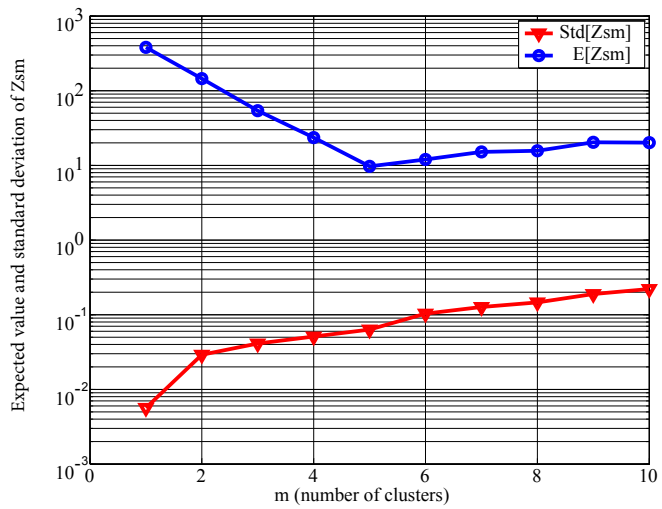


Figure 3.3: Expected value and standard deviation of Z_{Sm} for a range of m (here, $m^* = 5$, $\sigma_w^2 = 1$, $d = 3$, $N = 500$).

much smaller than the expected value itself and can be ignored in comparison. Therefore, in comparing m-clustering, we can focus on estimating and comparing the expected value for feasible sets of m clusters. Consequently, we only need to estimate the $\|A_{mi}C^*_{x_{mi}}\|_2^2/n_i$ in (3.15), which requires to have the variance σ_w^2 in advance. In the following two subsections, we provide methods for estimating these values by only using the observed data.

3.3.1 Estimating $1/n_i\|A_m C^*_{xmi}\|_2^2$ using the available cluster compactness

The available cluster compactness y_{Sm} in (3.6) is a sample of random variable Y_{Sm}. Average cluster compactness error in the ith cluster is:

$$Y_{S_{mi}} = \frac{1}{n_i}\|X_{mi} - \hat{C}_{mi}\|_2^2, \tag{3.17}$$

by simplifying (3.17), we will have (See Appendix B for more details):

$$Y_{Smi} = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 - \frac{2}{n_i^2}\sum_{j\neq k}^{n_i} W_j W_k + \frac{n_i-1}{n_i^2}\sum_{j=1}^{n_i} W_j^2 - \frac{2}{n_i^2}\sum_{j=1}^{n_i} W_j \sum_{k=1}^{n_i} c^*_{x_k} + \frac{2}{n_i}\sum_{j=1}^{n_i} W_j c^*_{x_j}, \tag{3.18}$$

consequently, the expected value and variance of cluster compactness are as follows:

$$E[Y_{Smi}] = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{n_i-1}{n_i}\sigma_w^2, \tag{3.19}$$

$$= \frac{1}{n_i}\sum_{i=j}^{n_i} c^{*2}_{x_j} - \frac{1}{n_i^2}(\sum_{j=1}^{n_i} c^*_{x_j})^2 + \frac{n_i-1}{n_i}\sigma_w^2, \tag{3.20}$$

$$var[Y_{Smi}] = \frac{4\sigma_w^2}{n_i^2}\sum_{j=1}^{n_i} c^{*2}_{x_j} - \frac{4}{n_i^3}\sigma_w^2(\sum_{j=1}^{n_i} c^*_{x_j})^2 + \frac{2(n_i-1)}{n_i^2}\sigma_w^4, \tag{3.21}$$

An example of expected value and standard deviation of Y_{Sm} is shown in Figure 3.4. Comparing (3.20) and (3.21), the variance is of order of $1/n_i$th smaller than the expected value, therefore we can assume that the available y_{Smi} is a good representative of its expected value. Therefore, from (3.19) the following relation will be given:

$$\frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 \approx y_{Smi} - \frac{n_i-1}{n_i}\sigma_w^2. \tag{3.22}$$

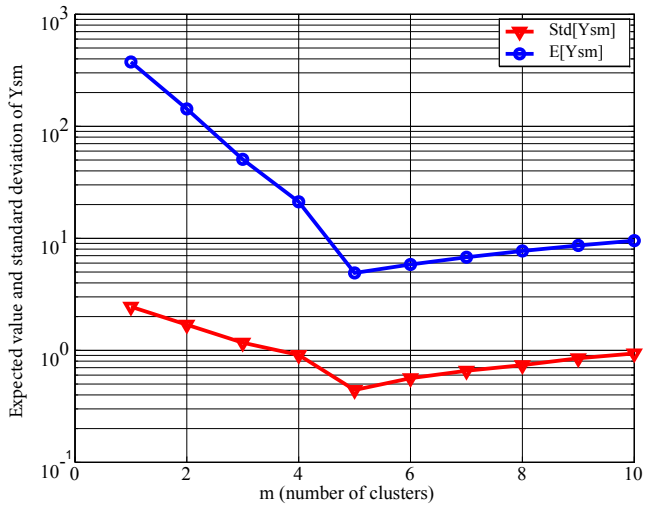


Figure 3.4: Expected value and standard deviation of Y_{Sm} for a range of m (here, $m^* = 5$, $\sigma_w^2 = 1$, $d = 3$, $N = 500$).

3.3.2 Estimating the Variance (σ_w^2) using the available cluster compactness

In this section, we use the available cluster compactness to find an estimate of variance. Using (3.20) for the mi^{th} cluster of m-clustering, we have:

$$y_{Smi} = \frac{1}{n_i}\sum_{j=1}^{n_i} c_{x_j}^{*2} - \frac{1}{n_i^2}(\sum_{j=1}^{n_i} c_{x_j}^{*})^2 + \frac{n_i - 1}{n_i}\sigma_w^2 + \epsilon_i(m), \tag{3.23}$$

where $\epsilon_i(m)$ represents the divergence of y_{Smi} from its mean and therefore, $E[\epsilon_i(m)] = 0$. For the overall cluster compactness of m-clustering we have:

$$y_{Sm} = \sum_{i=1}^{m}(\frac{1}{n_i}\sum_{j=1}^{n_i} c_{x_j}^{*2} - \frac{1}{n_i^2}(\sum_{j=1}^{n_i} c_{x_j}^{*})^2 + \frac{n_i - 1}{n_i}\sigma_w^2) + \epsilon(m), \tag{3.24}$$

$$= T_1(m) + T_2(m) + \epsilon(m), \tag{3.25}$$

where

$$T_1(m) = \sum_{i=1}^{m}(\frac{1}{n_i}\sum_{j=1}^{n_i} c_{x_j}^{*2} - \frac{1}{n_i^2}(\sum_{j=1}^{n_i} c_{x_j}^{*})^2), \tag{3.26}$$

$$T_2(m) = \sum_{i=1}^{m}(\frac{n_i - 1}{n_i}\sigma_w^2), \tag{3.27}$$

$$\epsilon(m) = \sum_{i=1}^{m}\epsilon_i(m), \quad E[\epsilon(m)] = 0, \tag{3.28}$$

A typical behavior of terms $T_1(m)$ and $T_2(m)$ is shown in Figure 3.5. $T_2(m)$ is mainly a

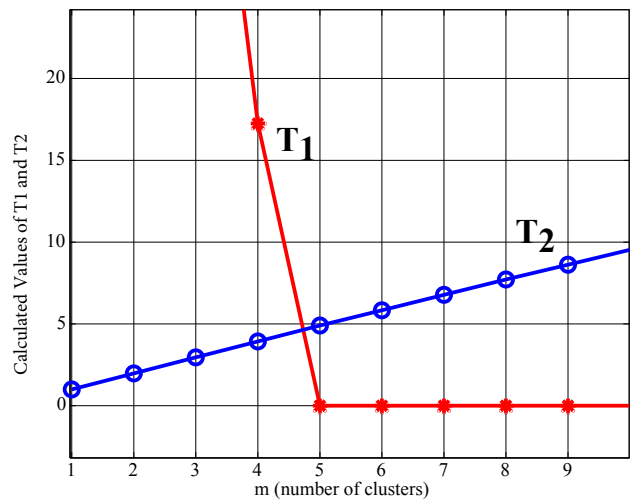


Figure 3.5: $T_1(m)$ and $T_2(m)$ when $m^* = 5$, $\sigma_w^2 = 1$ and, $n_i = 100$.

function of variance and the number of elements in each cluster. In general, if n_i is large enough such that $(n_i - 1)/n_i \approx 1$, we have:

$$T_2(m) = m\sigma_w^2. \tag{3.29}$$

On the other hand, as m grows to be larger than the true m^*, for each mi^{th} cluster, we have $c_{xj}^* \approx c_{mi}^*$, where c_{mi}^* is one single true unknown center. Although c_{mi}^* is not

known, this property causes $T_1(m)$ in (3.26) to be negligible for $m \geq m^*$. Consequently, for the available cluster compactness in (3.23) with the range of m-clustering we have:

$$y_{Sm} = \begin{cases} T_1(m) + T_2(m) + \epsilon(m) & m < m^* \\ T_2(m) + \epsilon(m) & m \geq m^* \end{cases} \tag{3.30}$$

while m^* is unknown, the behavior of this cluster compactness is such that it can help us in finding an estimate of variance. An estimate of variance can be calculated as follows:

$$k = \arg\min_m (y_{Sm}), \tag{3.31}$$

$$\hat{\sigma}_w^2 = \frac{1}{m_{max} - k + 1} \sum_{m=k}^{m_{max}} \frac{y_{Sm}}{\sum_{i=1}^{m} \frac{n_i}{n_i - 1}}. \tag{3.32}$$

where the second equation is the result of using (3.27) and (3.30). Please note that, while it is known that minimizing y_{sm} for a range of m does not provide a consistent estimate of CNC, the above analysis shows that this minimization is beneficial in estimating the variance. Figure 3.6 shows a typical behavior of y_{Sm}, T_2, and how this averaging works. In this example, $m^* = 5$, and two of the clusters are highly overlapped, which has forced the expected value of y_{Sm} to give a minimum at $m = 4$. As it can be seen, T_2 is a function of σ_w^2, which for $m \geq 4$ is the same as y_{Sm}. Therefore, minimum of y_{Sm} can be used to estimate the variance σ_w^2, but it does not give the correct m directly. In other word, if we ignore the effect of $l2$-norm in calculation of y_{Sm}, y_{Sm} will be always a monotonically decreasing function of m. Nevertheless, as Figure 3.6 confirms the minimum of $E[Z_{Sm}]$ occurs at CNC =5.

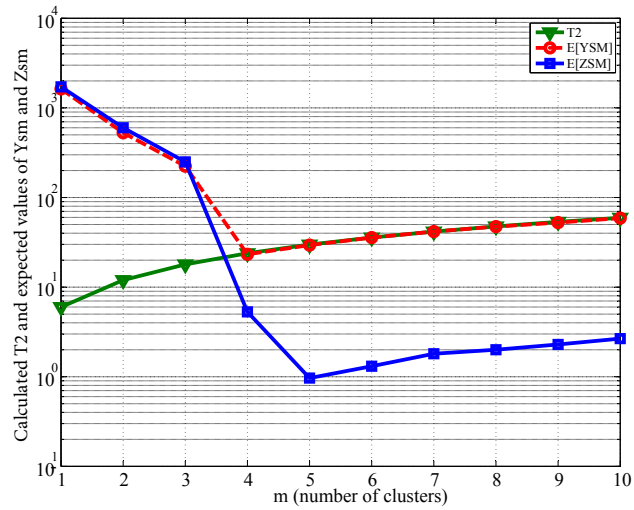


Figure 3.6: Typical behavior of y_{Sm}, ($m^* = 5$, $N = 500$, $\sigma_w^2 = 2$, $\hat{\sigma}_w^2 = 2.09$, $\hat{m} = 5$).

3.4 Average Central Error Estimate

To estimate the ACE based on (3.22) and (3.32), we have:

$$\frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 = y_{Smi} - \frac{n_i - 1}{n_i}\hat{\sigma}_w^2, \tag{3.33}$$

using this result and (3.32) in (3.15) follows as:

$$\hat{z}_{Smi} = y_{Sm} - \frac{n_i - 2}{n_i}\hat{\sigma}_w^2, \tag{3.34}$$

which can be used to provide the following estimate for ACE:

$$\hat{z}_{Sm} = \sum_{i=1}^{m} \hat{z}_{Smi}, \tag{3.35}$$

consequently, the estimated CNC by MACE is as follows:

$$\hat{m} = \arg \min_{m \in [m_{min}, m_{max}]} \hat{z}_{Sm}. \tag{3.36}$$

An example of true and estimate of z_{Sm} is shown in Figure 3.7.

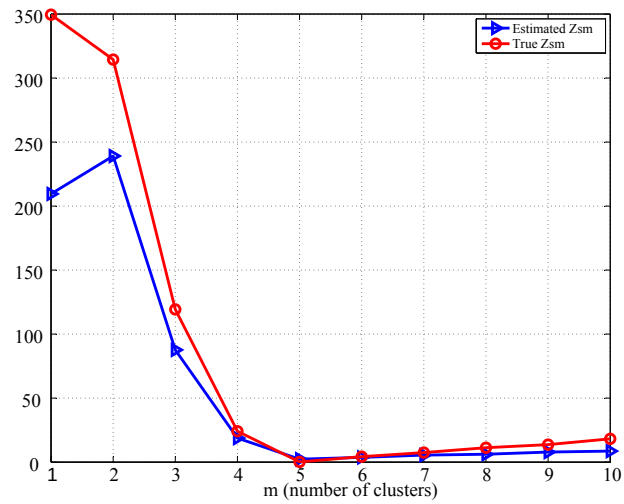


Figure 3.7: True z_{Sm} and its estimate in (3.35), when m ranges between 2 and 15 ($m^* = 5, \hat{m} = 5, N = 500, d = 3$).

3.4.1 MSDL-means clustering

MACE-means clustering can also be denoted as Minimum Structure Description Length (MSDL-means clustering).

According to (3.3), the density function of the observed data based on the true clusters is:

$$f(X; C_x^*) = \frac{1}{(\sqrt{2\pi\sigma_w^2})^N} \exp^{-\|X - C_x^*\|_2^2 / 2\sigma_w^2}, \tag{3.37}$$

Therefore, the description length of the observed samples can be modeled as follows [63]:

$$\mathrm{DL}(X; C_x^*) = -\frac{1}{N} \log_2(f(X; C_x^*)), \tag{3.38}$$

Consequently, the description length of the associated cluster centers for m-clustering is:

$$\mathrm{DL}(\hat{C}_m; C_x^*) = -\frac{1}{N}\log_2(f(\hat{C}_m; C_x^*)) = \log_2\sqrt{2\pi\sigma_w^2} + \frac{\log_2 e}{2\sigma_w^2} Z_{S_m}. \tag{3.39}$$

where the last equation is the result of (3.8). This shows how minimizing the ACE is equivalent to minimizing the description length based on m-clustering that we denote by m-clustering structure description length.

3.5 Computational Complexity Analysis and Comparison

Computational complexity of K-means is $O(mNdl)$, where m is the number of clusters, N is the length of the data, d is the dimension of the data and l is the fixed number of iterations in the optimization stage. Computational complexity of MACE-means is analogous to G-means, which is $O(m) \times O(mNdl)$. This is obtained based on $m+1$ required runs of K-means for estimating the minimum Z_{Sm} at m. Here, Z_{Sm} is calculated based on Y_{Sm} which is given by K-means. Therefore Z_{Sm} calculation doesn't impose a significant computational complexity on the method. Number of iterations for K-means algorithm in G-means is $l = 100$, and in KL, CH, DB, wtertra, Sil and MACE-means is $l = 35$, while this value for Expectation-Maximization (EM) optimization algorithm in PG-means is $l = 10$. Table 3.1 gives a comparison of computational complexity for different methods [2].

Table 3.1: Time complexity

Method	Time complexity	
MACE-means	$O(m) \times O(mNdl)$	
KL and CH	$O(Nd) \times O(mNdl)$	[67]
DB and wtertra	$O(d(m^2+N)) \times O(mNdl)$	[67]
Sil	$O(dN^2+Nm) \times O(mNdl)$	[67]
G-means	$O(m) \times O(mNdl)$	[32]
PG-means	$O(m^2Nd^2l + mN\log(N))$	[34]

[2] Time complexity of DaSpec was not available, but it was comparable to indexed-based clustering methods. In our simulation experiments, the complexity of X-means will be discussed in Section 3.6.

3.6 Experimental Results

In this section we compare the performance of MACE-means clustering with other known approaches. The compared methods are stand alone approaches, such as PG-means [34], G-means [32], X-means [31], and Data Spectroscopic clustering (DaSpec) or they are validity indexes used with K-means, such as Silhouette (Sil) [25], Davies-Bouldin index (DB), Calinski-Harabasz index (CH) [26], Krzanowski-Lai index (KL) [27] and, weighted inter-to intra-cluster ratio (wtertra) [28].

Our results are shown for the following three sets of data:

1. Six available data sets from UCI Machine Learning repository [68], that satisfy clustering problem statement in Section 3.1: Breast, Vertebral [69], Seeds, Wave Forms, Multiple Features (dutch handwritten) and Water Treatment Plant. Table 3.2 shows characteristics of these data sets.

2. Four synthetic data sets (S1 to S4) introduced in [70] are selected because of their Gaussian nature which are suitable for MACE-means clustering. These four data sets are generated based on different levels of overlap between Gaussian clusters. Table 3.2 shows characteristics of these data sets.

3. A large set of synthetic data in 2D and 3D feature spaces with the main focus on estimating the number of clusters in low dimension. The sets are generated with random centers and various levels of overlapping.

The experimental result for the first data set is shown in Table 3.3. It includes the mean and STD of the estimated number of clusters for 50 runs. The ARI and VI values are also averaged over 50 runs. The CNC is presented by m^*. Note that for some of the data sets, X-means did not converge to a solution, these data sets are marked by (N/A).[3]

[3] In calculating the computational complexity, X-means seems to be comparable with MACE-means and G-means. However, even though X-means is based on Kd-tree [31], that is supposed to speed up the method, the algorithm is very slow and in occasions does not converge.

Table 3.2: Real and synthesized benchmark data sets from the literature.

Data set	Number of data vectors	Dimension of data	Number of clusters
Breast	699	9	2
Vertebral	310	6	3
Seeds	210	7	3
Wave Form	5000	21	3
Multiple Features	2000	649	10
Water Treatment Plant	527	38	13
$S_1 - S_4$	5000	2	15

ARI and VI indexes are only calculated for data sets that have the true class labels and clustering methods that provide the estimated labels in addition to the estimated number of clusters. As the table shows, most of the methods give a robust estimation, i.e., their STD values are negligible. However, on average, MACE-means suggests a closer estimation to the accurate number of clusters and better values for ARI and VI.

The experimental results for 50 runs on the second data set in 2D space is presented in Table 3.4, where the number of clusters is known and the true labels are not available. As it is shown in the table, MACE-means, Sil, DB and CH are giving the most accurate estimations of the number of clusters. But among the mentioned methods, only MACE-means estimates the m^* *robustly* with zero error. The remaining of the methods such as X-means, G-means and PG-means tend to overestimate the number of cluster. Our synthetic 2D data sets are generated by random selection of centers in a square of 20×20 for a range of true number of clusters. The variance in the clusters is one or two, and each data set is generated with 100 samples per cluster. Table 3.5 shows the results of different clustering methods for 50 runs over the explained data sets. For example, the first value in the second column ($\mathbf{3}\pm\mathbf{0}$) is the average of 50 simulations. Where for all of the simulations, the cluster variance is one and in each run the algorithm selects three random centers with uniform distribution in the square of 20×20. Note that with this set up we are covering a large set of possibilities as in each run there is a uniform chance

Table 3.3: Mean and standard deviation of estimated number of clusters ($E[\hat{m}] \pm STD[\hat{m}]$) for real data sets (average over 50 runs).

$E[m^*] \pm STD[m^*]$						
Method	Vertebral	Breast	Seeds	Multiple Features	Wave Forms	Water Treatment Plant
	$m^* = 3$	$m^* = 2$	$m^* = 3$	$m^* = 10$	$m^* = 3$	$m^* = 13$
MACE-means	**3±0**	**2±0**	**3 ±0**	**11±0**	**3±0**	**10.1±0.31**
ARI	N/A	0.211±0.423	0.119±0.292	0.019±0.077	0.035±0.063	N/A
VI	N/A	0.081±0.162	0.111±0.273	0.149±0.599	0.087±0.349	N/A
X-means	N/A	N/A	8±0	16±0	N/A	N/A
G-means	5±0	92±0	2±0	33±0	13±0	6±0
ARI	N/A	0.011±0.023	0.078±0.191	0.010±0.040	0.012±0.048	N/A
VI	N/A	0.849±1.698	0.136±0.335	0.209±0.838	0.133±0.533	N/A
PG-means	1±0	10±0	1 ±0	1±0	6±0	1±0
ARI	N/A	0.041±0.082	0±0	0±0	0.018±0.074	N/A
VI	N/A	0.441±0.883	0.183±0.448	0.143±0.575	0.087±0.349	N/A
DaSpec	2±0	1±0	2±0	1 ±0	1±0	7±0
ARI	N/A	0±0	0.076±0.187	0±0	0±0	N/A
VI	N/A	0.161±0.323	0.135±0.332	0.143±0.575	0.068±0.2746	N/A
Sil+K-means	2±0	2±0	2±0	2±0	2±0	2 ±0
DB+K-means	2±0	2±0	2±0	7 ±0	2±0	5±0
KL+K-means	2±0	2±0	3±0	18±0	3±0	2±0
wtertra+K-means	3±0	4±0	3±0	4±0	4±0	4±0
CH+K-means	2±0	2±0	3±0	18±0	2±0	5±0

for different degrees of cluster overlapping.

For each method, mean and standard deviation of the estimated number of clusters ($\hat{m}$) along with the success rate in predicting the CNC is provided in the Table 3.5. The success rate is the percentage of the times that the true number of clusters is estimated correctly. As the table shows, by increasing the number of generated clusters as well as increasing the variance, recognizing the overlapped clusters will be a challenging task. All the comparing methods seem to tolerate and distinguish overlapping clusters with a minimum distance between centers equal to $3\sigma_w^2$, while MACE-means performs well even for center distances as small as $2\sigma_w^2$. This performance was evaluated by 90 percent success rate in estimating the CNC. Table 3.5 also confirms that MACE-means provides a reliable success rate as the mean is closer to CNC and *robustness* of MACE-means

Table 3.4: Mean and standard deviation of estimated number of clusters ($E[\hat{m}] \pm STD[\hat{m}]$) for S data sets (average over 50 runs).

	$E[\hat{m}] \pm STD[\hat{m}]$			
Method	S_1 $m^* = 15$	S_2 $m^* = 15$	S_3 $m^* = 15$	S_4 $m^* = 15$
MACE-means	**15±0**	**15±0**	**15±0**	**15±0**
X-means	20±0	19±0	16±0	N/A
G-means	95±0	77±0	87±0	63±0
PG-means	19±0	30±0	18±0	24±0
DaSpec	5±0	1±0	1±0	1±0
Sil+K-means	14.03±0.18	14.03±0.18	**15±0**	**15±0**
DB+K-means	14±0	14.03±0.18	15.96±0.18	13.96±0.18
KL+K-means	15.96±0.18	4±0	2±0	5±0
wtertra+K-means	7.26±1.46	14.03±0.18	15.86±0.73	17.03±0.18
CH+K-means	15.96±0.18	15.96±0.18	**15±0**	**15±0**

relative to other methods is proved by the small standard deviation values. The average of ARI and VI values over 50 simulations also confirms the accuracy of MACE-means compared with other methods, where larger ARI and smaller VI values show better clustering results. It seems that one of the main factors for larger standard deviation in other methods is due to level of sensitivity to the initial optimization parameters and being trapped in local minima. Table 3.6 shows similar results when the data dimension is increased to 3. In this case, for each of the 50 runs the centers are chosen with uniform distribution in a cube of 20×20×20. This increase in dimension of the data makes it easier to distinguish the clusters. Therefore, the methods were able to give a better clustering result for a larger number of clusters and variance values compared to Table 3.5. As the table shows, MACE-means is the most accurate and robust method in estimating the m^* for all of the data sets. The averaged ARI and VI values also confirm the superiority of MACE-means over the other methods.

Table 3.5: Mean and standard deviation of estimated number of clusters ($E[\hat{m}] \pm STD[\hat{m}]$) for our 2D synthetic data sets (averaged over 50 runs).

	$m^* = 3$		$m^* = 4$		$m^* = 5$		$m^* = 6$	
d=2	$\sigma_w^2 = 1$	$\sigma_w^2 = 2$	$\sigma_w^2 = 1$	$\sigma_w^2 = 2$	$\sigma_w^2 = 1$	$\sigma_w^2 = 2$	$\sigma_w^2 = 1$	$\sigma_w^2 = 2$
MACE-means	3±0	**3±0**	**3.9±0.307**	**3.95±0.223**	**5.3±0.470**	**4.75±0.910**	**5.95±0.223**	**5.65±0.587**
Success (%)	100	**100**	**90**	**95**	**70**	**45**	**95**	**55**
ARI	0.166±0.408	0.133±0.327	0.067±0.191	0.072±0.203	0.096±0.305	0.074±0.235	0.078±0.271	0.081±0.282
VI	0±0	0.082±0.202	0.123±0.348	0.132±0.374	0.012±0.038	0.066±0.209	0.017±0.061	0.006±0.021
X-means	3±0	2.75±0.444	3.8±0.410	3.1±0.967	4.4±0.502	3.75±0.966	5.4±1.095	5.4±1.500
Success (%)	100	75	80	50	40	30	65	35
G-means	3±0	2.5±0.51299	3.6±0.680	3.5±1.357	4.7±0.470	4.05±0.944	5.7±0.656	5.25±0.850
Success (%)	100	50	70	20	70	30	50	35
ARI	0.166±0.408	0.089±0.218	0.038±0.109	0.058±0.164	0.096±0.305	0.074±0.235	0.078±0.271	0.079±0.275
VI	0±0	0.103±0.254	0.132±0.374	0.104±0.295	0.012±0.038	0.066±0.209	0.017±0.061	0.014±0.051
PG-means	3±0	3.25±1.118	3.45±0.686	3.15±0.988	4.1±0.852	3.3±0.571	5.45±0.510	4.95±0.686
Success (%)	100	50	55	55	40	5	45	5
ARI	0.166±0.408	0.071±0.175	0.040±0.113	0.058±0.164	0.096±0.303	0.060±0.191	0.078±0.271	0.081±0.281
VI	0±0	0.155±0.381	0.118±0.334	0.105±0.297	0.0133±0.042	0.059±0.187	0.017±0.060	0.007±0.027
DaSpec	3±0	2.2±0.410	2.85±1.04	2.25±1.118	2.95±1.276	2.15±0.875	3.55±0.604	2.25±0.716
Success (%)	100	20	35	10	5	0	0	0
ARI	0.166±0.408	0.088±0.217	0±0	0±0	0.078±0.247	0.060±0.191	0.047±0.164	0.036±0.126
VI	0±0	0.093±0.228	0.173±0.490	0.173±0	0.027±0.087	0.059±0.187	0.045±0.158	0.076±0.266
Sil+K-means	2.5±0.512	2.2±0.410	2.95±0.887	3.15±0.988	3.1±0.552	3.05±0.686	3.95±1.145	3.45±1.145
Success (%)	50	20	35	55	5	5	10	5
DB+K-means	3±0	2.2±0.410	3.35±0.670	3.15±0.988	3.7±0.571	2.95±0.510	4.3±1.080	4±1.297
Success (%)	100	20	45	55	5	0	10	15
KL+K-means	3±0	3.65±1.308	4.5±1.539	3.15±0.988	3.4±1.095	4.6±2.303	5.35±2.539	7.15±2.719
Success (%)	100	30	50	55	10	5	15	5
wtertra+K-means	3±0	3±0	2.9±0.640	3.4±0.598	4.25±3.35±0.670	3.5±0.760	3.8±0.767	
Success (%)	100	100	15	45	60	10	0	0
CH+K-means	3±0	2.75±0.444	3.7±0.470	3.15±0.988	4.7±0.470	3.7±0.923	5.85±0.366	5.3±0.923
Success (%)	100	75	70	55	70	25	85	45

As the Table 3.5 and Table 3.6 show, MACE-means gives the best performance over the 2D and 3D Gaussian data sets. X-means is the second best method in terms of estimating the number of clusters. G-means and PG-means are the next successful methods but have less accuracy compared with MACE-means in terms of ARI, VI and estimated number of clusters. The provided results are highly affected by the number of iterations (see l in Section 3.5) in each run of the K-means or EM algorithm. In other words, both

of the mentioned algorithms are sensitive to the initialization error and can easily get trapped in local minima of their objective functions. To solve this issue, it is suggested to choose a large enough value for l and then choose the best solution among all of the iterations (the most optimized objective function). The choice of l should be a trade off between accuracy and computational complexity. In all of the simulations, we limited the l parameter of our method to its minimum value which is used by other methods.

3.7 Conclusions

In this Chapter, MACE-mean clustering was proposed as a wrapper around K-means for simultaneous clustering and estimating the correct number of clusters. We defined the Average Central Error (ACE) and proposed a method for calculation of CNC based on minimizing this error. One of the main contributions of this work was to provide probabilistic bounds for the unavailable ACE using the available cluster compactness. In clustering approaches, the initialization error propagates to the clustering process and affects estimation of CNC. Robustness of MACE-means is due to the choice of a single objective function that is used in both clustering and order selection. Comparison between MACE-means and widely used clustering methods demonstrated the robustness and accuracy of the proposed method in estimating the CNC, even for highly overlapped clusters. Time complexity of clustering methods are compared and it was shown that MACE-means has one of the lowest time complexities among them.

Table 3.6: Mean and standard deviation of estimated number of clusters ($E[\hat{m}] \pm STD[\hat{m}]$) for our 3D synthetic data sets (averaged over 50 runs).

	$m^* = 5$		$m^* = 6$		$m^* = 7$		$m^* = 8$	
d=3	$\sigma_w^2 = 2$	$\sigma_w^2 = 3$	$\sigma_w^2 = 2$	$\sigma_w^2 = 3$	$\sigma_w^2 = 2$	$\sigma_w^2 = 3$	$\sigma_w^2 = 2$	$\sigma_w^2 = 3$
MACE-means	**5±0.324**	**4.8±0.615**	**5.95±0.394**	**5.75±0.444**	**6.85±0.489**	**6.7±0.571**	**7.9±0.447**	**7.6±0.598**
Success (%)	**75**	**76**	**85**	**75**	**75**	**75**	**80**	**65**
ARI	0.098±0.309	0.066±0.211	0.079±0.274	0.057±0.200	0.064±0.240	0.061±0.228	0.057±0.228	0.043±0.173
VI	0±0	0.007±0.024	0.014±0.051	0.073±0.255	0.024±0.091	0.032±0.121	0.021±0.084	0.039±0.156
X-means	4.95±0.510	4.65±0.587	5.8±0.410	5.5±0.888	6.9±1.209	6.2±0.951	7.5±1.147	7.15±0.812
Success (%)	75	70	80	55	35	40	35	40
G-means	5±0.648	4.6±0.820	5.8±0.695	5.7±0.978	6.8±0.695	6.2±0.615	7.5±1.051	7±1.123
Success (%)	60	60	65	40	65	30	30	25
ARI	0.090±0.287	0.066±0.211	0.079±0.274	0.056±0.194	0.057±0.215	0.058±0.217	0.050±0.202	0.043±0.173
VI	0.026±0.082	0.075±0.239	0.014±0.051	0.063±0.220	0.028±0.106	0.028±0.105	0.030±0.121	0.039±0.156
PG-means	4.7±0.656	4.5±1.051	5.95±0.944	5.4±0.994	6.85±0.988	5.9±0.788	7.1±1.071	6.65±0.988
Success (%)	65	40	60	30	65	20	15	25
ARI	0.097±0.306	0.046±0.146	0.078±0.272	0.053±0.183	0.056±0.211	0.057±0.214	0.049±0.198	0.042±0.171
VI	0.012±0.038	0.079±0.252	0.015±0.055	0.057±0.197	0.030±0.115	0.027±0.101	0.028±0.113	0.039±0.159
DaSpec	3.2±0.951	2.65±0.875	3.5±1.051	3±0.794	3.8±0.894	2.8±1.056	3.5±1.192	2.5±0.760
Success (%)	10	0	0	0	0	0	0	0
ARI	0.056±0.179	0.020±0.064	0.027±0.095	0.011±0.040	0.007±0.027	0.033±0.127	0±0	0.009±0.038
VI	0.075±0.237	0.117±0.371	0.077±0.266	0.111±0.387	0.109±0.410	0.061±0.231	0.129±0.519	0.096±0.386
Sil+K-means	3.9±0.967	3.6±1.231	4.3±1.260	4.25±1.251	5.5±1.277	4.25±1.208	5.8±1.641	5.1±1.651
Success (%)	30	30	15	20	20	0	15	5
DB+K-means	3.95±0.887	3.85±1.136	4.35±0.988	4.2±1.239	5.3±0.978	4.45±0.944	5.9±1.619	5.55±1.316
Success (%)	30	35	10	25	10	0	15	10
KL+K-means	5.45±1.959	4.5±1.192	5.95±1.848	6.35±2.73	7.75±3.109	6.8±3.205	11.2±3.847	9.75±3.850
Success (%)	55	50	45	30	45	0	5	5
wtertra+K-means	3.75±0.786	3.9±0.911	3.9±1.29	4.45±0.887	4.35±1.424	4.3±1.454	4.45±1.234	4.5±1.317
Success (%)	20	20	25	10	10	10	0	0
CH+K-means	4.6±0.753	4.5±0.827	5.65±0.745	5.45±0.759	6.4±1.046	5.75±1.482	7.3±0.923	6.95±0.998
Success (%)	70	65	75	60	65	35	55	35

Chapter 4

Signature Testing (Sigtest) in Clustering

We propose a new statistical test denoted by signature testing (Sigtest) with the application in clustering and image classification. Sigtest relies on probabilistic validation of empirical distribution function of data. We implement Sigtest to estimate the number of clusters in hierarchical and partitional clustering. In addition we propose a new adaptive estimation of the vocabulary size in image classification. Simulation results on both real and synthetic data confirm superiority of Sigtest over existing statistical tests in both hierarchical and partitional clustering as it estimates the number of clusters more accurately. Sigtest also shows advantages in terms of adjusted rand index (ARI) and variation of information (VI). In addition, using Sigtest for adaptive choice of vocabulary size in bag of visual words improves the efficiency of the Support Vector Machines (SVM) classifier as well as reducing the time complexity of the overall algorithm.

4.1 Introduction

Sigtest can be employed for estimating the number of clusters. Unlike existing statistical tests it can be used with *any* prior assumption on the distribution of the clusters (Dcdf). In this Chapter, we focus on the application of Sigtest when prior assumption on distribution of clusters is Gaussian. Using the method with any other prior assumption is analogous to what is presented here.

The chapter is organized as follows: Section 4.2 gives details of deriving Sigtest. Section 4.3 shows applications of Sigtest in hierarchical and partitional clustering along with BOVW image classification. Section 4.5 shows simulation results of using Sigtest on real and synthetic data sets, and Section 4.6 presents the conclusion.

4.2 Signature Testing

As discussed in Section 2.2, the goal of all statistical testing methods is to propose an efficient distance measure between the Dcdf and ecdf. In the following, we elaborate on an idea that our desired cdf can be represented by its signature through transformation of data which can benefit drastically in defining a distance measure. A signature of a cdf is a new cdf that is derived from the original cdf, which has much smaller variations and represents a set of dense data samples. The following example illustrates the notion of signatures.

Consider a data that has been generated independently and identically distributed (iid) from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with zero mean and variance σ^2. Figure 4.1 (a) shows the histogram of 500 samples of a Gaussian random variable with zero mean and $\sigma = 1$, $\mathcal{N}(0, 1)$.

Figure 4.1 (b) shows 100 of such samples (length of the data is 500) that are plotted simultaneously over each other. As this figure shows, the original samples vary between $\pm 3.5\sigma$ ($\sigma = 1$) which theoretically happens with the probability of 0.9995 for a Gaussian

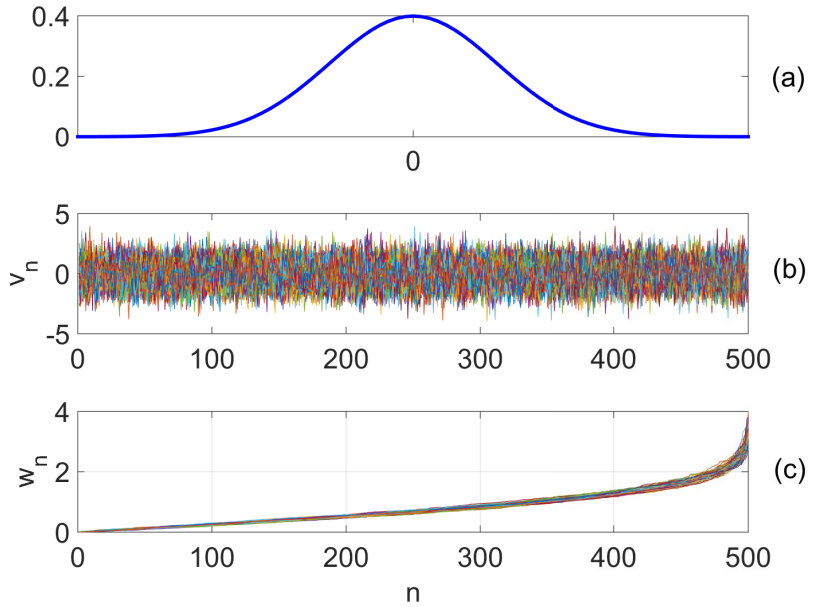


Figure 4.1: (a) Histogram of 500 randomly generated samples belong to a Gaussian distribution with zero mean and unit variance. (b) The actual data without any manipulation. (c) Sorted absolute values of the samples.

distribution. However, Figure 4.1 (c) shows the same 100 samples when the absolute value of those Gaussian samples are sorted.

Based on this observation, it seems that the cdf of the sorted absolute version of the Gaussian distribution has much smaller variance compared to the cdf of data itself. For example, while the ranges in Figure 4.1 (b) and (c) are identical ($0 < xlabel < 500$, and $0 < ylabel < 4$), the area shown in Figure 4.1 (c) compare to the area shown in Figure 4.1 (b) for exactly the same data is much smaller and denser. Figure 4.2 shows the same transformation on a mixture of two Gaussians that leads to a denser area (Figure 4.2 (c)) which can be used for signature testing.

We denote such dense transformations of the original cdfs as the signature of those cdfs. In the following, we describe how those signatures can be used as statistical tests to compare ecdfs with the desired cdf.

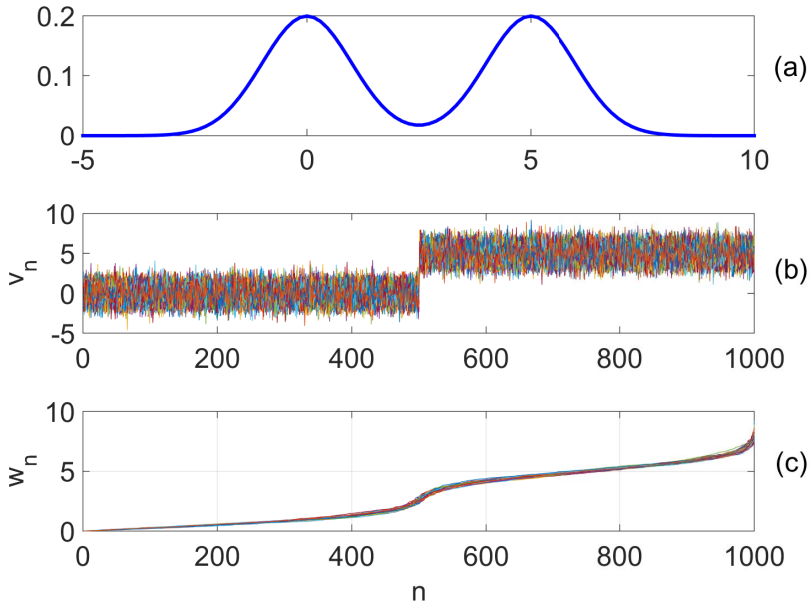


Figure 4.2: (a) Histogram of 1000 randomly generated samples drawn from a Gaussian mixture model with mean values equal to zero and 5, and unit variances. (b) The actual data without any manipulation. (c) Sorted absolute values of the samples.

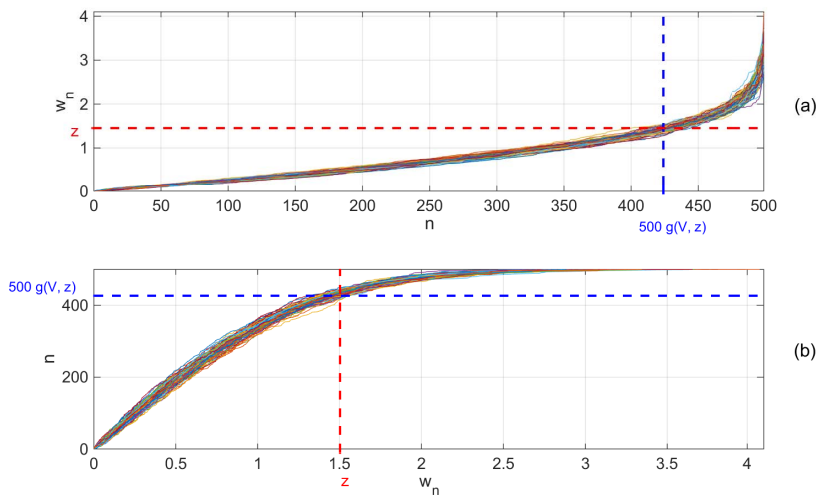


Figure 4.3: (a) sorted absolute version of 100 samples belong to a Gaussian distribution with zero mean and unit variance. (b) the same plot while x axis and y axis are swapped.

4.2.1 Formulation of Signature testing (Sigtest)

Let $V = [V_1, V_2, \cdots, V_N]^T$ be a vector of iid random variables of length N, where $v = [v_1, v_2, \cdots, v_N]^T$ is a sample of that random variable. For any given z the following random variable:

$$g(V, z) = \frac{1}{N} \sum_{i=1}^{N} I(v_i, z) \tag{4.1}$$

depicts the averaged number of v_is less than z, where $I(v_i, z)$ was defined in (2.26) . If we let $w = [w_1, w_2, \cdots, w_N]^T$ to represent the vector of sorted absolute values of v, then:

$$g(v, w_n) = \frac{n}{N} \tag{4.2}$$

that means the $g(v, w_n)$ is the normalized index of the absolute sorted version of v. Figure 4.3 (a) and (b) show this relationship by swapping the xlabel and ylabel.

It can be shown that [64]:

$$E[g(V,z)] = F_a(z) \tag{4.3}$$

$$var[g(V,z)] = \frac{1}{N}F_a(z)(1-F_a(z)) \tag{4.4}$$

where F_a is the cdf of absolute value of v_is.

This confirms that the variance of the index of the sorted version is of order of $1/N^{th}$ of the original random variable (F_a and F have variances of the same order). Consequently sorted version of the original random variable is a good signature candidate of the original random variable.

This signature has been used in denoising approaches to validate which small values of an observed samples are members of the noise distribution by comparing the sorted version of the absolute value of the observed data with the following boundaries:

$$\overline{S(z,\alpha)} = E[g(V,z)] + \alpha\sqrt{var[g(V,z)]} \tag{4.5}$$

$$\underline{S(z,\alpha)} = E[g(V,z)] - \alpha\sqrt{var[g(V,z)]}$$

where $\overline{S_K(z,\alpha)}$ and $\underline{S_K(z,\alpha)}$ are the probabilistic upper and lower bounds and the α parameter is related to probabilistic validation to satisfy the desired confidence probability p_c for estimating the bounds of the index[1].

An example of this signature testing for denoising is shown in Figure 4.4, where (a) is noisy signal with $SNR = 5$. Figure 4.4 (b) shows sorted version of the noisy data (blue line) and confidence region of the noisy data (red line). As the figure shows, at $w_n = 1.47$ data is crossing the lower boundary which suggests values between zero and

[1]Based on the Chebychev inequality, for the confidence probability p_c:

$$Pr\{|g(V,z) - E[g(V,z)]| \leq J\} = p_c \tag{4.6}$$

we will have $J = \alpha\sqrt{var[g(V,z)]}$, and it leads to the $\alpha \leq \sqrt{\frac{1}{1-p_c}}$ which gives the upper limit of α.

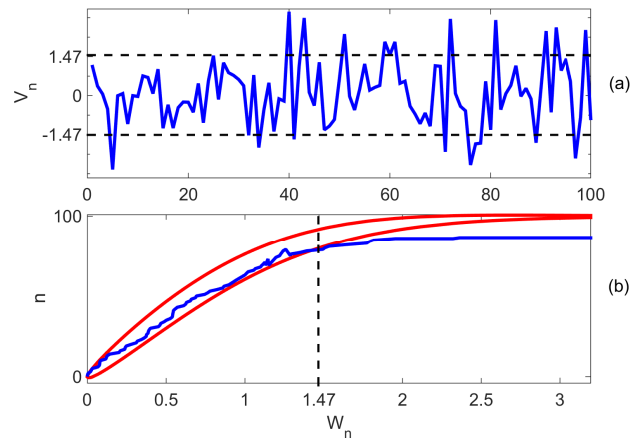


Figure 4.4: (a) Solid blue line is 100 samples of the noisy observed data (SNR = 5). (b) Blue line is 100 samples of the sorted absolute values of the noisy observed data crossing the noise confidence region (Red line) at $w_n = 1.47$. The area between the red lines is the noise confidence region with probability 0.999997.

w_n are related to noise.

While this idea can be used for any assumption on the desired distribution, in the following we provide the details for when the desired distribution is Gaussian.

Calculation of F_a: Calculating a closed form for cdf of the absolute value of the random variable can be cumbersome. Note that as the desired cdf is known, in practical applications, we can find a good estimate of F_a by using sampling approaches such as Importance Sampling, Inverse Transform Sampling and Markov Chain Monte Carlo (MCMC).

For a Gaussian distribution F, however, the closed form is (details are in Appendix C):

$$F_a(z) = \frac{1}{2}[erf(\frac{z-\mu}{\sigma\sqrt{2}}) + erf(\frac{z+\mu}{\sigma\sqrt{2}})] \tag{4.7}$$

where μ and σ are mean and standard deviation of V respectively. The above cdf is identical to (4.3) and can be used in (4.5) to define the signature of Dcdf. In the following section, we show that this result can be easily extended for Gaussian mixture

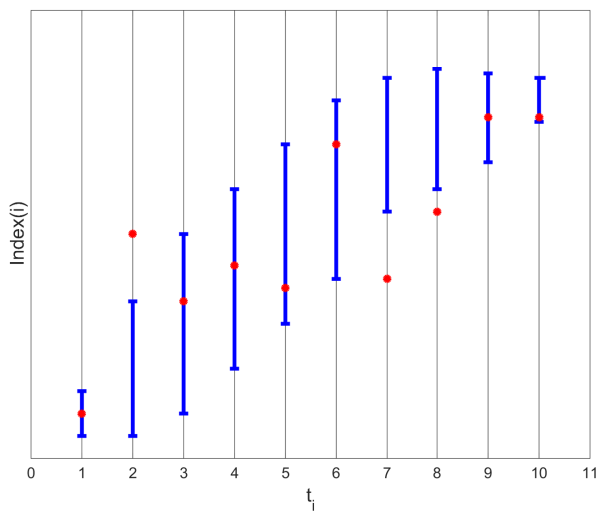


Figure 4.5: Ten observed samples (red dots) and their corresponding boundaries (blue bars).

models.

4.2.2 Sigtest in Statistical Testing

In Sigtest statistical testing, we would like to verify whether the observed data $y = [y_1, \cdots, y_N]^T$ (defined in Section 2.2) belongs to a desired cdf (Dcdf). To use the sorting signature for such verification, we first sort the absolute value of the observed y and denote it by $t = [t_1, \cdots, t_N]^T$. We then compare i, which is the index of t_i for any $z = t_i$, with $\overline{S(t_i, \alpha)}$ and $\underline{S(t_i, \alpha)}$ that are the upper and lower bounds from the Dcdf in (4.5) for that observed t_i. Consequently, each observed data will be tested against the bounds as follows:

$$c_i(\alpha) = \begin{cases} 0 & \underline{S(t_i, \alpha)} < \frac{i}{N} < \overline{S(t_i, \alpha)} \\ 1 & otherwise \end{cases} \tag{4.8}$$

this value is a flag to check whether the ith sample is inside the provided boundaries by Dcdf.

The overall $Sigtest_{score}$ for y is suggested to be the percentage of consistency of the observed data with the Dcdf:

$$Sigtest_{score}(\alpha) = \frac{1}{N}\sum_{i=1}^{N} c_i(\alpha) \tag{4.9}$$

$Sigtest_{score}$ should be less than critical value T to accept the null hypothesis that ecdf is a sample of the Dcdf (H_0), otherwise it is not a sample of the Dcdf (H_1):

- H_0: The observed sample (ecdf) is a sample of the Dcdf $\leftrightarrow Sigtest_{score}(\alpha) < T$
- H_1: The observed sample (ecdf) is not a sample of the Dcdf $\leftrightarrow Sigtest_{score}(\alpha) \geq T$

For example, Figure 4.5 shows the results for $(\alpha) = 4.5$ when 10 observed data samples are sorted. $\overline{S(t_i, \alpha)}$ and $\underline{S(t_i, \alpha)}$ for each sample are calculated, where blue bars represent the validated boundaries of the samples. As the figure shows, samples 2, 7 and 8 are outside of the boundaries and the rest are within the boundaries, where $Sigtest_{score}(4.5) = 0.7$.

Similar to other statistical testing, The parameters α and T need to be chosen through some statistical analysis. In Appendix D, we provide detailed steps in choosing α and T for the case of Gaussian distribution using genetic algorithm. The optimal values with this approach are α=0.53, $T = 1.72$ respectively. Note that for any other desired cdf same approach can be used for calculation of this parameters. Algorithm 1 shows $Sigtest$ in statistical testing.

4.3 Sigtest in Clustering

In the following, we demonstrate how Sigtest can be used as a statistic test for estimating the number of clusters in hierarchical and partitional clustering. We also illustrate advantages of using Sigtest in image retrieval based on BOVW in order to estimate the size of visual vocabulary and improving the accuracy of classification.

Algorithm 1 Sigtest in Statistical Testing

Input: Input sample y of length N, model assumption (desired cdf), critical values T and α.
Output: result of the test for the model.

```
Sigtest_score(α) ← 0
t ← sort(abs(y))
compute S̄ and S̲ from (4.5)
for i = 1 to N do
    if i/N > S̄  or  i/N < S̲ then
        Sigtest_score(α) ← Sigtest_score(α) + 1/N
    end if
end for
if Sigtest_score(α) < T then
    H_0: y ∈ Dcdf
else
    H_1: y ∉ Dcdf
end if
```

4.3.1 Sigtest in Hierarchical Clustering

In divisive or top down hierarchical clustering methods, we start at the top with all of the samples in one cluster. If the number of clusters is not known, samples will be split recursively until every cluster has only one sample. Adding a splitting criterion at each splitting stage can also estimate the number of clusters in hierarchical clustering.

As a result, the process of cluster splitting will be stopped at the estimated number of clusters. In general, the splitting criterion is a statistic test. It compares the ecdf of the data and the desired cdf (Dcdf) as a splitting criterion:

- H_0: The cluster data (ecdf) is a sample of the Dcdf (Split: No).
- H_1: The cluster data (ecdf) is not a sample of the Dcdf (Split: Yes).

Figure 4.6 shows an example of such hierarchical clustering with order selection in form of splitting criterion. Methods such as G-means and Dip-means clustering are examples of such clustering. The shaded block is the splitting step, i.e., the statistical test. The splitting test for G-means is AD test (Subsection 2.2.2) and for Dip-means is Dip test (Subsection 2.2.3).

We propose to use Sigtest as the splitting criterion of the hierarchical methods. This

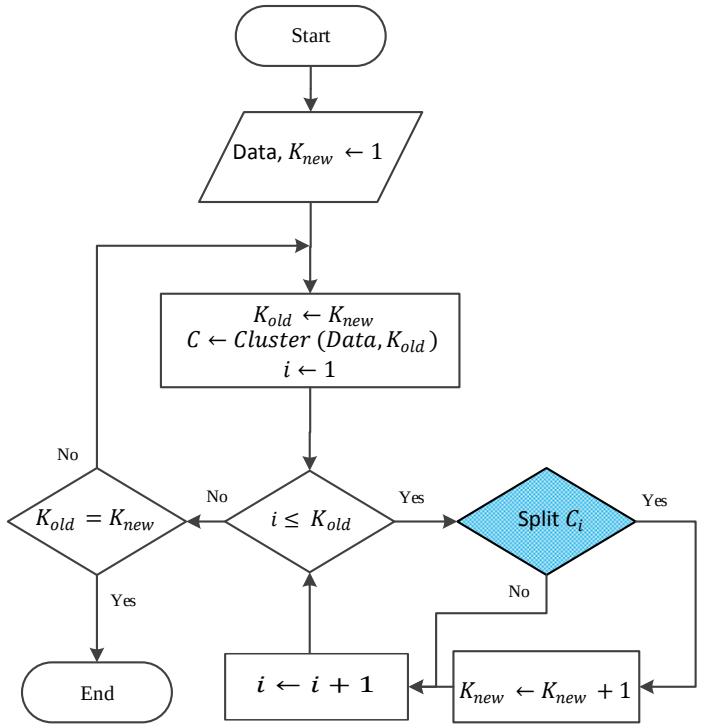


Figure 4.6: Hierarchical clustering with data splitting criterion.

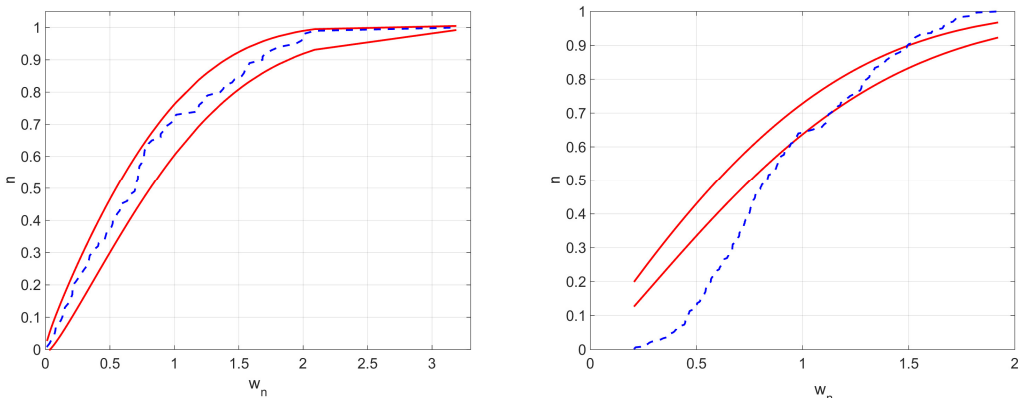
test can be used for both G-means and Dip-means clusterings and replace AD and Dip test. Figure 4.7 shows how this splitting criterion works. The solid lines are pre-calculated boundaries of sorted elements t_i ($\overline{S(t_i, \alpha)}$ and $\underline{S(t_i, \alpha)}$ in (4.5)). The dashed line shows the sorted version of the absolute value of the data for two clusters.

As it can be shown in Figure 4.7 (b), Sigtest suggests to split data with an ecdf which is not a sample of Dcdf. While in Figure 4.7 (a), it suggest that the ecdf is a sample of Dcdf and the related cluster will not be split.

Detailed examples are provided in the simulations section.

4.3.2 Sigtest in Partitional Clustering

If the number of clusters is known to be K, partitional clustering method minimizes a given clustering criterion by iteratively relocating data points between K clusters until



(a) (b)

Figure 4.7: Sigtest in hierarchical clustering: (a) H_0 holds (no split), and (b) H_1 holds (split).

a (locally) optimal partition is attained [71].

If the number of clusters is not known, it needs to be estimated using a proper statistical test. For a considered range of $K \in [K_{min}, K_{max}]$, ecdf of the data and the Dcdf of a model will be compared for:

- H_0: The observed data (ecdf) is a sample of the model with Dcdf of K clusters.
- H_1: The observed data (ecdf) is not a sample of the model with Dcdf K clusters.

Starting from K_{min}, the statistical test increases the value of K until H_0 is satisfied. Figure 4.8 shows a partitional clustering method based on Gaussian mixture models (GMM), where Expectation Maximization (EM) is employed to estimate the parameters of the mixture model (center of clusters μ, covariance matrices Σ and components mixing factors π).

The shaded block shows the statistical testing step. This test in PG-means clustering is the KS test (briefly described in Subsection 2.2.1). We propose to replace KS with Sigtest and show the advantageous in the simulation section. In the case of Gaussian

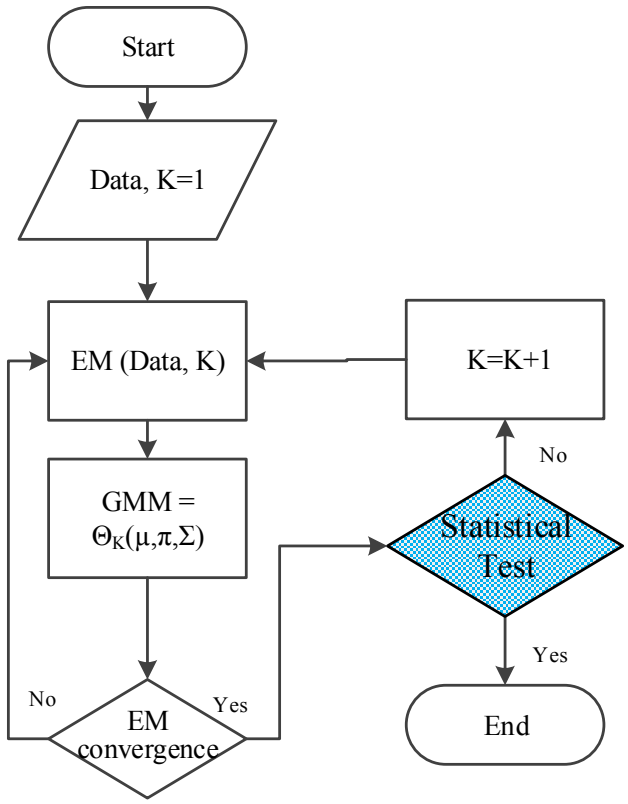


Figure 4.8: General procedure of partitional clustering with order selection.

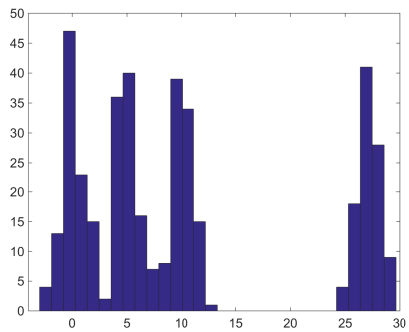
mixture models, the desired $F_a(z)$ used in (4.7) is in the form of [72] [2]:

$$F_a(z) = \sum_{j=1}^{K} \pi_j F_{aj}(z) \tag{4.10}$$

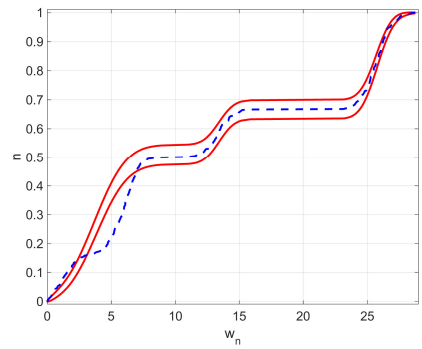
where $F_{aj}(z)$ is the Gaussian cdf of the jth component, where provided in (4.7) and π_j is the mixing factor of that component in the mixture.

Figure 4.9 shows an example of using Sigtest for such verification. In this example, the true number of clusters is 4 and Figure 4.9a shows the histogram of the data (y).

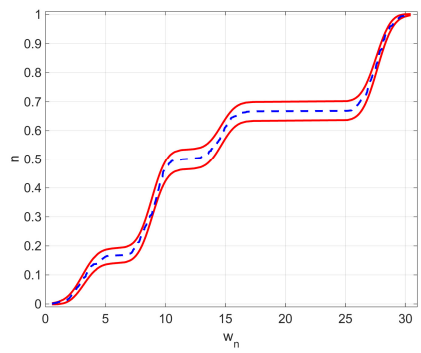
[2]Details for calculating $F_{aj}(z)$ is provided in C.



(a) Histogram of projected data belong to a mixture of four Gaussians.



(b) Ecdf of data (dashed line) is not a sample of Dcdf (solid line) for $K = 3$.



(c) Ecdf of data (dashed line) is a sample of Dcdf (solid line) for $K = 4$.

Figure 4.9: Sigtest for model verification in Gaussian mixture models.

If K is considered to be 3, the upper bound and lower bounds of Sigtest are solid lines in Figure 4.9b. The ecdf in this case, however, is the dashed line and as the figure shows it falls out of the boundaries. The method therefore increases the value of K to 4. Figure 4.9c shows the boundaries for $K = 4$. As the figure shows, in this case the ecdf completely fits within the boundaries. Sigtest stops and the estimated number of clusters is 4. [3]

[3] The optimum choice of alpha and T has improved the result of Sigtest compare to [73] and [74].

4.4 Optimum vocabulary size in bag of visual words using Sigtest

Image classification using bag of visual words is constructed based on transforming a 2D image and representing that into a 1D histograms. This approach has the following main steps: 1) feature extraction (methods such as scale-invariant feature transform (SIFT) [75], Dense SIFT [76] and Histograms of Oriented Gradients (HOG) [77]) 2) feature quantization to build a visual vocabulary of size K, 3) training and classification using Support Vector Machines (SVM) or similar classifiers.

SIFT is an image descriptor for image matching and image recognition. It is computed from image intensities around the key point locations in image. SIFT is invariant to scaling transformations, rotations and translations in image domain and it is robust to moderate changes in illumination. Dense SIFT is a similar approach where SIFT descriptor is computed over dense grids in the image domain. HOG descriptor splits image into overlapped cells and computes histograms of gradients for each cell. Unlike SIFT, this method is not rotation invariant, however it is normalized with respect to image contrast.

The concentration of our work is on the second step using SIFT feature of image. Conventionally for this step a fixed size for visual vocabulary is considered. For example, for most cases, they start with K = 500 words. This number is then given to k-means for calculation of those 500 visual words (centers). The SIFT of each image is then compared to these visual words and the histogram of the membership of these SIFTs to the centers is then provided. Figure 4.10 shows an example of a test image from Caltech101 data set (Figure 4.10 (a)) along with its histogram (Figure 4.10 (c)). Figure 4.10 (b) is a general example to show the middle step for feature quantization.

Note that prefixing the vocabulary size, K, is providing a suboptimal solution for the problem of bag of visual words. Motivated by this fact, we propose to find an

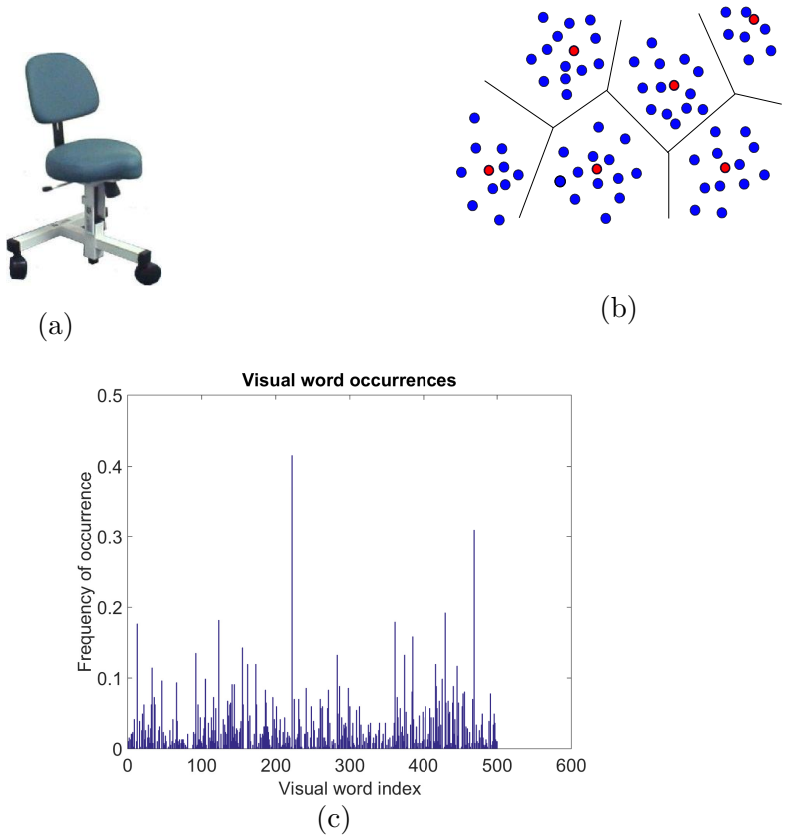


Figure 4.10: (a) Test image from Caltech101, (b) general example of quantizing features (blue dots) with their nearest centers (red dots) and (c) representing them as a histogram over the visual words.

adaptive number of vocabulary size K by using the hierarchical G-means-Sigtest. This preprocessing algorithm in step 2 can benefit the next step and the overall answer of the classification.

In the following next section, benefits of such preprocessing is elaborated for a set of image data sets (Subsection 4.5.3).

4.5 Experimental Results

We use real data sets with the following characteristics :

Table 4.1: Benchmark datasets.

Data set	Number of samples	Dimension of data	Number of clusters
Iris	150	4	3
Pendigits	3498	16	10
15 objects category	45312	128	15
4 objects category	10008	128	4
Breast cancer	699	9	2
Leukemia	70	40	3
Optical digits	1797	64	10
Seed	210	7	3
Wave Form	5000	21	3
Human activity	2947	561	6
MNIST	4000	784	10
COIL20	1440	1024	20

Where 15 objects and 4 objects categories are from test cases in Caltech101 data sets [78]; MNIST and COIL20, from [79] and [80]; Rest of data sets are obtained from UCI repository [81].

In addition to the benchmark data sets, the synthetic data is a set of Gaussian clusters with $\sigma = 1$ and 100 samples in each cluster (Table 4.2). The centers of clusters are chosen randomly inside a hypercube with each side of 20σ.

Table 4.2: Synthetic data sets.

Data set	Number of samples	Dimension of data	Number of clusters
S1	1000	4	10
S2	2000	10	20
S3	3000	16	30
S4	4000	32	40

Adjusted Rand Index (ARI) and Variation of Information (VI) are used for comparison and to measure the quality of clustering. A more efficient clustering has a smaller value of VI and larger value of ARI [82], [83]. In the following table, N/A shows that clustering method was unable to converge to a solution.

4.5.1 Hierarchical Clustering

The comparison results between G-means and G-means-Sigtest is shown in Table 4.3. G-means-Sigtest is the G-means where the AD splitting criterion is replaced by Sigtest. Each number in the table is in the form of $E[\cdot] \pm std[\cdot]$ which shows the mean and standard deviation of the estimated values based on the averaged results over 20 simulation.

As the table shows, G-means-Sigtest compared with the G-means has a better estimation of the number of clusters. For Gmeans-Sigtest also we have smaller VI and larger ARI indexes (better performance).

We denote Dip-means-Sigtest in which Dip statistical test is replaced by Sigtest. Table 4.3 shows the result for Dip-means and Dip-means-Sigtest for both synthetic and real data sets. As the table shows, replacing Dip test with Sigtest has significantly improved the result of clustering in terms of estimated number of clusters, ARI and VI indexes.

4.5.2 Partitional clustering

The comparison result between PG-means and PG-means-Sigtest where the statistical test KS is replaced by Sigtest is shown in Table 4.3. Each number in the table is in the form of $E[\cdot] \pm std[\cdot]$ which shows the mean and standard deviation values based on the averaged results over 20 simulation.

As the table shows, PG-means-Sigtest compared with the PG-means has a better estimation of the number of clusters. For PGmeans-Sigtest also we have smaller VI and larger ARI indexes.

Table 4.3 also shows a comparison between MACE-means clustering and the above mentioned methods. As the table shows, MACE-means has its best performance on Gaussian data sets and has difficulties in clustering real and non-Gaussian data sets.

Table 4.3: G-means and G-means-Sigtest

Data set	G-means	G-means-Sigtest	Dip-means	Dip-means-Sigtest	PG-means	PG-means-Sigtest	MACE-means
Iris	3±0	3±0	2±0	3±0	2±0	4±0	5±0
VI	0.52±0	0.52±0	0.60±0	0.67±0.13	0.46±0	0.413±0	0.133±0.326
ARI	0.56±0	0.56±0	0.53±0	0.56±0.11	0.56±0	0.841±0	0.101±0.248
Optical digits	18.50±1.32	6.36±0.77	1±0	16±0	N/A	N/A	1±0
VI	1.76±0.07	1.29±0.05	2.302±0	1.22±0			0.115±0.514
ARI	0.27±0.03	0.63±0.02	0±0	0.62±0			0±0
Leukemia	4±0	3±0	2±0	3±0	4±0	6±0	2±0
VI	0.53±0	0.36±0	0.76±0	0.30±0	1.21±0	1.30±0	0.095±0.261
ARI	0.75±0	0.84±0	0.52±0	0.88±0	0.003±0	0.001±0	0.065±0.179
Seed	2±0	2±0	1±0	3±0	N/A	N/A	3±0
VI	0.81±0	0.81±0	1.0986±0	0.66±0			0.111± 0.273
ARI	0.47±0	0.47±0	0±0	0.71±0			0.1194± 0.292
Pendigits	27.96±2.92	17.28±0.75	7±0	10.2±0.44	11±0	10±0	1±0
VI	1.63±0.06	1.55±0.02	1.586±0	1.401±0.00	1.45±0	1.38±0	0.115±0.514
ARI	0.46±0.02	0.50±0	0.34±0	0.57±0.00	0.47±50	0.48±0	0±0
COIL20	41.40±2.95	18.90±2.23	3±0	44.8±0.83	N/A	N/A	1±0
VI	1.56 ±0.06	0.73 ± 0.07	2.73±0	1.40±0			2.99±0
ARI	0.43±0.02	0.64 ±0.02	0.07±0	0.54±0			0±0
wave form	9±0	5±0	2±0	5±0	6±0	4±0	2±0
VI	1.76 ±0	1.49 ± 0	1.106±0	1.427±0	1.39±0	1.38±0	0.184±0.451
ARI	0.23±0	0.25±0	0.371±0	0.291±	0.24±50	0.29±0	0.061±0.151
Human Activity	29.20±3.27	4±0	3±0	21±1.4	N/A	N/A	1±0
VI	2.13 ±0.08	1.02 ± 0	0.770±0	1.59±0.01			1.78±0
ARI	0.32±0.02	0.53±0	0.49±0	0.37±0			0±0
Breast cancer	52±8.14	2±0	N/A	N/A	6±0	6±0	2±0
VI	2.38 ±0.20	0.32 ± 0			0.973±0	0.928±0	0.081±0.162
ARI	0.20±0.02	0.84±0			0.500±0	0.524±0	0.211±0.423
MNIST	23.80±0	10.20±1.09	1±0	1±0	N/A	N/A	1±0
VI	2.63 ±0.03	1.69 ± 0.02	2.299±0	2.299±0			0.115±0
ARI	0.28±0	0.45±0	0±0	0±0			0±0
S1	10.5±0.60	10.15±0.48	8.65±2.32	9.55±1.09	10±1.16	9.95±0.60	9.95±0.223
VI	0.09±0.07	0.07±0.08	0.08±0.09	0.05±0.06	0.08±0.09	0.05±0.06	0.003±0.015
ARI	0.95±0.02	0.97±0.05	0.81±0.31	0.92±0.15	0.954±0.05	0.972±0.04	0.049±0.219
S2	20.1±0.44	20.05±0.22	17.85±3.9	20.45±0.51	19.95±0.94	20.25±0.71	20±0
VI	0.003±0	0.001±0	0.27±0.52	0.01±0.01	0.08±0.04	0.02±0.02	0±0
ARI	0.99±0.0	0.99±0	0.82±0.32	0.99±0	0.94±0.02	0.98±0.01	1±0
S3	31.35±1.49	30.9±0.91	18±8.20	29.65±1.81	27.7±1.08	29.15±1.34	30.50 ± 0.527
VI	0.02±0.02	0.01±0.01	1.047±0.74	0.034±0.14	0.138±0.04	0.134±0.04	0.003±0.003
ARI	0.98±0.01	0.99±0	0.42±0.34	0.97±0.12	0.89±0.03	0.92±0.02	0.016 ± 0.128
S4	44.4±1.75	43.85±1.81	13.1±4.48	39.95±1.05	35.45±1.35	36.9±1.51	41±1.25
VI	0.07±0.02	0.06±0.03	2.07±0.32	0.01±0.04	0.24±0.09	0.15±0.08	0.09±0.01
ARI	0.97±0.01	0.97±0.01	0.096±0.02	0.98±0.04	0.75±0.09	0.84±0.10	0.88±0.11

4.5.3 Adaptive vocabulary size in bag of visual words

As discussed in Subsection 4.4, the size of vocabulary is a pre-assumed fixed number in the range of $K = [500, 1000]$ in most cases. Here, we suggest using G-means and G-means-Sigtest to adaptively estimate the size of the vocabulary for 15 objects category and 4 objects category data sets, instead of the traditional fixed value of $K = 500$. This two methods are chosen due to their high accuracy and fast convergence in large data sets.

Figure 4.11 illustrates the accuracy of classification for 15 class of objects based on different number of clusters or vocabulary sizes (blue line). In this figure, green and red dashed lines show the estimated number of clusters using G-means-Sigtest and original G-means respectively. As figure shows, G-means-Sigtest chooses 593 for the size of vocabulary that results in the highest accuracy of classification. Fixed value of K=500 has less accuracy of classification and G-means chooses 1184 as the vocabulary size with much less accuracy in classification.

Figure 4.12 shows similar results for BOVW experiment on 4 class of objects. As the figure shows, in this case both Gmeans-Sigtest and Gmeans choose less number of vocabulary size than 500. While Gmeans chooses 218, Gmeans-Sigstest chooses much smaller value of 74 with the highest accuracy in classification for different vocabulary sizes. Sigtest results in the smallest possible number of clusters along with the highest classification rate with less time complexity compared to when K is pre set to 500.

4.6 Conclusions

In this Chapter, we proposed Signature Testing (Sigtest) as a new statistical testing method for estimating the number of clusters in both hierarchical and partitioning clustering methods. In addition we propose using Sigtest in image classification using bag of visual words for adaptive choice of the size of visual vocabulary. Simulation results

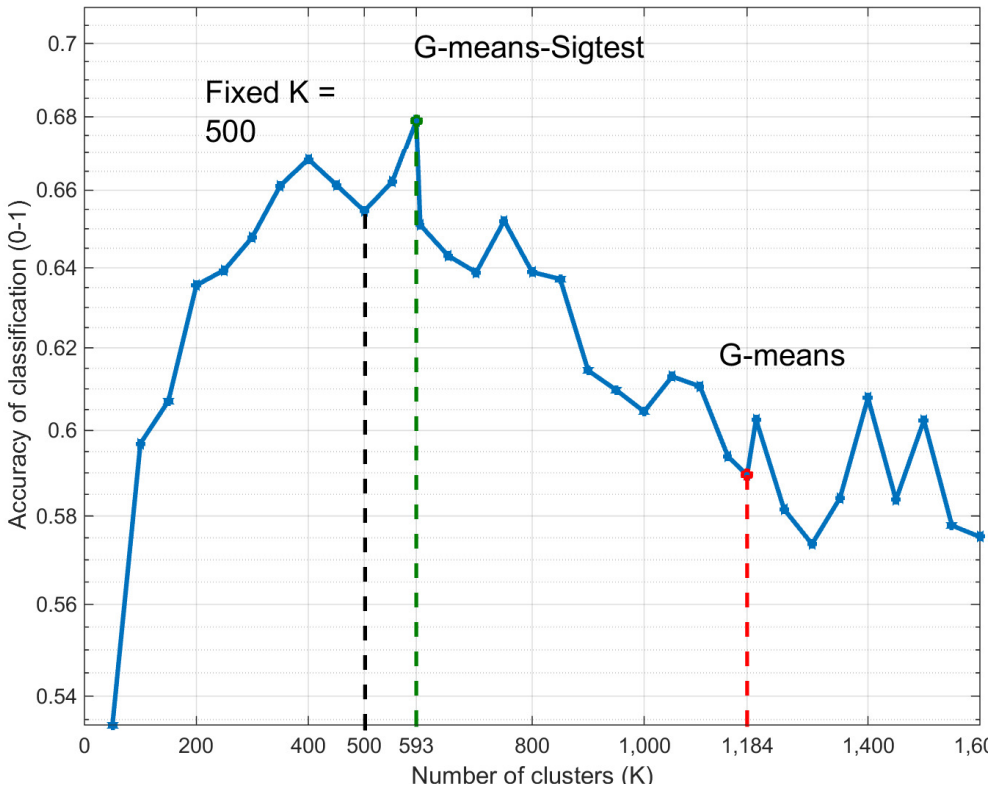


Figure 4.11: Accuracy of SVM classifier for different number of clusters (size of visual vocabulary) for 15 objects category from Caltech101 data set. Black dashed shows the accuracy at the location of $K = 500$ (fixed size assumption), green dashed shows the chosen value K=593 by G-means-Sigtest, and red dashed line shows the accuracy of G-means for estimated $K = 1184$.

confirm advantageous of using Sigtest as the statistical test in clustering in terms of more accurate choice of number of clusters, and better values for ARI and VI. The results also show that Sigtest improves the accuracy of image classification as well as reducing the time complexity in bag of visual words.

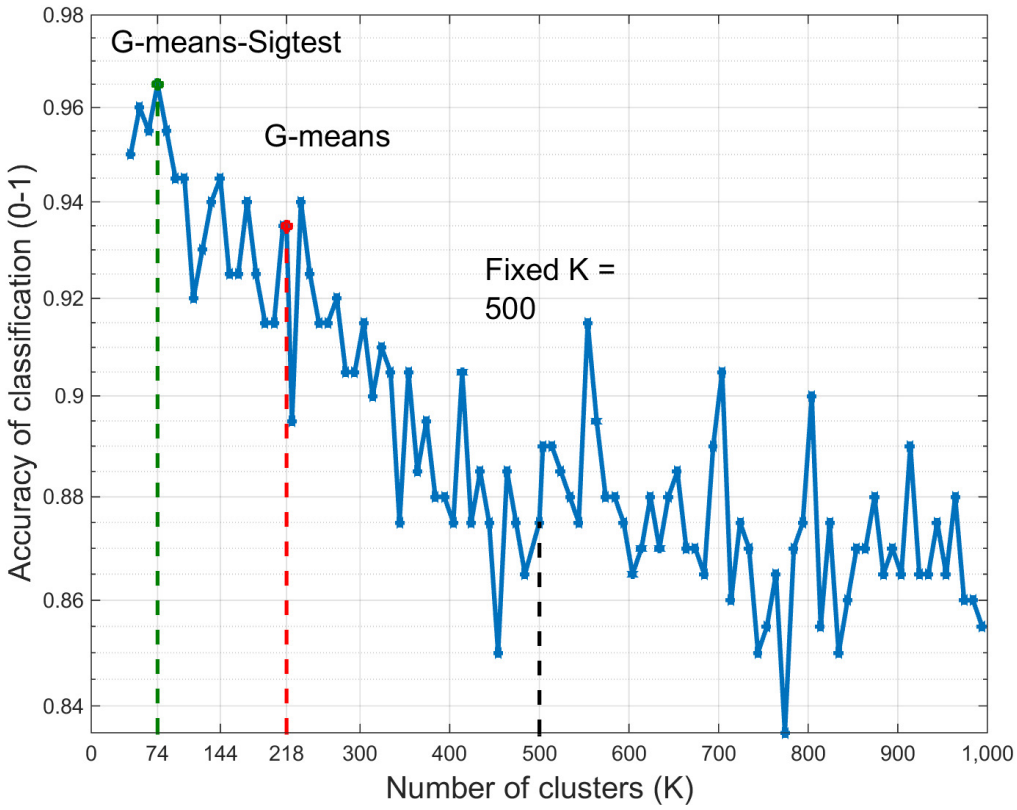


Figure 4.12: Accuracy of SVM classifier for different number of clusters (size of visual vocabulary) for 4 objects category from Caltech101 data set. Black dashed line shows the accuracy at the location $K = 500$ (fixed size assumption), green dashed line shows the chosen value K= 74 by G-means-Sigtest, and red dashed line shows the accuracy of G-means for estimated $K = 218$.

Chapter 5

Minimum Pathways in Arbitrary Shaped Clustering (minPAS clustering)

In this chapter we consider data clustering for arbitrary shaped clusters. Briefly, this class of clusters do not follow a simple and regular known distributions such as Gaussian or Log-normal. In general, shape of an arbitrary cluster cannot be easily modeled by a single mathematically available distributions. Therefore, majority of the model based clustering approaches are unable to cluster arbitrary shaped clusters with a reasonable level of error.

5.1 Data assumptions

The main assumptions in arbitrary shaped clustering are as follows: data samples represent clusters with arbitrary shapes, arbitrary densities and arbitrary sizes. Figure 5.1 is an example of arbitrary shaped clusters which shows a ring cluster with another cluster inside it. In this Figure, the center cluster could be approximated with a Gaussian

distribution while the ring cluster cannot be modeled with simple distributions.

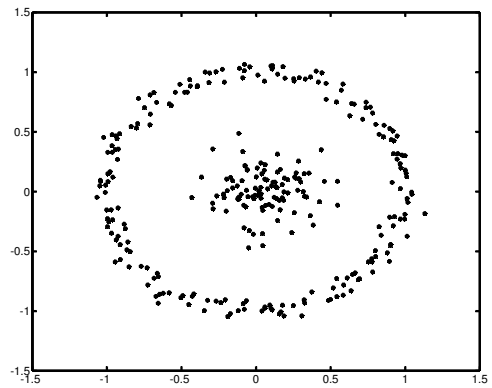


Figure 5.1: Two centered clusters.

consequently, clustering methods which are limited to a specific class of distributions will not be able to recover these clusters properly.

5.1.1 Data Skeleton Using Minimum Spanning Tree

Let $X = [x_1, x_2, \cdots, x_N]^T$ be a vector of N samples, where $x_i \in R^D$, and D shows the data dimensionality. $s_{N\times N}$ is a symmetric dissimilarity matrix for the samples, where $s(i,j) = d_{x_i x_j}$ ($d_{x_i x_j} \in R$) is a weight to measure the distance between x_i and x_j.

We define $G(X,E)$ as an undirected graph, where $E = \{e_{ij} : e(x_i, x_j), (i,j) \in [1, \cdots, N]\}$ is a vector of undirected edges between the samples in X. The weight of edge e_{ij} is denoted by $d_{x_i x_j}$ and can be calculated as follows:

$$d_{x_i x_j} = \|x_i - x_j\|_2^2 \tag{5.1}$$

We let $E^{'}$ be an acyclic subset of E ($E^{'} \subset E$) that connects all of the samples, and its size is $|E^{'}| = N - 1$. Therefore, the overall weight of edges, $W_{E'}$, can be given as follows:

$$W_{E'} = \sum_{E'} d_{x_i x_j} \tag{5.2}$$

Then, minimum spanning tree (MST) $T(X, E^*)$ is also an acyclic subgraph of G that passes through all of the samples in X and has the following set of edges [84], [85]:

$$E^* = \arg\min_{E'} \sum_{E' \subset E} W_{E'} \tag{5.3}$$

Therefore, among all of the possible trees in X, MST has the minimum overall weight W_{min}.

The MST of samples in Figure 5.1 is shown in Figure 5.2. As the figure shows, any two samples x_i and x_j have exactly one edge e_{ij} between them. The MST of data can be constructed by any of proposed algorithms in [86] or [87].

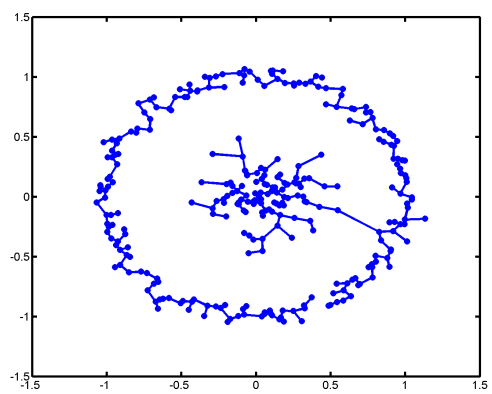


Figure 5.2: Minimum spanning tree of 300 samples.

In the following section, we use MST of data as a robust and unique structure to define the minimum pathways between samples in the tree.

5.1.2 Minimum Pathways in Arbitrary Shaped Data

In this section, we employ the minimum spanning tree of the data samples for defining a unique pathway between members of arbitrary shaped clusters. These pathways will be used as relational measures to evaluate the dependency of each sample to an exemplar candidate of each cluster.

The notion of MST requires that any pair of x_i and x_j in $T(X, E^*)$ have a unique shortest path between them. For an arbitrary shaped data set X, we denote $T_p(X_{x_i x_j}, E^*_{x_i x_j})$ as the minimum pathway between x_i and x_j, which is a subtree of T ($T_p \subset T$). $X_{x_i x_j} = [x_i, x_{i+1}, \cdots, x_j]$ ($X_{x_i x_j} \subset X$) is a sequence of samples that construct the minimum path between x_i and x_j. $E^*_{x_i x_j} = \{e_{kk+1} : e(x_k, x_{k+1}), i \leq k < j\}$ represents the set of edges for each consecutive samples in $X^{'}$.

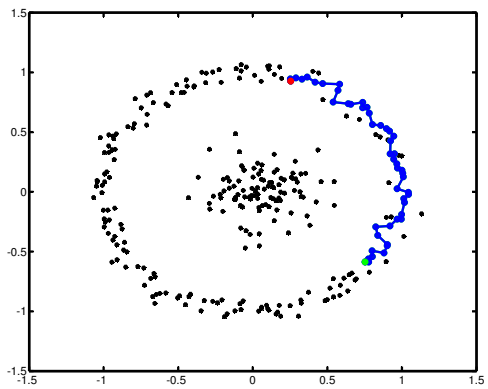
We let $d^p_{x_i x_j}$ be the distance weights of edges in E^* between x_i and x_j:

$$d^p_{x_i x_j} = [d_{x_i x_j}(1), \cdots, d_{x_i x_j}(k)] \tag{5.4}$$

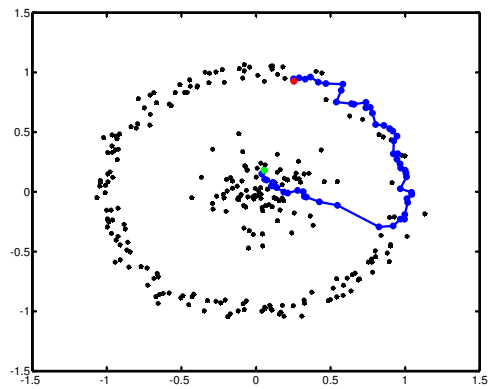
In other words, $d^p_{xix_j}$ includes all of the step sizes that are required to traverse from x_i to x_j and vice versa.

Figure 5.3a shows an example of a minimum pathway in arbitrary shaped data (minPAS). In this figure, minPAS (the blue subtree) has connected two samples x_i (red dot) and x_j (green dot) from the same cluster. Connected samples are members of $X_{x_i x_j}$ and the step sizes are members of $d^p_{xix_j}$. Figure 5.3b shows another minPAS for the same data with two samples from different clusters.

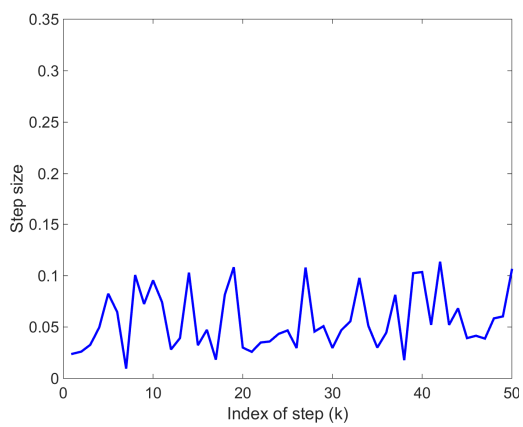
As these figures show, the step sizes in each pathway can be used for learning the level of similarity between samples in arbitrary shaped clusters.



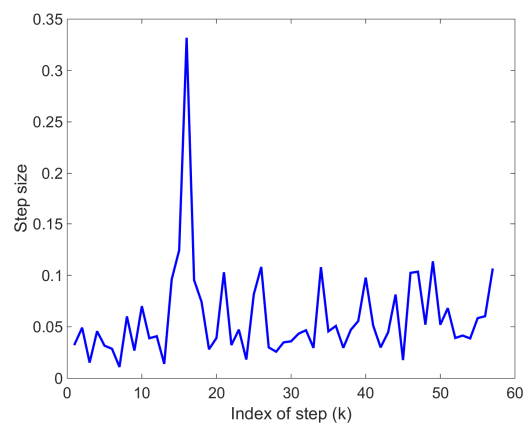
(a) minPAS between x_i and x_j when samples belong to the same clusters.



(b) minPAS between x_i and x_j when samples belong to different clusters.



(c) Step sizes in minPAS between x_i and x_j. Samples belong to the same cluster.



(d) Step sizes in minPAS between x_i and x_j. Samples belong to different clusters.

Figure 5.3: Minimum pathways between two samples from the same and different clusters.

5.1.3 Membership Score

Clustering methods with assumption of having a specific distribution for clusters cannot be applied directly on arbitrary shaped data. This is mainly a result of having distance based similarity measures in those clustering methods. As a result, simple similarity measures like euclidean distance cannot work well on clusters with complex geometries. For example, euclidean distances between the samples x_i and x_js in Figure 5.3a and Figure 5.3b suggest more similarity between the samples in Figure 5.3b compared to Figure 5.3a. Therefore, this can lead to a wrong sample membership in clusters. For example, clustering methods like K-means with assumption of spherical clusters are not able to cluster these samples correctly.

Our motivation in this Subsection is defining an efficient membership score which can assign arbitrary shaped samples to their related exemplars without imposing any predefined geometry on the data (for example, Gaussian assumption imposes spherical clusters). Using (5.4), we let $Score(x_i, x_j)$ be the dissimilarity score between the samples x_i and x_j:

$$Score(x_i, x_j) = \max_{1 \leq l \leq k} d^p_{x_i x_j}(l) \tag{5.5}$$

Figure 5.3c shows required step sizes for traversing between x_i and x_j in Figure 5.3a, where dissimilarity $Score(x_i, x_j) = 0.11$. In a similar example, Figure 5.3d shows required step sizes for traversing between x_i and x_j in Figure 5.3b, where dissimilarity $Score(x_i, x_j) = 0.33$

As the figures show, this simple dissimilarity score can be employed as a membership score to assign samples to their related exemplars. For example, two samples with larger jumps in the pathway between them are less likely to be from the same cluster compared with two other samples that have smaller jumps in their pathway. However, this is only one of the possible membership scores based on the minPAS between two samples and

it can be extended for designing similar scores.

5.2 minimum Pathway in Arbitrary Shaped clustering (minPAS)

In the previous section, we explained details of extracting a unique tree structure for arbitrary shaped data. We showed that the minimum pathways between samples can be used as a dissimilarity measure between samples.

In the following, we show steps of our proposed clustering method, denoted by minimum Pathways in Arbitrary Shaped clustering (minPAS clustering).

minPAS clustering groups samples in one cluster and then iteratively increases the number of clusters until a stopping criterion is satisfied.

Lets C_i be the first cluster exemplar chosen from the data samples with the following conditions: 1) C_i is not a leaf in the MST, 2) C_i has the minimum averaged distance with its nearest neighbors in the MST. It follows that for any C_i, the dissimilarity $Score(C_i, X)$ between samples and the exemplar can be categorized into two main regions with the maximum and minimum dissimilarities. The region of minimum dissimilarity, denoted by R_i^{min}, includes all of the members of exemplar C_i. The region with maximum dissimilarity are samples that cannot be assigned to the exemplar. At each iteration of the algorithm, R^{max} includes a subset of the region with maximum dissimilarity. We iteratively subtract R^{max}s from the range of available samples and assign the final remaining samples to R_i^{min}.

We let Z be an intermediate set to track the available samples. The initial set of values in Z is X:

$$Z_{i,0} = \{x_1, x_2, \cdots, x_N\} \tag{5.6}$$

I) ***Excluding non-members of*** C_i***:***

For a chosen C_i, the $R^{max}_{i,k+1}$ at the Kth step is defined as follows:

$$R^{max}_{i,k+1} = \{x | Score(C_i, x) = \max_{y \in Z_{i,k}} Score(C_i, y)\} \tag{5.7}$$

if the size of $R^{max}_{i,k+1}$ is greater than δ, where δ is the minimum possible number of members in a cluster (δ is a predefined parameter in majority of arbitrary shaped clustering methods):

$$|R^{max}_{i,k+1}| > \delta \tag{5.8}$$

then $R^{max}_{i,k+1}$ can be one or more potential cluster(s) that we are going to discover in the next steps. Therefore, we exclude it from $Z_{i,k}$ to reach to the final members of C_{i+1}:

$$Z_{i^*,k+1} = \{x | x \in Z_{i,k} \; and \; x \notin R^{max}_{i,k+1}\} \tag{5.9}$$

The above routine (I) will be repeated until $|R^{max}_{i,k+1}| \leq \delta$. Z_{i^*} will be members of the cluster with exemplar C_i:

$$R^{min}_i = Z^*_i \tag{5.10}$$

II) ***Stopping Criterion at Each C_i:***

We let R^s_i be the union set of all members of the previous exemplars:

$$R^s_i = \bigcup_{j=1}^{i} R^{min}_i \tag{5.11}$$

if the size of R^s_i is smaller than N (total number of samples):

$$|R^s_i| < N \tag{5.12}$$

This indicates that not all of the data set is labeled. Therefore, we continue searching

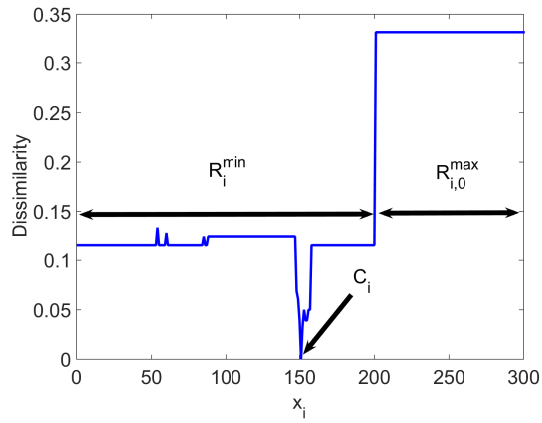


Figure 5.4: Dissimilarity Scores based on assumption of having $C_1 = x_{150}$.

for new clusters C_is and redefine the Z as follows:

$$Z_{i+1,0} = \{x | x \in X \; and \; x \notin R_i^s\} \tag{5.13}$$

Consequently, we choose a new exemplar $C_{i+1} \in Z$ (C_{i+1} should not be a leaf in the MST and it should have the minimum averaged distance with its neighbors in the MST) and calculate $Score(C_{i+1}, X)$. Next, for the updated Z and C_{i+1}, we start over from the procedure (I) (5.7) to the end except (5.12) is not true.

The dissimilarity scores of the exemplars can be used to find the label of each sample x_i:

$$label(i) = \arg\min_j (Score(C_j, x_i)) \tag{5.14}$$

For example, Figure 5.4 shows the dissimilarity score of each sample related to $C_i = x_{150}$ as the exemplar ($Score(C_i, X)$). Here, there is only one $R_{i,0}^{max}$ and excluding that from the samples leads to the region of R_i^{min}. Then, the new exemplar $C_{i+1} = x_{201}$ is chosen from $R_{i,0}^{max}$, which leads to $Score(C_{i+1}, X)$ in Figure 5.5. Since $R_i^{min} \cup R_{i+1}^{min}$ covers all of the N samples, the algorithm stops. Figure 5.6 shows the final scores for two possible exemplars. Consequently, using (5.14) the labels of clusters will be provided.

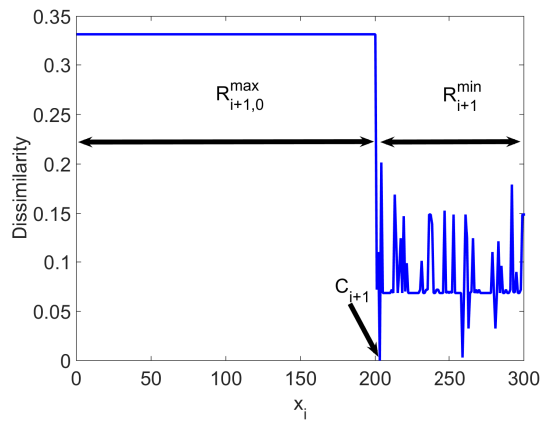


Figure 5.5: Dissimilarity Scores based on assumption of having $C_1 = x_{201}$.

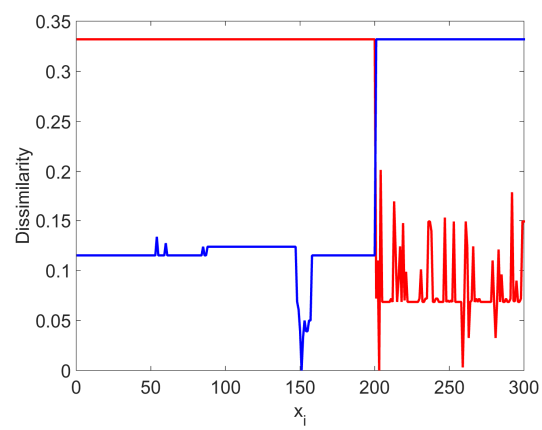


Figure 5.6: Sample Scores for C_i (blue line) and C_{i+1} (red line).

Algorithm 2 shows steps of the explained procedure in minPAS clustering.

Algorithm 2 minPAS clustering

Input: data X, δ.
Output: Cluster labels, number of clusters K.

```
T(X, E*) ← MST(X)
i ← 0
Z ← X
R^s ← ∅
while |R^s| < N do
    i ← i + 1
    Z ← X − R_s
    choose C_i ∈ Z
    calculate Score(C_i, X) using (5.5)
    calculate R^max using (5.7)
    while |R^max| > δ do
        Z ← Z − R^max
        calculate R^max using (5.7)
    end while
    R_i^min ← Z
    R^s ← ⋃_{j=1}^{i} R_j^min
end while
K ← i
for i = 1 → N do
    label(i) = arg min_j(Score(C_j, x_i))
end for
```

5.3 Computational Complexity Comparison

Computational complexity of minPAS clustering is $O(N^2)+O(KE\log(N))$, where $O(N^2)$ is related to the calculation of similarity matrix for N samples. $O(KE\log(N))$ is the required computational complexity in Dijkstra's algorithm for calculating the shortest pathways between samples, where E is the number of edges in the minimum spanning tree of data and K is the number of clusters. DBSCAN has a computational complexity of $O(N^2)$. Affinity Propagation has a computational complexity of $O(N^2l)$, where l is the number of iterations in the algorithm.

5.4 Experimental Results

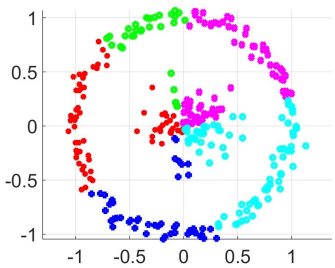
In this section we compare minPAS clustering with Data Spectroscopic clustering (DaSpec), Affinity Propagation (AP), DBSCAN and Agglomerative clustering using ward's method. In our experiments, the number of clusters is provided to ward's method clustering, while the rest of methods can estimate it independently.

The methods are compared based on arbitrary shaped datasets such as Atom and Chain link in [88], and our synthetic datasets as well as real data sets such as Iris, Seeds and Wine. The adjusted random index (ARI) and variation of information (VI) are two quality measures that we have employed in our analysis [83] [82].

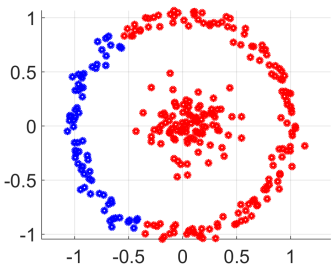
Figure 5.7 shows the comparison between methods on a ring cluster with a Gaussian cluster in the center. We have repeated the experiment for different clustering parameters (minPts in DBSCAN and δ in minPAS clustering) and only provided the distinct results. As the figure shows, AP and ward's method clustering cannot distinguish the clusters correctly. minPAS, DaSpec and DBSCAN are the only clustering methods that for a specific range of parameters can cluster the samples correctly. DBSCAN can only recognize the clusters for a small range of $10 \leq minPts \leq 20$, while the parameter of minPAS clustering is less sensitive and provides the correct results for a wider range of $5 \leq \delta \leq 198$. Table 5.1 shows the values of ARI and VI, where smaller VI and larger ARI show a better clustering result. As the table shows, minPAS clustering has the best ARI and VI values among the methods.

Table 5.1: Quality of clustering in ring data set.

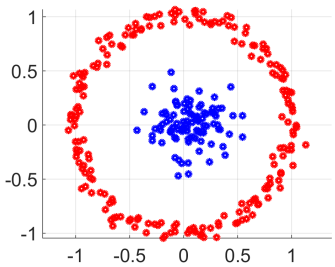
Method	Parameter	ARI	VI
minPAS	[5,198]	1	0
Affinity Propagation		-0.002	2.152
Ward's method		-0.021	0.956
DBSCAN	5	0.986	0.044
DBSCAN	[10,20]	1	0
DBSCAN	30	0	0.636
DaSpec		1	0



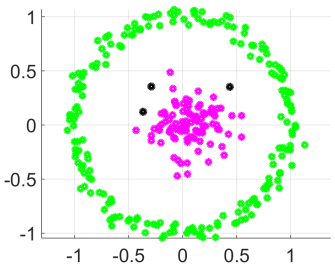
(a) Affinity Propagation.



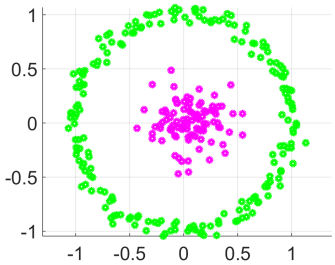
(b) Ward's method.



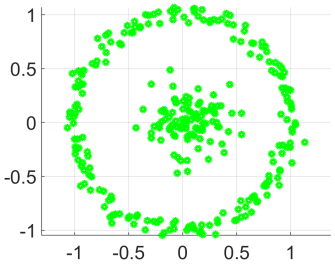
(c) minPAS for $5 \leq \delta \leq 198$.



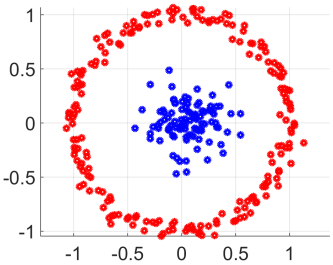
(d) DBSCAN for minPts = 5.



(e) DBSCAN for $10 \leq minPts \leq 20$.



(f) DBSCAN for minPts = 30.



(g) DaSpec.

Figure 5.7: Ring data set.

Figure 5.8 shows the result of clustering for a 3-dimensional spiral cluster with a Gaussian ball on top of it. As the figure shows, only minPAS was able to recognize the clusters with a significantly better VI and ARI values (Table 5.2), where changing the parameters didn't make any improvement in the results.

Table 5.2: Quality of clustering spiral and ball data set.

Method	Parameter	ARI	VI
minPAS		0.977	0.041
Affinity Propagation		-0.063	1.179
Ward's method		-0.078	0.896
DBSCAN		0	0.376
DaSpec		0	0.376

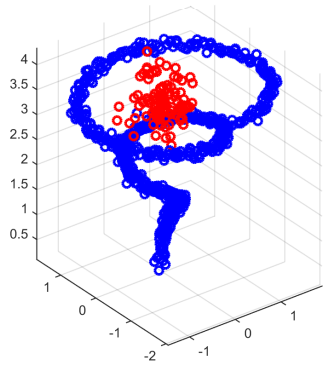
Figure 5.9 shows the comparison between methods on heart data set. minPAS provides an accurate clustering result for a wide range of parameter $5 \leq \delta \leq 48$. The second successful method is DBSCAN which in its best case is limited to minPts = 5 and it was not able to cluster all of the samples correctly. Table 5.3 shows better ARI and VI values for minPAS clustering on heart data set.

Table 5.3: Quality of clustering in heart data set.

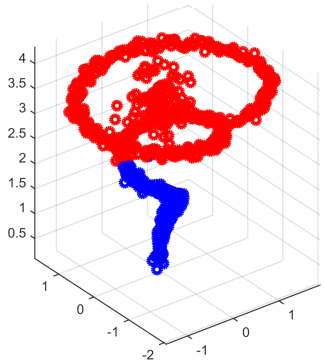
Method	Parameter	ARI	VI
minPAS	[5,48]	1	0
Affinity Propagation		0.120	1.523
Ward's method		0.189	1.294
DBSCAN	5	0.997	0.050
DBSCAN	10	0.994	0.018
DaSpec		0	1.053

Figure 5.10 shows the simulation results on Atom data set consists of two clusters, a small ball in the center of a spherical cluster. The best result of clustering belong to minPAS, where DaSpec, DBSCAN, AP and ward's method had the worst clustering results (Table 5.4).

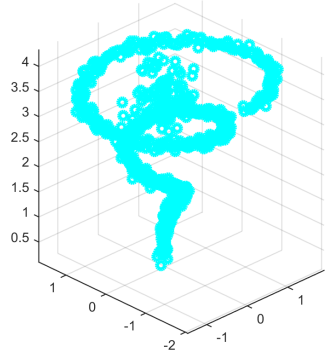
Figure 5.11 shows the simulation results of Chain link data set. Chain link consists of two linked rings in 3-dimensional space. As the figure shows, minPAS clustering



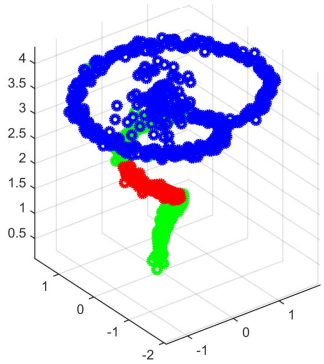
(a) minPAS.



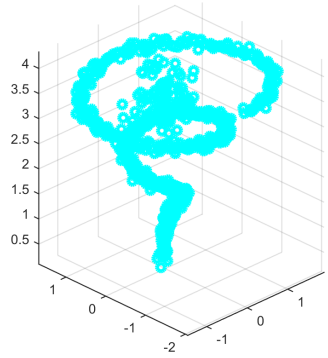
(b) Ward's method .



(c) DBSCAN.

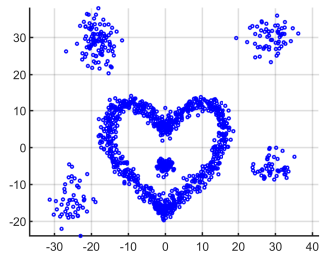


(d) Affinity Propagation.

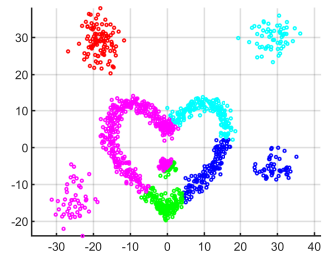


(e) DaSpec.

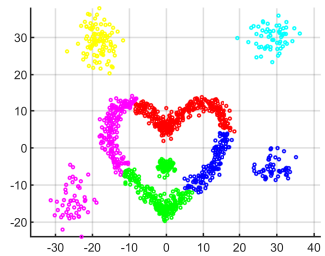
Figure 5.8: Spiral and ball data set.



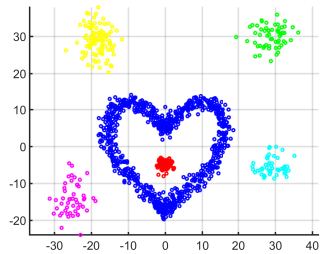
(a) DaSpec.



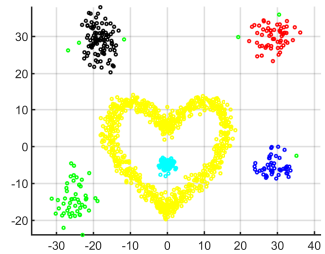
(b) Affinity Propagation.



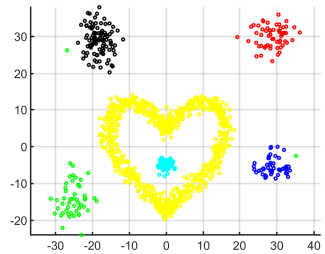
(c) Ward's method .



(d) minPAS for $5 \leq \delta \leq 48$.

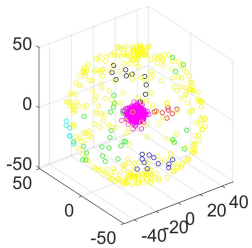


(e) DBSCAN for minPts = 5.

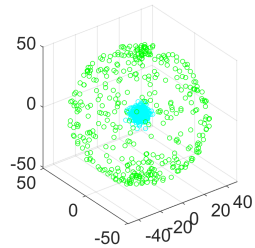


(f) DBSCAN for minPts = 10.

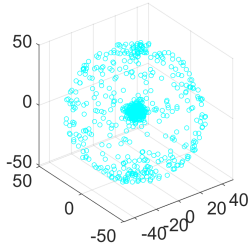
Figure 5.9: Heart.



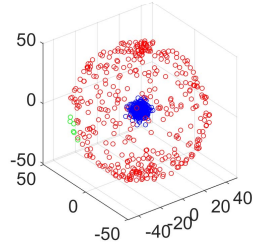
(a) DBSCAN for $4 \leq minPts \leq 8$.



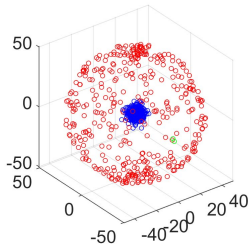
(b) DBSCAN for $9 \leq minPts \leq 184$.



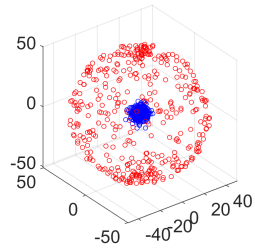
(c) DBSCAN for $minPts \geq 185$.



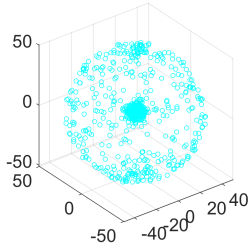
(d) minPAS for $4 \leq \delta \leq 140$.



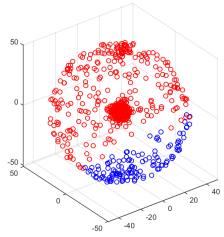
(e) minPAS for $141 \leq \delta \leq 283$.



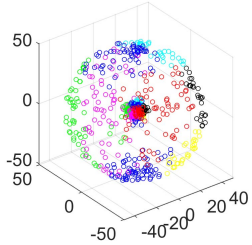
(f) minPAS for $284 \leq \delta \leq 398$.



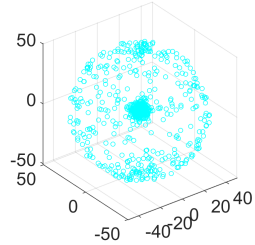
(g) minPAS for $\delta \geq 399$.



(h) Ward's method .



(i) Affinity Propagation.



(j) DaSpec.

Figure 5.10: Atom data set.

Table 5.4: Quality of clustering in Atom data set.

Method	Parameter	ARI	VI
minPAS	[4,140]	0.985	0.038
minPAS	[141,283]	0.995	0.0157
minPAS	[284,398]	1	0
DBSCAN	[4,8]	0.8554	0.353
DBSCAN	[9,184]	1	0
Affinity Propagation		0.260	1.118
Ward's method		0.2384	1.065
DaSpec		0	0.832

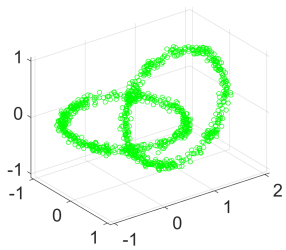
can recognize two clusters for the wide range of $5 \leq \delta \leq 498$, while in DBSCAN the same result is given for a much smaller range of parameter minPts, $5 \leq minPts \leq 153$. This fact shows that minPAS clustering is less sensitive to the choice of its parameter compared to DBSCAN. The ARI and VI values are shown in Table 5.5.

Table 5.5: Quality of clustering in chain link data set.

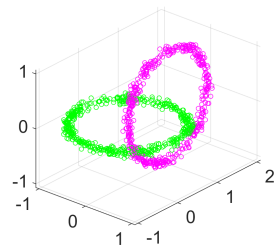
Method	Parameter	ARI	VI
minPAS	[5,498]	1	0
DBSCAN	[5,153]	1	0
Affinity Propagation		0.213	1.504
Ward's method		0.280	0.806
DaSpec		0	0.693

Figure 5.12 shows the simulation results of half moon data sets. Among the methods, minPAS has the most accurate result of clustering for a wide range of δ. DBSCAN is the second accurate method with more sensibility to the parameter minPts. Table 5.6 shows minPAS clustering has the best ARI and VI values among other methods.

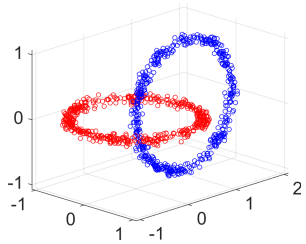
Table 5.7 shows the comparison between clustering methods on three real data sets Iris, Seeds and Wine, where the number of clusters in each of them is three. As the table shows, Ward's method has the best result in terms of ARI and VI. The second best method is DaSpec. minPAS and AP provide a better estimation of the number of clusters but have difficulties in partitioning the clusters.



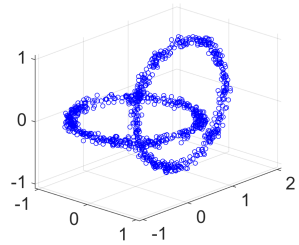
(a) DBSCAN for $minPts \geq 154$.



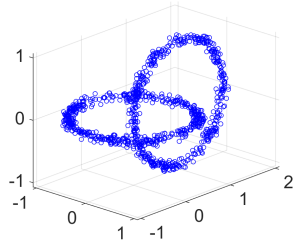
(b) DBSCAN for $5 \leq minPts \leq 153$.



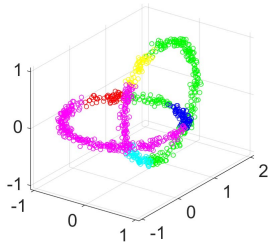
(c) minPAS for $5 \leq \delta \leq 498$.



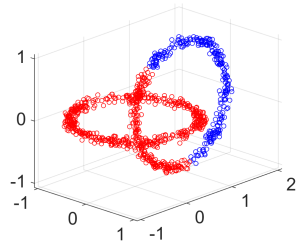
(d) minPAS for $\delta \geq 499$.



(e) DaSpec.

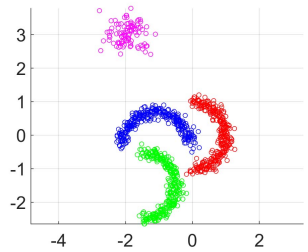


(f) Affinity Propagation.

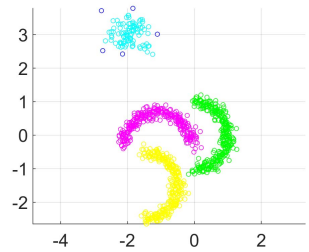


(g) Ward's method.

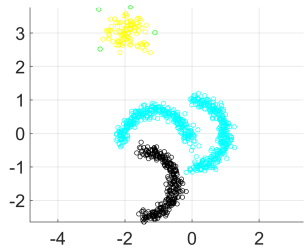
Figure 5.11: Chainlink data set.



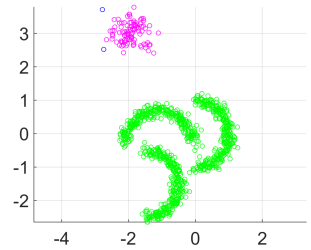
(a) minPAS for $5 \leq \delta \leq 98$.



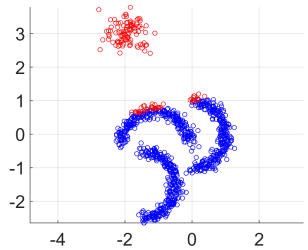
(b) DBSCAN for $minPts = 5$.



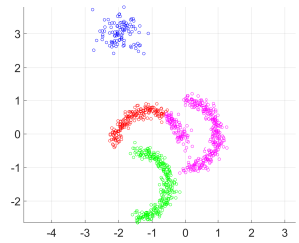
(c) DBSCAN for $minPts = 7$.



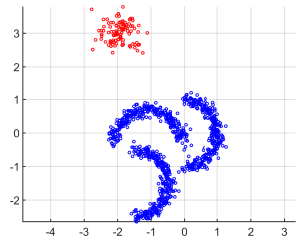
(d) DBSCAN for $minPts = 10$.



(e) Affinity Propagation.



(f) Ward's method



(g) DaSpec.

Figure 5.12: Half moon data set.

Table 5.6: Quality of clustering in half moon data set.

Method	Parameter	ARI	VI
minPAS	[5,98]	1	0
DBSCAN	5	0.599	0.783
DBSCAN	7	0.458	0.814
DBSCAN	10	0.031	1.380
Affinity Propagation		0.146	1.204
Ward's method		0.754	0.418
DaSpec		0.157	0.988

Table 5.7: Quality of clustering in real data sets.

Method	Iris	Seeds	Wine
minPAS			
Estimated K	3	3	1
ARI	0.563	0.002	0
VI	0.508	1.196	1.086
Affinity Propagation			
Estimated K	2	3	3
ARI	0.449	0.001	0.296
VI	0.845	1.253	1.391
Ward's method			
Estimated K	N/A	N/A	N/A
ARI	0.731	0.713	0.931
VI	0.499	0.587	0.198
DBSCAN			
Estimated K	2	3	1
ARI	0.568	0.001	0
VI	0.462	1.253	1.086
DaSpec			
Estimated K	2	2	3
ARI	0.568	0.459	0.371
VI	0.462	0.815	1.017

5.5 Conclusion

In this Chapter, we proposed minimum pathways in arbitrary shaped clustering (minPAS clustering) for clustering arbitrary shaped data. minPAS can estimate the number of clusters independently and measure the similarity of samples based on minimum pathways between them. The proposed method does not impose any assumption on the

distribution of data and can recognize arbitrary shaped clusters accurately. The simulation results on a wide range of arbitrary shaped clusters shows the superiority of minPAS clustering over similar methods in terms of accuracy and insensitivity to the algorithm parameter.

Chapter 6

Conclusions and Future Works

In this thesis, we studied three problems of data clustering in terms of estimating the number of clusters, the role of statistical tests as splitting criteria in partitional and hierarchical clustering and challenges in clustering and estimating the number of clusters in arbitrary shaped data.

We proposed minimum averaged central error (MACE)-means clustering as a new clustering method in Chapter 3. MACE-means clusters data and estimate the correct number of clusters (CNC) by minimizing the data reconstruction error. We derived the probabilistic bounds on unobservable data reconstruction error by using observable data error, and showed that minimizing the upper bound of this error leads to estimation of the CNC. Unlike majority of the clustering methods which have different objective functions for estimating the CNC and clustering data, MACE-means is constructed based on a unique objective function for both of them. The experimental results showed superiority of MACE-means in estimating the number of clusters over similar approaches as well as ARI and VI values. Note that MACE-means dependency on the assumption of having the same variance in clusters is a disadvantage of the method which should be addressed in the future work. MACE-means was proposed based on K-means due to its low computational complexity and simplicity. Nevertheless, MACE has the potential to

be used with other clustering methods. Another potential future work will be extending the MACE fundamentals to use with clustering methods with wider range of assumptions beyond the spherical Gaussian. As MACE-means has better performance on low dimensional data, working on scalability of the algorithm will be another direction for improving MACE-means.

We proposed Signature testing (Sigtest) as a statistical test in Chapter 4. Sigtest is motivated by this fact that sorted absolute value of observed data has much smaller variation compared to the original data and the resulted data will be represented in a much denser space. This dense region of the transformed data is used for designing a signature for any desired cumulative distribution function (cdf). We showed analytical steps for deriving upper bound and lower bound of the designed signature and used it for comparison with the empirical cumulative distribution function (ecdf) of the test data. We showed applications of Sigtest in hierarchical and partitional clustering algorithms where statistical tests in G-means, PG-means and Dip-means clustering algorithms were replaced by Sigtest. The simulation results showed that resulted clustering algorithms, denoted by G-means-Sigtest, PG-means-Sigtest and Dip-means-Sigtest have significantly improved the accuracy of clustering compared to the original methods. Another proposed application of Sigtest was adaptively estimating the size of vocabulary in bag of visual words (BOVW) problem. While majority of the BOVW methods use a prefixed vocabulary size, we show that using Sigtest can improve the accuracy of image classification and also decrease the time complexity of the algorithm. As fundamentals of Sigtest has been proposed for any considered distribution, applying Sigtest on non-Gaussian distributions is a possible interesting future work to extend the applications of Sigtest.

In Chapter 5 we proposed minimum Pathways in Arbitrary Shaped (minPAS) clustering for data sets with arbitrary distributions. MinPAS is constructed based on minimum spanning tree structure of samples. We showed that having the tree structure of data, each sample can be related to the exemplar of the cluster using a minimum pathway. As

a result, the similarity measure between samples will be highly affected by geometry of the data samples without relying on distribution assumptions. The experimental results showed that minPAS is more efficient than state of the art methods such as DBSCAN, DaSpec, and Affinity Propagation in terms of accuracy in clustering and having less sensitivity to the choice of minimum size of a cluster.

Currently, we select exemplars from the samples which are not leaves in the tree structure and have the minimum averaged distance with their neighbors. While the experimental results show this choice of exemplars is promising, one future work for minPAS clustering could be a robust approach for selecting the best exemplars. Another possible future work can be detecting the outliers and excluding them from minPAS clustering.

Data visualization could be another useful future work for deciding about the choice of clustering algorithm. Visualizing high dimensional data can give a better understanding about the distribution of clusters, and consequently makes it easier to select an appropriate algorithm for arbitrary or non-arbitrary shaped data.

Data stream clustering using the proposed clustering methods could be another direction for future work. While widely used stream clustering methods such as BIRCH [89] and C2ICM [90] need to have the number of clusters as a predefined value, the proposed methods could be employed for stream clustering and estimating the number of clusters simultaneously.

Appendix A

Average Central Error (Z_{Sm})

From equation (3.9), we have

$$Z_{Smi} = \frac{1}{n_i}\|I \begin{pmatrix} c^*_{x^1_{m_i}} \\ \vdots \\ c^*_{x^{n_i}_{m_i}} \end{pmatrix} - B_{mi} \begin{pmatrix} X^1_{m_i} \\ \vdots \\ X^{n_i}_{m_i} \end{pmatrix} \|_2^2, \tag{A.1}$$

Where I is an identity matrix and B_{mi} has the following format:

$$B_{mi} = \begin{pmatrix} \frac{1}{n_i} & \cdots & \frac{1}{n_i} \\ \vdots & \ddots & \vdots \\ \frac{1}{n_i} & \cdots & \frac{1}{n_i} \end{pmatrix}, \tag{A.2}$$

Therefore:

$$Z_{Smi} = \frac{1}{n_i}\|IC^*_{x_{mi}} - B_{mi}\, X_{m_i}\|_2^2, \tag{A.3}$$

where $X_{m_i} = C^*_{x_{mi}} + W_{mi}$ gives:

$$Z_{Smi} = \frac{1}{n_i}\|(I - B_{mi})C^*_{x_{mi}} - B_{mi}W_{mi}\|_2^2, \tag{A.4}$$

where $I - B_{mi} = A_{mi}$ and then Z_{Sm_i} will be given as follows:

$$Z_{Sm_i} = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}} - B_{mi}W_{mi}\|_2^2, \tag{A.5}$$

since $A^T_{mi}B_{mi} = 0$, therefore:

$$Z_{Sm_i} = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{1}{n_i}\|B_{mi}W_{mi}\|_2^2, \tag{A.6}$$

$$\|B_{mi}W_{mi}\|_2^2 = \frac{1}{n_i^2}\sum_{i=1}^{n_i} W_i^2 + \frac{2}{n_i^2}\sum_{j\neq k}^{n_i} W_jW_k, \tag{A.7}$$

assuming that all of the clusters have the same σ_w^2, the $E[Z_{Sm_i}]$ will be:

$$E[Z_{Sm_i}] = \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{1}{n_i}\sigma_w^2. \tag{A.8}$$

To find the variance of Z_{Sm_i}, we need to derive the variance and covariance terms in equation (A.6). The first term is a constant with zero variance. The remaining terms are related to (A.7) and have covariance and variances as follows:

$$var[\frac{1}{n_i^2}\sum_{i=1}^{n_i} W_i^2] = \frac{1}{n_i^4}(2n_i\sigma_w^4), \tag{A.9}$$

where the above equation is derived based on the definition of a zero mean ($E[w_i] = 0$) chi-squared random variable.

$$var[\frac{2}{n_i^2}\sum_{j\neq k}^{n_i} W_jW_k] = \frac{4}{n_i^4}\frac{n_i(n_i-1)}{2}\sigma_{w_j}^2\sigma_{w_k}^2, \tag{A.10}$$

by assuming the same σ_w^2 for all of the clusters, it can be simplified as follows:

$$var[\frac{2}{n_i^2}\sum_{j\neq k}^{n_i} W_jW_k] = \frac{4}{n_i^4}\frac{n_i(n_i-1)}{2}\sigma_w^4, \tag{A.11}$$

$$\frac{4}{n_i^4} cov(\frac{1}{n_i^2}\sum_{i=1}^{n_i} W_i^2, \sum_{j\neq k}^{n_i} W_j W_k) = 0, \tag{A.12}$$

where (A.12) is always equal to zero, and that is because of having i.i.d. data samples. In other words, both $E[W_i^2 W_j W_k]$ and $E[W_i^3 W_j]$ are zero. Finally, there will be the following statement for $var[Z_{Sm_i}]$:

$$var[Z_{Sm_i}] \quad = \quad \frac{2}{n_i^3}\sigma_w^4 + \frac{2(n_i-1)}{n_i^3}\sigma_w^4 = \frac{2}{n_i^2}\sigma_w^4. \tag{A.13}$$

Appendix B

Cluster Compactness Y_{Sm}

From equation (3.17), we have

$$Y_{Smi} = \tfrac{1}{n_i}\|I \begin{pmatrix} X_1 \\ \vdots \\ X_{n_i} \end{pmatrix} - B_{mi} \begin{pmatrix} X_1 \\ \vdots \\ X_{n_i} \end{pmatrix} \|_2^2, \tag{B.1}$$

Where B_{mi} is defined in A. We set $A_{mi} = I - B_{mi}$:

$$Y_{Smi} = \tfrac{1}{n_i}\| \begin{pmatrix} 1-\frac{1}{n_i} & \cdots & -\frac{1}{n_i} \\ \vdots & \ddots & \vdots \\ -\frac{1}{n_i} & \cdots & 1-\frac{1}{n_i} \end{pmatrix} \begin{pmatrix} X_1 \\ \vdots \\ X_{n_i} \end{pmatrix} \|_2^2 = \frac{1}{n_i}\|A_{mi}X_{mi}\|_2^2, \tag{B.2}$$

For each sample we have $X_i = C^*_{x_i} + W_i$, therefore:

$$\begin{aligned} Y_{Smi} &= \frac{1}{n_i}\|A_{mi}(C^*_{x_{mi}} + W_{mi})\|_2^2 \qquad \text{(B.3)} \\ &= \frac{1}{n_i}\|A_{mi}C^*_{x_{mi}}\|_2^2 + \frac{1}{n_i}\|A_{mi}W_{mi}\|_2^2 + \frac{1}{n_i}(W_{mi}^T A_{mi} C^*_{x_{mi}} + C^{*T}_{x_{mi}} A_{mi} W_{mi}), \end{aligned}$$

knowing that $A_{mi}^T A_{mi} = A_{mi}$ it follows:

$$\|A_{mi}W_{mi}\|_2^2 = \frac{n_i - 1}{n_i}\sum_{i=1}^{n_i} W_i^2 - \frac{2}{n_i}\sum_{j\neq k}^{n_i} W_j W_k, \tag{B.4}$$

and

$$C^{*T}_{x_{mi}} A_{mi} W_{mi} = \sum_{i=1}^{n_i} W_i c^*_{x_i} - \frac{1}{n_i} \sum_{i=1}^{n_i} W_i \sum_{j=1}^{n_i} c^*_{x_j}, \tag{B.5}$$

therefore, Y_{Smi} will be given as follows:

$$Y_{Smi} = \frac{1}{n_i} \|A_{mi} C^*_{x_{mi}}\|_2^2 - \frac{2}{n_i^2} \sum_{j \neq k}^{n_i} W_j W_k + \frac{n_i - 1}{n_i^2} \sum_{i=1}^{n_i} W_i^2 - \frac{2}{n_i^2} \sum_{i=1}^{n_i} W_i \sum_{j=1}^{n_i} c^*_{x_j} + \frac{2}{n_i} \sum_{i=1}^{n_i} W_i c^*_{x_i}, \tag{B.6}$$

Assuming that all of the clusters have the same σ_w^2, the $E[Y_{Smi}]$ will be given as follows:

$$E[Y_{Smi}] = \frac{1}{n_i} \|A_{mi} C^*_{x_{mi}}\|_2^2 + \frac{n_i - 1}{n_i} \sigma_w^2, \tag{B.7}$$

where

$$\frac{1}{n_i} \|A_{mi} C^*_{x_{mi}}\|_2^2 = \frac{1}{n_i} \sum_{i=1}^{n_i} c^{*2}_{x_i} - \frac{1}{n_i^2} (\sum_{i=1}^{n_i} c^*_{x_i})^2, \tag{B.8}$$

therefore

$$E[Y_{Smi}] = \frac{1}{n_i} \sum_{i=1}^{n_i} c^{*2}_{x_i} - \frac{1}{n_i^2} (\sum_{i=1}^{n_i} c^*_{x_i})^2 + \frac{n_i - 1}{n_i} \sigma_w^2. \tag{B.9}$$

$var[Y_{Smi}]$ will be given by calculating the variance of each term in equation(B.6):

$$var[-\frac{2}{n_i^2} \sum_{j \neq k}^{n_i} W_j W_k] = \frac{4}{n_i^4} \sum_{j \neq k}^{n_i} \sigma_{wj}^2 \sigma_{wk}^2, \tag{B.10}$$

$$var[\frac{n_i - 1}{n_i^2} \sum_{i=1}^{n_i} W_i^2] = \frac{(n_i - 1)^2}{n_i^4} (2n_1 \sigma_{w_1}^4 + \cdots + 2n_i \sigma_{w_{n_i}}^4), \tag{B.11}$$

$$var[\frac{-2}{n_i^2} \sum_{i=1}^{n_i} W_i \sum_{j=1}^{n_i} c^*_{x_j}] = \frac{4}{n_i^2} (\sum_{j=1}^{n_i} c^*_{x_j})^2 \sum_{i=1}^{n_i} \sigma_{w_i}^2, \tag{B.12}$$

$$var[\frac{2}{n_i} \sum_{i=1}^{n_i} c^*_{x_i} W_i] = \frac{4}{n_i^2} \sum_{i=1}^{n_i} \sigma_{w_i}^2 c^{*2}_{x_i}, \tag{B.13}$$

$$cov(\frac{-2}{n_i^2}\sum_{i=1}^{n_i} W_i \sum_{j=1}^{n_i} c^*_{x_j}, \frac{2}{n_i}\sum_{i=1}^{n_i} c^*_{x_i} W_i) = \frac{-8}{n_i^3}\sum_{i=1}^{n_i} c^*_{x_i} \sum_{j=1}^{n_i} c^*_{x_j}\sigma^2_{w_j}, \tag{B.14}$$

The rest of possible covariance terms will be zero as samples are i.i.d.:

$$var[Y_{Smi}] = \frac{2(n_i-1)}{n_i^2}\sigma_w^4 - \frac{4}{n_i^3}\sigma_w^2(\sum_{i=1}^{n_i} c^*_{x_i})^2 + \frac{4}{n_i^2}\sigma_w^2\sum_{i=1}^{n_i} c^{*2}_{x_i}. \tag{B.15}$$

Appendix C

Folded Normal Distribution

Let θ be a sample of standard Gaussian distribution, $\theta \sim \mathcal{N}(0,1)$, with density function $\phi(\theta)$ as follows:

$$\phi(\theta) = \frac{1}{\sqrt{2\pi}} e^{-\theta^2/2}, \quad \theta \in R \tag{C.1}$$

therefore, the distribution function of θ can be given as follows:

$$\Phi(\theta) = \int_{-\infty}^{\theta} \phi(v)dv = \int_{-\infty}^{\theta} \frac{1}{\sqrt{2\pi}} e^{-v^2/2} dv, \quad \theta \in R \tag{C.2}$$

We let v be a sample of Gaussian distribution with mean μ and standard deviation σ, $v \sim \mathcal{N}(\mu, \sigma)$. Therefore, random variable V can be written as $V = \mu + \sigma\Theta$. It follows that $W = |V| = |\mu + \sigma\Theta|$ is a random variable with a folded normal distribution, where all of the negative values of V are folded to the positive region of the distribution.

Consequently, cdf of the sorted sample w, where $w \in [0, \infty)$, will be as follows:

$$\begin{aligned} F_a(w) &= P(W \leq w) = P(|V| \leq w) = P(|\mu + \sigma\Theta| \leq w) \\ &= P(-w \leq \mu + \sigma\Theta \leq w) \\ &= P(\frac{-w-\mu}{\sigma} \leq w \leq \frac{w-\mu}{\sigma}) \\ &= \Phi(\frac{w-\mu}{\sigma}) - \Phi(\frac{-w-\mu}{\sigma}) \end{aligned} \tag{C.3}$$

Since $\Phi(-\theta) = 1 - \Phi(\theta)$, we will have:

$$\begin{aligned} F_a(w) &= \Phi(\frac{w-\mu}{\sigma}) - \Phi(\frac{-w-\mu}{\sigma}) \\ &= \Phi(\frac{w-\mu}{\sigma}) + \Phi(\frac{w+\mu}{\sigma}) - 1 \\ &= \int_0^w \frac{1}{\sigma\sqrt{(2\pi)}}\{\exp[-\frac{1}{2}(\frac{v+\mu}{\sigma})^2] \\ &+ \exp[-\frac{1}{2}(\frac{v-\mu}{\sigma})^2]\}dv \end{aligned} \tag{C.4}$$

In case of Gaussian mixture models, the above cdf will be calculated as follows [72]:

$$F_a(z) = \sum_{j=1}^{K} \pi_j F_{aj}(z) \tag{C.5}$$

where $F_{aj}(z)$ is the Gaussian cdf of the jth component, and π_j is the mixing factor of that component in the mixture.

Appendix D

Estimation of α and T

Here, we propose an approach to determine the proper values of α and T with assumption of Gaussian Dcdf. Lets consider a set of possible combinations of α and T values. For each combination, the similarity score $A(\alpha, T)$ can be calculate using (4.9):

$$A(\alpha, T) = \begin{cases} 1 & Sigtest_{score}(\alpha) < T \qquad (H_0) \\ 0 & Sigtest_{score}(\alpha) \geq T \qquad (H_1) \end{cases} \tag{D.1}$$

We consider $A(\alpha, T)$ for two different scenarios: 1) $A_0(\alpha, T)$, where ecdf of the data is related to a single cluster and Dcdf is a single Gaussian as well, 2) $A_1(\alpha, T)$, where, ecdf of the data is relate to two overlapped clusters and Dcdf is a single Gaussian. A proper combination of α and T should suggest to split the two clusters, and at the same time should not split the single cluster. In Figure D.1, the synergistic combinations are shown by warmer colors for different distances between two overlapped clusters.

Consequently, the following minimization problem can be used to estimate the proper α and T:

$$[\hat{\alpha}, \hat{T}] = \min_{\alpha, T} [A_1(\alpha, T) - A_0(\alpha, T)] \tag{D.2}$$

where $0 < T \leq 1$ and $0 < \alpha \leq \sqrt{\frac{1}{1-p_c}}$ (i.e. $p_c = 0.99$ in (4.6)) are possible constraints

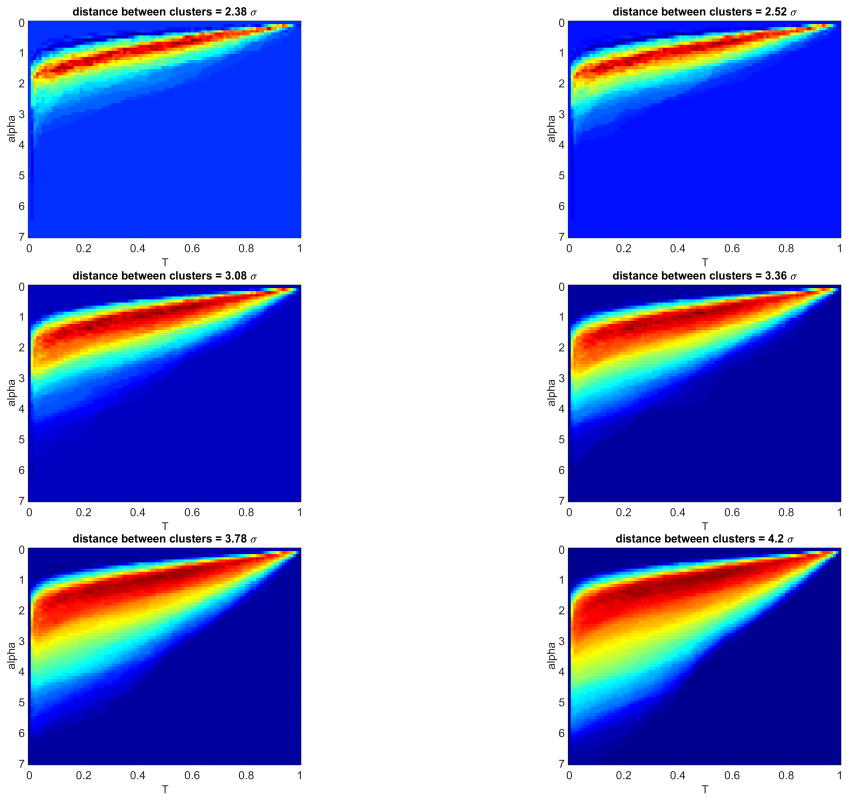


Figure D.1: Increasing the distance between clusters and representing the result of Sigtest for different combinations of α and T (warmer points show more reliable combinations).

for the minimization problem.

To solve the (D.2), we have employed genetic algorithm (GA) with a population size of 100. Figure D.2 shows the result of GA simulations for estimating α and T for different distances between the center of clusters. This result is consistent with the synergistic regions in the Figure D.1 for the optimum α and T. Consequently, to be in a safe range for the optimum behavior of Sigtest, we set the averaged values of 1.72 and 0.53 for α and T respectively.

In the case of Gaussian mixture models, the T value will be adaptively calculated based on the assumed mixing factor π_j in (C.5). As a result, the T value in a mixture model will be give as follows:

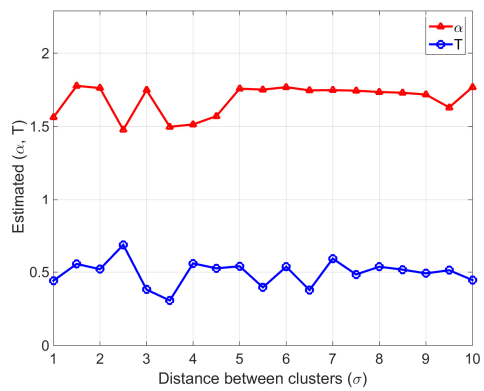


Figure D.2: Estimated α and T parameters using Genetic algorithm for different distances between clusters.

$$T = 0.53 \max_j \pi_j \tag{D.3}$$

Bibliography

[1] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[2] F. U. Siddiqui and N. A. M. Isa. Enhanced moving k-means (emkm) algorithm for image segmentation. *Consumer Electronics, IEEE Transactions on*, 57(2):833–841, 2011.

[3] M. Hua, M. K. Lau, J. Pei, and K. Wu. Continuous k-means monitoring with low reporting cost in sensor networks. *Knowledge and Data Engineering, IEEE Transactions on*, 21(12):1679–1691, 2009.

[4] C. C. Hung, , and L. Wan. Hybridization of particle swarm optimization with the k-means algorithm for image classification. In *In Computational Intelligence for Image Processing CIIP'09. IEEE Symposium on*, pages 60–64, 2009.

[5] Woncheol Jang and Martin Hendry. Cluster analysis of massive datasets in astronomy. *Statistics and Computing*, 17(3):253–262, 2007.

[6] Daniel B Neill and Andrew W Moore. A fast multi-resolution method for detection of significant spatial disease clusters. In *Advances in Neural Information Processing Systems*, page None, 2003.

[7] Marcilio CP De Souto, Daniel SA De Araujo, Ivan G Costa, Rodrigo GF Soares, Teresa B Ludermir, and Alexander Schliep. Comparative study on normalization

procedures for cluster analysis of gene expression datasets. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 2792–2798. IEEE, 2008.

[8] Noam Kaplan, Moriah Friedlich, Menachem Fromer, and Michal Linial. A functional hierarchical organization of the protein sequence space. *BMC bioinformatics*, 5(1):196, 2004.

[9] Deng Cai, Xiaofei He, Zhiwei Li, Wei-Ying Ma, and Ji-Rong Wen. Hierarchical clustering of www image search results using visual, textual and link information. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 952–959. ACM, 2004.

[10] Sara Dolničar and Friedrich Leisch. *Behavioral market segmentation of binary guest survey data with bagged clustering*. Springer, 2001.

[11] Michel Wedel and Wagner A Kamakura. *Market segmentation: Conceptual and methodological foundations*, volume 8. Springer Science & Business Media, 2012.

[12] Michael J Shaw, Chandrasekar Subramaniam, Gek Woo Tan, and Michael E Welge. Knowledge management and data mining for marketing. *Decision support systems*, 31(1):127–137, 2001.

[13] Stijn Van Dongen. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, 30(1):121–141, 2008.

[14] R. Xu and D. Wunsch. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645–678, 2005.

[15] P. D. McNicholas and S. Subedi. Clustering gene expression time course data using mixtures of multivariate t-distributions. *Journal of Statistical Planning and Inference*, 142(5):1114–1127, May.

[16] A. K. Jain. Data clustering: 50 years beyond k-means,. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[17] C. C. Aggarwal and C. K. Reddy. volume 31. CRC Press, 2013.

[18] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[19] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108, 1979.

[20] L. Baibing. A new approach to cluster analysis: the clustering function based method. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):457–476, 2006.

[21] R. De la Cruz-Mesa, F. A. Quintana, and G. Marshall. Model-based clustering for longitudinal data. *Computational Statistics and Data Analysis*, 52(3):1441–1457, 2008.

[22] M. Chiang and B. Mirkin. Intelligent choice of the number of clusters in kmeans clustering: An experimental study with diferent cluster spreads. *Journal of Classification*, 27(1):3–40, 2010.

[23] X. Xie and G. Beni. A validity measure for fuzzy clustering. *Pattern Analysis and Machine Intelligence, IEEE Trans*, 8:841–847, 1991.

[24] J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4(1):95–104, 1974.

[25] L. Kaufman and P. J. Rousseeuw. Finding groups in data: an introduction to cluster analysis. *John Wiley and Sons*, 344, 2008.

[26] T. Caliski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[27] W. J. Krzanowski and Y. T. Lai. A criterion for determining the number of groups in a data set using sum-of-squares clustering,. *Biometrics*, 44(1):23–34, 1988.

[28] A. Strehl. *Relationship-based clustering and cluster ensembles for high dimensional data mining*. PhD thesis, The University of Texas at Austin, May 2002.

[29] N. Yousri, M. Kamel, and M. Ismail. A novel validity measure for clusters of arbitrary shapes and densities. pages 1–4. Pattern Recognition 2008. ICPR 2008. 19th International Conference on, 2008.

[30] Gayar J. Kittler K. Kryszczuk, P. Hurley and F. Roli (Eds.). Estimation of the number of clusters using multiple clustering validity indices, in: N. *in Multiple Classifier Systems*, pages 114–123, 2010.

[31] D. Pelleg and A. W. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. *Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00, Morgan Kaufmann Publishers Inc.*, pages 727–734, 2000.

[32] G. Hamerly and C. Elkan. Learning the k in kmeans. *in Neural Information Processing Systems*, 17(281), 2004.

[33] A. Kalogeratos and A. Likas. Dip-means: an incremental clustering method for estimating the number of clusters. *In Advances in neural information processing systems*, pages 2393–2401, 2012.

[34] G. Hamerly and Y. Feng. Pg-means: learning the number of clusters in data. In *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, p. 393. MIT Press, 2007. vol. 19.

[35] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2000.

[36] Shalini S Singh and NC Chauhan. K-means v/s k-medoids: A comparative study. In *National Conference on Recent Trends in Engineering & Technology*, volume 13, 2011.

[37] Alan P Reynolds, Graeme Richards, and Vic J Rayward-Smith. The application of k-medoids and pam to the clustering of rules. In *Intelligent Data Engineering and Automated Learning–IDEAL 2004*, pages 173–178. Springer, 2004.

[38] J. Zheng K. Wang, J. Zhang, and J. Dong. Estimating the number of clusters via system evolution for cluster analysis of gene expression data. *Information Technology in Biomedicine, IEEE Transactions on*, 13(5):848–853, 2009.

[39] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.

[40] Tao Shi, Mikhail Belkin, and Bin Yu. Data spectroscopy: Eigenspaces of convolution operators and clustering. *The Annals of Statistics*, pages 3960–3984, 2009.

[41] Ana Fred. Finding consistent clusters in data partitions. In *Multiple classifier systems*, pages 309–318. Springer, 2001.

[42] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd*, 96(34):226–231, August 1996.

[43] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[44] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV SLVC Workshop*, 2004.

[45] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[46] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[47] L. I. Kuncheva and D. P. Vetrov. Evaluation of stability of k-means cluster ensembles with respect to random initialization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1798–1808, 2006.

[48] B. S. Everitt, S. Landau, M. Leese, and D. Stahl. *Hierarchical clustering.* Cluster Analysis, 71-110, 5th edition, 2011.

[49] T. K. Moon. The expectation-maximization algorithm. *Signal processing magazine, IEEE*, 13(6):47–60, 1996.

[50] B. Flach and P. Hlavac. Expectation maximization algorithm. *Computer Vision: A Reference Guide*, pages 265–268, 2014.

[51] S. Borman. The expectation maximization algorithm-a short tutorial. *Submitted for publication*, pages 1–9, 2004.

[52] T. W. Chen, C. H. Sun, H. H. Su, S. Y. Chien, D. Deguchi, I. Ide, and H. Murase. Power-efficient hardware architecture of k-means clustering with bayesian-information-criterion processor for multimedia processing applications. *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on*, 1(3):357–368, 2011.

[53] Q. Zhao, M. Xu, and P. Franti. Knee point detection on bayesian information criterion. pages 431–438, Vol. 2, 2008. In Tools with Artificial Intelligence ICTAI'08, 20th IEEE International Conference on.

[54] F. J. Jr Massey. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[55] H. W. Lilliefors. On the kolmogorov-smirnov test of normality with mean and variance unknown. *Journal of the American Statistical Association*, 62(318):399–402, 1967.

[56] T. W. Anderson and A. Darling. A test of goodness of fit. *Journal of the American Statistical Association*, 49(268):765–769, 1954.

[57] C. D. Sinclair, B. D. Spurr, and M. I. Ahmad. Modified anderson darling test. *Communications in Statistics-Theory and Methods*, 19(10):3677–3686, 1990.

[58] J. A. Hartigan and P. M. Hartigan. The dip test of unimodality. *The Annals of Statistics*, 13(1):70–84, 1985.

[59] I. Jolliffe. *Principal component analysis*. John Wiley and Sons, Ltd, 2002.

[60] N. Vlassis J. J. Verbeek and B. Krse. A k-segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23(8):1009–1017, June 2002.

[61] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[62] FRBMI Jordan and F Bach. Learning spectral clustering. *Adv. Neural Inf. Process. Syst*, 16:305–312, 2004.

[63] S. Beheshti and M. A. Dahleh. Noisy data and impulse response estimation. *IEEE Transactions on Signal Processing*, 58(2):510–521, 2010.

[64] S. Beheshti, M. Hashemi, X. P. Zhang, and N Nikvand. Noise invalidation denoising. *Signal Processing, IEEE Transactions on*, 58(12):6007–6016, 2010.

[65] S. Beheshti and M. Dahleh. A new information-theoretic approach to signal denoising and best basis selection. *Signal Processing, IEEE Transactions on*, 53(10):3613–3624, 2005.

[66] M. Shahbaba and S. Beheshti. Mace-means clustering. *Signal Processing, Elsevie*, 105:216–225, 2014.

[67] L. Vendramin, R. J. G. B. Campello, and E. R. Hruschka. Relative clustering validity criteria: A comparative overview. *Statistical Analysis and Data Mining*, 3(4):209–235, 2010.

[68] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[69] A. R. da Rocha Neto, R. Sousa, G. de A. Barreto, and J. S. Cardoso. Diagnostic of pathology on the vertebral column with embedded reject option. In *Proceedings of the 5th Iberian conference on Pattern recognition and image analysis, IbPRIA'11, Springer-Verlag, BerlinHeidelberg*, pages 588–595, 2011.

[70] P. Fränti and O. Virmajoki. Iterative shrinking method for clustering problems. *Pattern Recognition*, 39(5):761–775, 2006.

[71] S. Ayramo and T. Karkkainen. Introduction to partitioning-based clustering methods with a robust example, reports of the dept. of math. Technical report, Inf. Tech. (Series C. Software and Computational Engineering), 1/2006, University of Jyvskyl, 2006.

[72] C. R. Shalizi. *Advanced Data Analysis from an Elementary Point of View.* ch.19, sec. 4, pp. 378- 384. [Online]. Available, 2012.

[73] M. Shahbaba and S. Beheshti. *Efficient unimodality test in clustering by signature testing*. Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on, 2014.

[74] M. Shahbaba and S. Beheshti. Model verification of gmm clustering based on signature testing. Electrical and Computer Engineering (CCECE), 2014 IEEE 27th Canadian Conference on, 2014.

[75] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[76] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image classification using random forests and ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[77] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[78] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *on Generative-Model Based Vision, Workshop*, 2004.

[79] D. Cai, X. He, J. Han, and T. Huang. Graph regularized non-negative matrix factorization for data representation. In *PAMI*, 2011.

[80] D. Cai, X. He, and J. Han. Speed up kernel discriminant analysis. *The VLDB Journal*, 20(1):21–33, 2011.

[81] M. Lichman. *UCI Machine Learning Repository*. University of California, School of Information and Computer Science, CA, 2013.

[82] M. Meil. Comparing clusterings-an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007.

[83] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193218, 1985.

[84] S. Pemmaraju and S. Skiena. Minimum spanning trees. *Computational Discrete Mathematics: Combinatorics and Graph Theory in Mathematica. Cambridge, England: Cambridge University Press*, 8(2):335–336, 2003.

[85] L. Galluccio, O. Michel, P. Comon, M. Kliger, and A. O. Hero. Clustering with a new distance measure based on a dual-rooted tree. *Information Sciences*, 251:96–113, 2013.

[86] R. C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389,1401, November 1957.

[87] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 7:48–50, 1956.

[88] A. Ultsch. Clustering with som: U*c. *In Proc. Workshop on Self-Organizing Maps, Paris France*, pages 75–82, 2005.

[89] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996.

[90] Fazli Can. Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems (TOIS)*, 11(2):143–164, 1993.

Glossary

ACE Averaged Central Error.

AD Anderson-Darling Statistical Test of Gaussianity.

ARI Adjusted Rand Index.

AP Affinity Propagation.

BIC Bayesian Information Criterion.

BOVW Bag of Visual Words.

cdf cumulative distribution function.

CH index Calinski-Harabasz index.

CNC Correct Number of Clusters.

DB index Davies-Bouldin index.

DBSCAN Density-based spatial clustering of applications with noise.

Dcdf Desired cumulative distribution function.

Dip Dip statistic's Hartigan Test of Unimodality.

ecdf empirical cumulative distribution function.

EM Expectation Maximization.

GA Genetic Algorithm.

GMM Gaussian Mixture Model

HOG Histograms of Oriented Gradients.

KL index Krzanowski-Lai index.

KS Kolmogrov-Smirnov Statistical Test of Gaussianity.

MACE Minimum Averaged Central Error.

MCMC Markov Chain Monte Carlo.

minPAS Minimum Pathways in Arbitrary Shaped Clustering.

minPts minimum number of points for each cluster in DBSCAN.

MSDL Minimum Structure Description Length.

MST Minimum Spanning Tree.

N-cut Normalized cut.

PC Principal Component.

PCA Principal Component analysis.

SIFT Scale Invariant Feature Transformation.

Sigtest Signature Testing.

Sil index Silhouette index.

SNR Signal To Noise Ratio.

STD Standard Deviation.

SVM Support Vector Machines.

VI Variation of Information.

wtertra weighted inter-to intra-cluster ratio.