Intelligent K-Means Clustering in L_2 and L_1 Versions: Experimentation and Application

Ming-Tso Chiang

A Thesis Submitted in Fulfillment of the Requirements for
the Degree of Doctor of Philosophy in the
University of London

June 2009
School of Computer Science and Information Systems
Birkbeck College, University of London

Declaration

I hereby declare that the work presented in this thesis is my own, and that it has not previously been submitted for a degree or award at this or any other academic institution.

Signed:

Ming-Tso Chiang

Abstract

A promising clustering method, “intelligent” version of K-Means, iK-Means, which finds the number of clusters K and initializes K-Means with the so-called Anomalous pattern (AP) clusters, has been proposed and tested on several real-world data sets (Mirkin 2005). The subject of this thesis is to further analyse the performance of iK-Means method, in two versions – L_2 and L_1, involving respectively the squared Euclidean distance and mean centroids, and the city block distance and median centroids. Firstly, one needs to see if there is any difference between results of these methods at all, and if there is, what data structures are better served by each. Secondly, one needs to compare the iK-Means with a host of other methods for obtaining the number of clusters published in the literature and to this end, to adopt or develop a technique for simulation studies. We propose a technique for modelling Gaussian clusters and their intermix. In experiments conducted over this model, iK-Means appear to outperform the others on the cluster and centroid recovery, though it may fail sometimes on the number of clusters. The iK-Means methods are applied then to solving an unconventional task in gene expression analysis: finding genes differently expressed in different types of cells.

Contents

Acknowledgement

First and foremost, my gratitude goes out to my supervisor Prof. Boris Mirkin. What I have learned for these years is not only machine learning knowledge, especially clustering, but also the attitude of doing research in the future. I will always be grateful for his encouragement, continuous support and indefatigable guidance.

I would also like to thank the friendly members of the Systems Group of School of Computer Science and Information Systems: Phil Gregg, Phil Docking, Andrew Watkins and Petar Konovski, which have given me full technical support. Especially, Phil Gregg and Petar gave me useful suggestions and support when running the experiments. I would specifically like to thank Prof. B Chain from the Virology Department of UCL, which provides the gene expression data for analysis.

Last, but certainly not least, my utmost gratitude must go to my parents, to whom I will be forever indebted for their love, support, wise guidance and their dedication throughout the years, without which I would have never been in the position to write this thesis. Finally, I owe a deep appreciation to my wife Yongshuo for her inexhaustible patience, devotion during these years when we started this long and unpredictable journey together.

List of Tables

List of Figures

List of Acronyms

AL_1:	Least Moduli Criterion of HT-adjusted iK-Means Clustering
AL_2:	Least Square Criterion of HT-adjusted iK-Means Clustering
AP:	Anomalous Pattern
ARI:	Adjusted Rand Index
BIC:	Bayesian Information Criterion
CCIA:	Cluster Centre Initialization Algorithm
CD:	Consensus Distribution area
CDF:	Cumulative Distribution Function
cDNA:	Complementary DNA
CH:	Calinski and Harabasz Index
DBMSDC:	Density-Based Multi Scale Data Condensation
DC:	Dendritic Cells
DD:	Average Distance between Partitions
DT:	Discarding Threshold
GS:	Gap Statistic
HT:	Hartigan's Rule
iK-Means:	Intelligent K-Means
ISODATA:	Iterative Self-Organizing Data Analysis Techniques
JS:	Jump Statistic
KDD:	Knowledge Discovery in Database
L_1:	Least Moduli Criterion of iK-Means Clustering
L_2:	Least Square Criterion of iK-Means Clustering
LOESS:	Locally Estimated Scatterplot Smoothing
LOWESS:	Locally Weighted Scatterplot Smoothing
LVQ:	Learning Vector Quantization
DL:	Minimum Discription Length
MST:	Minimum Spanning Tree
PB:	Pivot-based without removal normalization method
PBR:	Pivot-based with removal normalization method
PPCA:	Probabilistic Principal Component Analysis
VQ:	Vector Quantization

List of Publications

Chiang M. M.T. and Mirkin B., Intelligent choice of the number of clusters in K-Means clustering: an experimental study with different cluster spreads, Journal of Classification, In press.

Chiang M. M.T. and Mirkin B. (2007), Experiments for the number of clusters in K-Means, Progress in Artificial Intelligence, EPIA 2007, LNAI 4874, 395-405.

Chiang M. M.T. and Mirkin B. (2006), Determining the number of clusters in the Straight K-Means: Experimental comparison of eight options, Proceeding of the 2006 UK workshop on Computational Intelligence, 119-126.

Chapter 1

K-Means Clustering and its Issues

There are a lot of data and reports generated in the public and private sectors everyday and how to deal with them efficiently and transfer them into useful information for decision support is a very important issue. In order to achieve this goal, one needs data collection, analysis and evaluation process. Generally, this process is called knowledge discovery and because the data is stored in a database, it is also known as knowledge discovery in databases (KDD) or data mining (Liu and Motoda 1998).

The definitions of data mining have been proposed in many publications (Cabena et al. 1997, Grupe and Owrang 1995, Berry and Linoff 1997, Kleissner 1998, Frawley et al. 1992) and the definition that Frawley et. al (1992) proposed is the most common version, that is, the non trivial extraction of implicit, previously unknown, and potentially useful information from data. Berry and Linoff (1997) described a four-stage process of data mining: identifying problems, transferring data into results, analyzing and evaluating results and these stages are repeated during data mining. Data mining involves the use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data sets (Edelstein 1999, Adriaans and Zantinge 1996).

Cluster analysis is an important technique in data mining and the process is to partition data into clusters (groups or classes) so that objects in the same cluster have high similarity in comparison to each other, that is, homogeneous, but are very dissimilar to objects in other clusters, that is, heterogeneous (Aldenderfer and Blashfield 1984, Tian et al. 2005). K-Means is the simplest, fastest and the most commonly used clustering method (see Bock 2007, Steinley 2006) that applies to a data set involving the set of *N* entities, *I*, the set of *M* features, *V*, and the entity-to-feature matrix $Y=(y_{iv})$, where y_{iv} is the value of feature $v \in V$ at entity $i \in I$. The method produces a partition $S=\{S_1, S_2,\ldots, S_K\}$ of *I* in *K* non-overlapping classes S_k, referred to as clusters, each with a centroid $c_k=(c_{kv})$, an M-dimensional vector in the feature space (k=1,2,…K). Centroids form set $C=\{c_1, c_2,\ldots, c_K\}$. The criterion, minimized by the method, is the within-cluster summary distance to centroids:

$$W(S, C)=\sum_{k=1}^{K}\sum_{i\in S_k} d(i, c_k) \qquad (1)$$

where *d* is a distance measure, typically the Euclidean distance squared or Manhattan distance. In the former case criterion (1) (see page 2) is referred to as the square error criterion (least square criterion (L_2)) and in the latter, the absolute error criterion (least moduli criterion (L_1)).

Given *K* M-dimensional vectors c_k as cluster centroids, the algorithm updates clusters S_k according to the Minimum distance rule: For each entity *i* in the data table, its distances to all centroids are calculated and the entity is assigned to its nearest centroid. Given clusters S_k, centroids c_k are updated according to the distance *d* in criterion (1) (see page 2), k=1, 2, …, K. Specifically, c_k is calculated as the vector of within-cluster averages if *d* in (1) is Euclidean distance squared and as of within-cluster medians if *d* is Manhattan distance. This process is reiterated until clusters S_k stabilize. Before running the algorithm, the original data is to be pre-processed (standardized) by subtracting the grand mean from each feature and further divided it by its range in our experimental settings, described on page 46. This algorithm will be referred to as Straight or Batch K-Means which will be implemented in my research.

When the distance d in (1) is indeed the squared Euclidean distance, K-Means can be seen as an implementation of the alternating optimization procedure for maximization of the maximum likelihood under the assumed mixture of "spherical" Gaussian distributions model, in which all covariance matrices are equal to a diagonal matrix $\sigma^2 I$ where I is the identity matrix and σ^2 the variance value (Hartigan 1975, Banfield and Raftery 1993, McLachlan and Peel 2000). Another, somewhat lighter interpretation comes from the data mining paradigm, in which (1) is but the least-squares criterion for approximation of the data with a data recovery clustering model (Mirkin 1990, 2005) that states that every entry y_{iv} in the data matrix (*i* denotes an entity and *v* a feature), can be presented as approximated by the "hidden" set of clusters $S=\{S_1, S_2,\ldots, S_K\}$ and their centers $\boldsymbol{C}=\{\boldsymbol{c}_1, \boldsymbol{c}_2,\ldots, \boldsymbol{c}_K\}$ through equations

$$y_{iv} = \sum_{k=1}^{K} c_{kv} s_{ik} + e_{iv}, \qquad (2)$$

where $\boldsymbol{s}_k=(s_{ik})$ is S_k membership vector in which $s_{ik}=1$ if $i \in S_k$ and $s_{ik}=0$ otherwise, and e_{iv} are residuals to be minimized over unknown $\mathbf{c}_k$ and $\mathbf{s}_k$ ($k=1,2,\ldots,K$). Criterion (1) (see page 2) is the least-squares or least-moduli fitting criterion for model (2) (see page 3) if *d* in (1) is the squared Euclidean distance or Manhattan distance, respectively.

A version of K-Means in which the number of clusters and initial centroids are determined beforehand with a procedure targeting anomalous patterns as the candidates for the initial centroids has been proposed in Mirkin (2005) under the title of "intelligent K-Means" (iK-Means). It initializes K-Means by standardizing the data in such a way that the origin is put into a point, usually the gravity centre of all the data points, rescaling it by dividing the range, and iterating then the so-called Anomalous Pattern algorithm described in the box below:

Anomalous Pattern (AP):

```
1. Find an entity in $I$, which is the farthest from the origin and put it as the
   AP centroid $\boldsymbol{c}$.
2. Calculate distances $d(y_i,\boldsymbol{c})$ and $d(y_i,0)$ for each $i \in I$, and assign $y_i$ to the
   AP cluster $S$ if $d(y_i\ \boldsymbol{c})<d(y_i,0)$.
3. Calculate the centroid $\boldsymbol{c}'$ of the $S$ found on step 2. If $\boldsymbol{c}'$ differs from $\boldsymbol{c}$, put
   $\boldsymbol{c}'$ as $\boldsymbol{c}$, and go to step 2, otherwise go to step 4
4. Output $S$ and its centroid $\mathbf{c}$ as the Anomalous Pattern.
```

The AP algorithm starts from that entity, which is the farthest from the origin, as the initial centroid ***c***. After that, a one-cluster version of the generic K-Means is utilized. The current AP cluster S is defined as the set of all those entities that are closer to ***c*** than to the origin, and the next centroid ***c*** is defined as the center of gravity of S. This process is iterated until convergence. The convergence is guaranteed because the process alternates between minimizing the criterion (1) (see page 2) at K=2 with $S_1=S$, $S_2=I-S$, and centroids $\boldsymbol{c_1=c}$ and $\boldsymbol{c_2=0}$, and the origin which is kept unchanged through the iterations. The final S, along with its centroid

c and its contribution to the data scatter, is the output AP cluster. After it is removed from the data set, the process of extracting AP clusters is reiterated without ever changing the origin, until no entity remains. Centroids of those AP clusters that have more than one entity are used as **c** set at the initialization of K-Means.

This is a version of the so-called Principal cluster analysis approach that emulates the one-by-one strategy of the Principal component analysis applied to model (2) (see page 3): an AP pattern is a cluster derived from model (2) (see page 3) at $K=1$ in such a way that it maximally contributes to the data scatter (Mirkin 1990). The fact that AP cluster is far away from the origin conforms to the notion of interestingness in data mining: the farther from origin, the more interesting (Fayyad et. al 1996). The iK-Means algorithm iteratively applies the Anomalous Pattern procedure to the yet un-clustered part of the data until no entities remain out of the anomalous patterns. Those of the anomalous patterns that are not numerous, that is, singletons and, in general, those whose cardinality is less than or equal to a pre-specified discarding threshold *DT*, are removed from the set of anomalous patterns. (In our experiments, $DT=1$.) Those remaining are used to initialize K-Means: K is the number of remaining APs, and their centroids are taken to initialize K-Means. The algorithm is formulated in the box.

Intelligent K-Means:

```
0. Put $t=1$ and $I_t = I$, the original entity set, and standardize the data in such
   a way that the origin is put in the grand mean; the feature ranges are used
   for scaling.
1. Apply AP to $I_t$ to find $S_t$ and $C_t$.
2. If $S_t \neq I_t$, put $I_t \leftarrow I_t - S_t$, $t \leftarrow t+1$ and go to step 1, otherwise, proceed to 3.
3. Remove all of the found clusters whose cardinality is less than or equal
   to the discarding threshold $DT$. Denote the number of remaining clusters
   by $K$ and their centroids by $c_1, c_2, \ldots, c_K$.
4. Do Straight K-Means with $c_1, c_2, \ldots, c_K$ as initial centroids.
```

The intelligent K-Means procedure seems appealing both intuitively and computationally, and it leads to interpretable solutions in real-world problems. Therefore, it seems reasonable to put it to empirical testing. A version of the method, with a pre-specified K and with no removal of singletons, has been tested by Steinley and Brusco (2007), leading to rather mediocre results in their experiments. Here we intend to test the original version of the iK-Means as a device for identifying both the number K and initial centroids.

The distance and centroids in iK-Means are defined differently depending on the criterion in the corresponding data recovery model. Specifically, with the least squares (square error) criteria, the distance is Euclidean squared and the cluster centroid is defined by the within-cluster feature averages. With the absolute error criterion, the distance is Manhattan, also referred to as city-block, and the cluster centroid is defined by the within cluster feature medians.

The iK-Means algorithm has the following features:

(a) it uses just one run of the iterative AP algorithm over set I,

(b) it utilizes yet another parameter, the discarding threshold, which is taken to be *DT=1* in the follow-up experiments,

(c) it involves an automatic determination of both the K and initial centroids.

The main difficulty remaining among the clustering methods is the determination of the “right” number of clusters (for reviews, see Jain and Dubes (1988), Dudoit and Fridlyand (2002), Mirkin (2005), Steinley (2006)). Ball and Hall (1965) proposed an ISODATA algorithm. The algorithm begins with a random partition and centroids and any clusters that do not have enough observations are discarded. Bischof et al. (1999) developed a method based on minimum description length (MDL). Starting from a large number of K, the algorithm removes clusters whenever the description length can be reduced, and any clustering algorithm, for example, K-Means can be used at each step to optimize the model fit to the data. The whole process is continued until it converges. Kothari and Pitts (1999) proposed a scale-based method for determining the number of clusters, which modified the within-cluster summary distance to centroids (see Eq (1) on page 2) of traditional K-Means.

Some papers propose a procedure for estimating the number of clusters and experimentally comparing it to some other methods and some authors do more comprehensive experiments and either arrive at some winning procedures, like Milligan and Cooper (1985) in their seminal study of 30 indexes for cutting cluster hierarchies, or obtain inconclusive results like Hardy (1996) and Dimitraidou et al.

(2002). Milligan and Cooper (1985) proposed a Monte Carlo evaluation of the performance of 30 numerous cluster numbers determination procedures when applied to the analysis of artificial data sets containing 2, 3, 4, or 5 distinct clusters by four hierarchical clustering methods. Milligan and Cooper (1987) wrote a clustering methodology review and gave the practitioners in clustering some useful recommendations not only in methods but also in applied analysis. Hardy (1996) evaluates 7 methods over 6 different data sets and suggests trying several clustering techniques on the data and gathers more information to determine the number of clusters. Dimitraidou et al. (2002) present a comparison of 15 different validity indexes for the binary data sets consisting of 4, 5, or 6 clusters by two clustering algorithms: K-Means and hard competitive learning, but come to no definite conclusions. Steinley and Henson (2005) pointed out that it is very important, in experiments with simulated data, to maintain a degree of cluster overlap to be able to derive any realistic conclusions, which was not the case in previously published experimental studies. They propose a model for data generation with overlapping clusters, which however contains too many parameters and can model only one-dimensional overlaps. In a follow-up experimental study of different initialization strategies, Steinley and Brusco (2007) come to the conclusion that cluster overlap is the property of generated data that most affects the cluster recovery.

A promising clustering method, “intelligent” version of K-Means, iK-Means, which initializes K-Means with the so-called Anomalous pattern (AP) clusters that are furthest away from the origin of the feature space, has been proposed and tested on several real-world data by Mirkin (2005) and this research is oriented towards investigation of this method. The most important question is

(1) Whether iK-Means is good for finding the number of clusters?

To answer this question, one needs to address the following issues:

1a. Make a review of the literature and select methods for finding K with which to compare

1b. Put a data generator that allows a comparison between methods along with addressing the issue of modelling the overlap between clusters

1c. Define evaluation criteria for the results of experiments

1d. Conduct the experiments

1e. Using the results of the experiments, find out if any improvement of iK-Means is possible at all

If the answer to question 1 is positive in general, as we expect, we are interested in

(2) further exploration of the relationship between L_2 and L_1 versions of the method. Specifically, we are interested to see:

2a. Do these methods give similar results on all data structures, or could they lead to different results?

If the answer to 2a is that these methods give different results, as we expect, then we have a more specific issue:

2b. Whether these methods are oriented at different data structures? That is, if there is a data structure type that is better suitable for L_2 version and a data structure type that is more suitable for L_2 version?

2c. Is it possible to utilise the differences between the two iK-Means methods in a concerted application of them to a real-world problem?

Accordingly, the contents of the thesis is organised along the lines of the inquiry. The question 1 is treated in Chapter 2 (devoted to 1a), Chapter 3 (devoted to 1a-1c) and Chapter 4 (devoted to 1d-1e). Chapter 2 contains a review of methods for finding the right *K* in K-Means in the published literature. We distinguish between five approaches as based primarily on: cluster variance, within-cluster cohesion versus between-cluster separation, consensus distribution, hierarchical clustering, and resampling. The setting of our experiments at the comparison of nine selected methods for finding the "right clustering" – the data sizes, the cluster shapes, the within- and between-cluster spread parameters, and evaluation criteria - is described in Chapter 3. Chapter 4 presents results of our experiments in tables containing the evaluation criteria values, averaged over multiple data generations at each of the twelve data settings, along with issues raised before the experiments and answers to them coming from the results. Question 2 is treated in Chapter 5 (2a and 2b) and 6 (2c). Finally the conclusion reviews the results and questions which remain unanswered.

Chapter 2

Choosing K in K-Means: A Review

There have been a number of different proposals in the literature for choosing the right K after multiple runs of K-Means (Halkidi et al. 2001, Maulik and Bandyopadhyay 2002a, Kothari and Pitts 1999, Vesanto 2001, Hansen and Mladenovic 2001, Steinley 2006, Steinley and Brusco 2007, Likas et al. 2003, Hand and Krzanowski 2005, Ray and Turi 1999, Sugar and James 2003, Steinley 2004, Shen et. al 2005, Pena et al. 1999, Pelleg and Moore 2000, Mirkin 1996, Mirkin 2005, Leisch 2006, Kuncheva, and Vetrov 2005, Krzanowski and Lai 1985, Kaufman and Rousseeuw 1985, Jain, and Dubes 1988, Fraley and Raftery 2002, Steinley 2003, Steinbach et. al 2000, Pena et. al 1999, Babu and Murty 1993, Thiesson et al. 1997, Khan and Ahmad 2004, He et al. 2004, Hamerly and Elkan 2002, Paterlini and Krink 2006, Redmond and Heneghan 2007, Jiwei 2001, Jain et. al 1999, Breckenridge 1989 etc.). We can categorize them into five main approaches:

A. Variance approach: comparing the within-cluster summary distance to centroids at different *K*;
B. Within-cluster cohesion vs. between-cluster separation: comparing values of another characteristic of the cluster structure;
C. Consensus approach: using on all random initizaliztion runs rather than on just the best one to arrive at a “compromise” solution;
D. Hierarchical approach: choosing *K* according to the results of a divisive

or agglomerative clustering procedure

E. Resampling approach: choosing K according to the similarity of clusterings generated on random samples or perturbed data

We describe them in the following subsections. Let us denote the minimum of criterion (1) (see page 2) at a specified K by W_K. Empirically, one can run K-Means R times starting using random subsets of K entities for initialization and use the minimum value of criterion (1) (see page 2) at obtained clusterings as a W_K estimate.

2.1 Variance approach

There have been several different W_k based indices proposed to estimate the number of clusters K (see Calinski and Harabasz (1974), Hartigan (1975), Krzhanowski and Lai (1985), Tibshirani et al. (2001), Sugar and James (2003)). The issue is that W_K itself cannot be used for the purpose since it monotone decreases when K grows. Thus, various “more sensitive” characteristics of the function have been utilized based on intuitive or statistical modeling of the situation. Of those, we choose the following four: two heuristic measures that have been experimentally approved by Milligan and Cooper (1985): a heuristic rule by Hartigan (Hartigan 1975), a Fisher-wise criterion by Calinski & Harabasz (Calinski and Harabasz 1974), and two model-based more recent indexes: Gap Statistic (Tibshirani et al. (2001)) and a statistical model based Jump Statistic (Sugar and

James 2003), as a representative set. Before running the algorithm, the original data is to be normalized in our experiments.

The heuristic rule by Hartigan (Hartigan 1975) utilizes the intuition that when clusters are well separated. "A crude rule of thumb", Hartigan (1975, p. 91) is proposed by calculating $HT=(W_K/W_{K+1}-1)(N-K-1)$, where N is the number of entities, while increasing K so that the very first K at which HT becomes less than 10 is taken as the estimate of K^*. Hartigan's rule can be considered a partition-based analogue to the Duda and Hart (1973) criterion involving the ratio of the criterion (1) (see page 2) at a cluster and at its two-cluster split, which came very close second-best winner in the experiments of Milligan and Cooper (1985). It should be noted that, in our experiments, the threshold 10 in the rule is not very sensitive to 10-20% changes.

The Fisher-wise criterion by Calinski and Harabasz (1974) finds K maximizing $CH=((T-W_K)/(K-1))/(W_K/(N-K))$, where $T=\sum_{i\in I}\sum_{v\in V} y_{iv}^2$ is the data scatter, that is, the sum of all entities y_{iv} squared. The data scatter can be seen as the summary contributions of all features, where the contribution of feature v to the data scatter is defined as the distance of the M-dimensional column from zero column: $T_v=\sum_{i\in I} y_{iv}^2$. The concept of the data scatter plays an important role in data standardization, which is explained in Section 3.1. This criterion showed the best performance in the experiments by Milligan and Cooper (1985), and was subsequently utilized by some authors for choosing the number of clusters (for example, Casillas et al. 2003).

The Gap Statistic introduced by Tibshirani et al. (2001) has become rather popular, especially, in the bioinformatics community. This method compares the value of (1) with its expectation under the uniform distribution. Analogously to the previously described methods, it takes a range of K values and finds W_K for each K. To model the reference values, a number, B, of uniform random reference datasets over the range of the observed data are generated so that criterion (1) (see page 2) values W_{Kb} for each $b=1,\ldots,B$ are obtained. The Gap statistic is defined as $Gap(K)=1/B\sum_b log(W_{Kb})-log(W_K)$. Then the average $GK = 1/B\sum_b log(W_{Kb})$ and its standard deviation $sd_k=[1/B\sum_b (log(W_{kb})-GK)^2]^{1/2}$ are computed leading to $s_K=sd_K\sqrt{1+1/B}$. The estimate of K^* is the smallest K such that Gap(K)$\geqq$ Gap($K+1$)- s_{k+1} (Tibshirani et al. 2001).

The Jump Statistic (Sugar and James 2003) utilizes the criterion W in (1) extended according to the Gaussian distribution model. Specifically, the distance between an entity and centroid in (1) is calculated as $d(i, S_k)=(y_i-C_k)^T\Gamma_k^{-1}(y_i-C_k)$, where Γ_k is the within cluster covariance matrix. The jump is defined as $W_K^{-M/2}$ - $W_{K-1}^{-M/2}$, assuming that $W_0^{-M/2}\equiv 0$ and M is the number of dimensions. The maximum jump $JS(K)$ corresponds to the right number of clusters. This is supported with a mathematical derivation stating that if the data can be considered a standard sample from a mixture of Gaussian distributions at which distances between centroids are great enough, then the maximum jump would indeed occur at K equal to the number of Gaussian components in the mixture (Sugar and James 2003).

Hartigan (HT):

- calculate $HT=(W_K/W_{K+1} \quad -1)(N-K-1)$, where N is the number of entities
- increase K from $K=2$ and pick the very first K at which HT becomes less than 10 (The threshold 10 here is "a crude rule of thumb" Hartigan (1975), p. 91, based on the intuition that if K is less than the "right number" of clusters, then a ($K+1$)-cluster partition should be equal to a K-cluster partition with one of its clusters split in two.)

Calinski and Harabasz (CH):

- calculate $CH=((T-W_K)/(K-1))/(W_K/(N-K))$, where $T= \sum_{i\in I}\sum_{v\in V} y_{iv}^2$ is the data scatter
- find the K which maximises CH

Jump Statistic (JS):

- for each integer K, clustering $S=\{S_1,S_2,\ldots,S_K\}$, and centroids $C=\{c_1,c_2,\ldots c_K\}$
- for each $i\in I$ and $k=1,2, \ldots,K$, calculate $d(i, S_k)=(y_i-C_k)^T\Gamma^{-1}(y_i-C_k)$, where Γ is the within cluster covariance matrix
- select a transformation power, typically $M/2$, where M is the number of dimension
- calculate the jumps $JS= W_K^{-M/2} - W_{K-1}^{-M/2}$ assuming that $W_0^{-M/2}\equiv 0$
- find the K that maximises JS

Gap Statistic (GS):

- Cluster the observed data and obtain W_K for each K
- Generate B uniform random reference datasets over the range of the observed data and obtained W_k for each datasets, where $k=1,2,\ldots,K$
- Compute the estimated Gap statistic: $Gap(k)=1/B\sum_b \log(W_{kb})-\log(W_k)$
- Let $GK=1/B\sum_b \log(W_{kb})$, compute the standard deviation $sd_k=[1/B\sum_b (\log(W_{kb})-GK)^2]^{1/2}$, and define $s_k=sd_k\sqrt{1+1/B}$
- Find the smallest K such that Gap(K)$\geqq$Gap($K+1$)-s_{k+1}

2.2 Within-cluster cohesion vs. between-cluster separation

A number of approaches utilize indexes comparing within-cluster distances with between cluster distances: the grater the difference the better the fit; many of them are mentioned in Milligan and Cooper (1985). The experiments and indices in Milligan and Cooper (1985) have been widely applied to different research fields, for example, bioinformatics. Some of the indices are specifically suitable for hierarchical clustering, for example, Mojena’s upper tail rule (Mojena 1977), Duda and Hart’s (Duda and Hart 1973) error ratio test, Gamma index (Baker and Hubert 1975), etc and these are described in Section 2.4. Some of those indices are distribution or likelihood based, for example, cubic clustering criterion, likelihood ratio, etc, which are beyond the scope of this thesis, which is confined to K-Means related methods only.

The rest of them will be described briefly in the following paragraphs. A modified version of the Gamma index is so called G (+) index, and the formula is $(2*S_-)/(n_d(n_d-1))$, where S- is the number of times that a pair of entities not in the same cluster had a smaller separation than a pair that were in the same cluster and n_d is the number of within cluster distances. The minimum G (+) index indicates the number of cluster in the data. Davis and Bouldin (1979) proposed an index, that is, $DB = \frac{1}{K}\sum_{i=1}^{K}\max(\frac{\alpha_i + \alpha_j}{d(c_i, c_j)})$, where α_i and α_j is the average within cluster distance of cluster i and j and the denominator is the distance between centroids c_i and c_j. The minimum value of *DB* indicates the number of clusters. This index has been widely used in some application, for example, a bioinformatics toolbox for microarray data analysis (Bolshakova et al. 2005), experimental comparison in color image segmentation (Ray and Turi 1999), etc. Petrović (2006) compared the Silhouette Width index with the Davis-Bouldin index and the clustering results of the Silhouette Width index is more accurate than the Davis-Bouldin index although the Davis-Bouldin index is more computational efficient.

Another within and between cluster related index is proposed by McClain and Rao (1975), that is, the ratio of the average within cluster distance divided by the number of within cluster distances over the average between cluster distances divided by the number of between cluster distances. The minimum of the index indicates the number of clusters. The McClain and Rao index shows an extremely good result in Milligan and Cooper (1985). Dunn's (1974) index, which is based on the idea of classifying well-separated data, is not included in Milligan and Cooper (1985) but has been widely compared in some publications and applied on several different fields, for example, image analysis in Ray and Turi (1999), and Boutin F. and Hascoët M. (2004), gene expression data analysis in Bolshakova and Azuaje (2003). This index is the ratio of the minimum intra-cluster distance (distance between two objects from different clusters) over the maximum inter-cluster distance (distance between two objects from the same clusters) within the range of 0 to ∞. The maximum value of this index indicates the number of clusters.

Two of those indexes in Milligan and Cooper (1985) are (a) the point-biserial correlation, that is, the correlation coefficient between the entity-to-entity distance matrix and the binary partition matrix assigning each pair of the entities 1, if they belong to the same cluster and 0 otherwise, that is, $(D_k-D_{min})/(D_{max}-D_{min})$, where D_k is the sum of the within cluster dissimilarity for a partition and D_{max} and D_{min} are the maximum and minimum of D_k respectively and (b) its ordinal version, the C index proposed by Hubert and Levin (1976). These two indexes show a very good performance in Milligan and Cooper's tests. This, however, perhaps can be an artifact of the very special type of cluster structure utilized by Milligan and Cooper (1985): almost equal sizes of the generated clusters. Indeed, a mathematical investigation described in Mirkin (1996, pp. 254-257) shows that the point-biserial correlation expresses the so-called "uniform partitioning" criterion, which tends to produce equal-sized clusters.

There are several other recent publications using the indexes relating to within- and between-cluster distance, for example, Ray and Turi (1999) proposed a simple validity index, which is the ratio of the average of distances between an item and its cluster centroid over the minimum of the distance between the item to other clusters to obtain the optimal number of clusters in colour image segmentation application and the clustering which gives a minimum value for the validity measure will tell us what the ideal number of clusters is. A more recent effort is described in Shen et al. (2005), which proposed a dynamic validity index based on the validity index proposed by Ray and Turi (1999) and Dunn index (Dunn 1974) so that the distance between an item to its cluster centroid is minimized and the distance between the item to others clusters is maximized. The dynamic validity index is incorporated into K-Means algorithm for microarray data clustering. Bel Mufti et al. (2005) used Loevinger's measure for the cluster stability, that is, $t(A, X') = 1 - \frac{n'(n'-1)m_{X';A}}{2n'_A(n'-n'_A)m_{X'}}$, where *A* is a cluster in a partition, *X'* is a sample of original data, *n'* is the sample size, n'_A is the cluster size of cluster A, $m_{X';A}$ is the number of entities of the samples that are in the same cluster, $m_{X'}$ is the number of entities of the original data that are in the same clusters. The stability measure is the average of the sum of Loevinger's measure over a large number of samples.

A well-balanced coefficient, the Silhouette Width index, which has shown good performance in experiments (Pollard and van der Laan 2002), was proposed by Kaufman and Rousseeuw (1990). The concept of silhouette width involves the difference between the within-cluster tightness and separation from the rest. First, the silhouette width is calculated for each entity, then the average silhouette width for each cluster and then the overall average silhouette width for the total clustering. Specifically, the silhouette width *s(i)* for entity $i \in I$ is defined as:

$$\text{s(i)}=\frac{b(i)-a(i)}{\max(a(i),b(i))} \qquad (3)$$

where *a(i)* is the average dissimilarity between *i* and all other entities of the cluster to which *i* belongs and *b(i)* is the minimum of the average dissimilarities between *i* and all the entities in other clusters. The silhouette width values lie in the range from –1 to 1. If the silhouette width value is close to 1, it means that the set I is well clustered. If the silhouette width value for an entity is about zero, it means that that the entity could be assigned to another cluster as well. If the silhouette width value is close to –1, it means that the entity is misclassified.

The largest overall average silhouette width indicates the best number of clusters. Therefore, the number of clusters with the maximum overall average silhouette width is taken as the optimal number of the clusters. The usage of this index is described in the box below.

Silhouette width (SW)

Given *K*, take the best clustering of the *R* runs. For each $i \in I$:

- calculate *a(i)* = the average dissimilarity between i and all other entities of the cluster to which i belongs, *b(i)* = the minimum of the average dissimilarity between i and all the entities in other clusters, and *s(i)* according to (2).
- calculate $SW_K = average\ s(i)$
- find the *K* that maximizes SW_K

2.3 Consensus approach

The consensus approach relies on the entire set of all *R* clusterings produced at multiple runs of K-Means, given *K,* rather than just the best of them. The intuition is that the clusterings should be more similar to each other at the right *K*. Thus, a measure of similarity between clusterings should be introduced and utilized. We consider two such measures. One is the Consensus distribution area introduced by Monti et al. (2003). To define the latter, the consensus matrix is calculated first. The consensus matrix $C^{(K)}$ is an $N \times N$ matrix whose *(i,j)*-th entry is the proportion of those clustering runs in which the entities $i,j \in I$ are in the same cluster. An ideal situation is when the matrix contains 0's and 1's only: this is the case when all the *R* runs lead to the same clustering. The cumulative distribution function (CDF) of entries in the consensus matrix is defined as usual:

$$\text{CDF}(x) = \frac{\sum_{i<j} 1\{C^{(K)}(i,j) \le x\}}{N(N-1)/2} \qquad (4)$$

where 1{cond} denotes the indicator function that is equal to 1 when cond is true, and 0 otherwise. The area under the CDF corresponding to C(K) is calculated using the conventional formula:

$$\text{A}(K) = \sum_{i=2}^{m} (x_i - x_{i-1})\text{CDF}(x_i) \qquad (5)$$

where set {x1,x2,…,xm} is the sorted set of entries of C(K).

We suggest that the average distance between the R partitions can be utilized as another criterion: the smaller, the better. This equals avdis(K)=$\frac{1}{R^2}\sum_{u,w=1}^{R} M(S^u, S^w)$, where distance M is defined as squared Euclidean distance between binary matrices of partitions S^u and S^w. A binary partition matrix is an entity-to-entity similarity matrix; its (i,j)-th entry is 1 if i and j belong to the same cluster, and 0, otherwise, so that consensus matrix C(K) is the average of all R binary partition matrices. Denote the mean and the variance of matrix C(K) by μ_K and σ_K^2, respectively. Then the average distance can be expressed as avdis(K)= $\mu_K*(1-\mu_K) - \sigma_K^2$, which also shows how close C(K) is to being binary.

The average distance avdis(K)= $\frac{1}{R^2}\sum_{u,w=1}^{R} M(S^u,S^w) = \frac{1}{R^2}\sum_{u,w=1}^{R}\sum_{i,j\in I}(S_{ij}^u - S_{ij}^w)^2$. This can be rewritten as avdis(K)= $\frac{1}{R^2}\sum_{i,j\in I}\sum_{u,w=1}^{R} S_{ij}^u + S_{ij}^w - 2S_{ij}^u S_{ij}^w$. A measure was suggested in Monti et al. (2003) based on the average partition matrix which is an entity-to-entity similarity matrix defined by μ(i,j)=$\frac{\sum_{t=1}^{R} s_{ij}^t}{R*(i,j)}$, where s^t is the binary relation matrix corresponding to S^t and s^t=0, otherwise, with R*(i,j) denoting the number of partitions S^t at which both i and j are present. Therefore, we obtain avdis(K)=

$$\sum_{i,j\in I}(\sum_{w=1}^{R}\mu(i,j)/R + \sum_{u=1}^{R}\mu(i,j)/R - 2\mu(i,j)\mu(i,j)) = \ 2\sum_{i,j\in I}(\mu(i,j) - \mu(i,j)^2) \ .$$

The proof follows then from the definition of the variance of the matrix, q.e.d.

To estimate “the right number of clusters”, the relative change of the indexes is utilized. Specifically, the relative change in the CDF area in (4) is defined as

$$\Delta(K+1) = \begin{cases} A(K), & K = 1 \\ \dfrac{A(K+1) - A(K)}{A(K)}, & K \geq 2 \end{cases} \qquad (\mathbf{6})$$

Then *K* which maximises *Δ(K)* is determined. The average distance based index is defined similarly except that it increases rather than decreases with the growth of *K*, so that *DD(K)=(avdis(K) - avdis(K+1))/avdis(K+1)*. The number of clusters is decided by the maximum value of *DD(K).*

Corresponding algorithms are presented in the boxes below.

```
Consensus Distribution Area (CD):
For each K in its range:
    For di=1: R
        - calculate the connectivity matrix $M^{(di)}$ where $M^{(di)}(i,j)=1$ if i and j
          belong to the same cluster, and 0, otherwise
    end di
        - calculate the consensus matrix $C^{(K)}(i,j)=\sum_{di} M^{(di)}(i,j)/R$
        - determine the cumulative distribution function $CDF(x)$ (3) and the
          area $A(K)$ in (4)
        - calculate $\Delta(K+1)$ (5)
        - find K maximizing $\Delta(K)$
```

```
Average distance between partitions (DD)
 - For each $K$, calculate the mean $\mu_K$ and variance $\sigma_K^2$ of the consensus matrix $C^{(K)}$
 - Compute $avdis(K)= \mu_K * (1- \mu_K) - \sigma_K^2$
 - $DD(K)=(avdis(K)-avdis(K+1))/avdis(K+1)$
 - Find $K$ maximizing $DD(K)$
```

A slightly different approach relating the average distance/Rand measure and the entropy of the consensus distribution on real and artificial data sets has been utilized by Kuncheva and Vetrov (2005).

2.4 Hierarchical approach

A number of approaches rely on the hierarchy of clustering solutions found by consecutive merging of smaller clusters into larger ones (agglomerative clustering) or by splitting larger clusters into smaller ones (divisive clustering). Some approaches are based on the distribution of the value of criterion function, where the criterion function, for example, could be the ratio of the within-cluster similarity over the between-cluster similarity in Pedersen and Kulkarni (2006). Mojena's upper tail rule (Mojena 1977) is one of the well-known criterion function distribution based indexes, which is, $\alpha_{j+1}>\mu_\alpha+c\sigma_\alpha$, where μ_α and σ_α is the mean and standard deviation of the distribution of clustering criterion value. It finds the first biggest jump of the series of the clustering criterion values as the number of cluster, which is in the upper tail of the clustering criterion value distribution for hierarchical agglomerative clustering. If no such number can be found then there is only one cluster. This index shows the best performance in the experiments of Milligan and Cooper (1985).

However, more indices are focused on the within and between cluster distances, for example, the widely implemented Ward's method (Ward 1963), Gamma index (Baker and Hubert 1975), error ratio index (Duda and Hart 1973), etc. Many papers have shown that Ward's method outperforms others under the condition of less outliers and cluster overlaps (Aldenderfer and Blashfield 1984). Ward's method (Ward 1963) minimizes the summary within cluster distance of two clusters that formed at each merging step, the so-called Ward distance. Each of the merged clusters is the smallest increase of the total within-cluster summary distance to the merged centroids and it tends to find smaller number of clusters (Hair et al. 1995). The above mentioned Ward method is for agglomerative hierarchical clustering, and for the divisive clustering, one needs to find the maximum Ward distance because of the nature of divisive clustering, that is, to build the cluster structure from the entire data, top to bottom. A specific K-Means at $K=2$ clustering can combine with the Ward divisive clustering method and the combined method, and has been named as bisecting K-Means (Steinbach et al. 2000). It has been tested on document clustering, suggesting that the bisecting K-Means outperforms the Straight K-Means and agglomerative hierarchical methods because of the features of documents.

Another widely used index for hierarchical clustering is Gamma index (Baker and Hubert 1975), and the formula is, $\frac{S_+ - S_-}{S_+ + S_-}$, where S_+ is the number of times that a pair of entities not in the same cluster had a larger separation than a pair that were in the same cluster and S_- represents the reverse outcome. The maximum Gamma index indicates the best partition. This is quite similar to the structural approach -- silhouette width (Kaufman and Rousseeuw 1990). The difference is that the Gamma index is only for hierarchical clustering because this index is defined for the tree diagram, also termed dendrogram. Duda and Hart's error ratio (1973), that is, *Je(2)/Je(1)*, where *Je(2)* is the sum of squared within cluster distance when data split into two clusters and *Je(1)* is the summary within cluster distance if only one cluster is present. It evaluates the cluster and its subcluster by taking the ratio of the summary Euclidean distance to the cluster centroids over the summary Euclidean distance to the subcluster centroids and a pre-defined threshold is computed from the standard normal distribution. This index showed very good performance in the experiments proposed by Milligan and Cooper (1985) and can be applied to agglomerative or divisive clustering methods.

Frey and Van Groenewoud (1972) proposed an index, that is, the ratio between the differences between the between cluster distances and mean within cluster distances from two sub-clusters in a hierarchy. The very last *K* at which the index becomes above 1 is taken as the estimate of *K**. There is only one cluster when no index is below 1. In the experiments of Milligan and Cooper (1985), this index tends to find too many clusters.

More recently, a more advanced statistical index for choosing K, the Bayesian information criterion (BIC), is utilized typically for model selection. It is based on the posterior probability rather than the distance measures and requires three parameters: log likelihood of the data model (ln (L)), number of parameters in the data model (p) and number of entities (n) and the formula is $BIC= -2 \ln (L) + p \ln (n)$. Pelleg and Moore (2000) included the Bayesian information criterion (BIC) to their X-means algorithm to determine the number of clusters using a divisive approach. The X-means algorithm is as follows: run conventional K-Means as initialization, then for each cluster, its BIC score is computed, the partition of the highest BIC score is kept, and the algorithm stops when reaches a pre-specified threshold. They tested conventional K-Means and X-means on both real and synthetic data and found that X-means outperforms not only on performance but also on computational time. An extended version of X-means is proposed by Ishioka (2005). The author modified the divisive procedures and the results have shown the later version is better. The modification includes a 2-means divisive method, that is, non-recursive divisive function is applied to one of the two clusters after each division, that is, to divide one cluster until no further cluster can be found and then deal with another. This will save the function call time if the loop of division is deep.

Feng and Hamerly (2006) also proposed a 2-means divisive method, named PG-means (PG stands for projected Gaussian), to learn the number of clusters in data. This method randomly projects the data and model to one dimension, test the goodness for each model and a model is selected if it has been accepted by two tests shown in the paper. They compare PG-means with three other methods, including X-means, and the experimental results of PG-means are better than the other methods.

Some authors propose versions involving several techniques simultaneously. Casillas et al. (2003) utilize Minimum Spanning Tree (MST) with a genetic algorithm using a rather arbitrary stopping condition to arrive at a number of clusters. They compare the Calinski & Harabasz stopping rule (Calinski and Harabasz 1974) and the genetic algorithm on a document containing 14,000 news items and claim that if the real number of clusters is close to 2, the Calinski & Harabasz stopping rule (Calinski and Harabasz 1974) performs better than the genetic algorithm, and otherwise, the genetic algorithm is better. Chae et al. (2006) proposed a method which applied six different agglomerative clustering algorithms and four different validity measures for comparing the partitions to the generated data and five of the six methods to real-world data from a beer consumer report in USA. The number of clusters at which these partitions are most similar is selected. This approach obviously can be counted as belonging to the consensus framework because they are based on the similarity measure on two partitions.

2.5 Resampling approach

Resampling means using many randomly generated copies of the data for assessing statistical properties of a utilized method (see, for instance, Mirkin 2005). This approach can be grouped into 4 main types: (a) random sub-samples of the data set;

(b) random splits of the data set into “training” and “testing” subsets, (c) bootstrapping, that is, randomly sampling entities with replacement, usually to their original numbers, and (d) adding random noise to the data entries. The intuition is that different random copies lead to more similar results at the right number of clusters, for example, Levine and Domany (2001), Bel Mufti et al. (2005), Minaei-Bidgoli et al. (2004) for type (a), Dudoit and Fridland (2002) for type (b), McLachlan and Khan (2004) and Wishart (2004) for type (c), and Kerr and Churchill (2001) and Möller and Radke (2006) for type (d). Each type is explained briefly in the remainder of the section.

(a) Subsampling

Levine and Domany (2001) proposed a resampling procedure based on the consensus matrix, which is described in Section 2.3. The samples are obtained by selecting fN size of the original data randomly, where f is named as dilution factor between 0 and 1 and N is the total number of entities. A clustering algorithm with pre-specified parameters is applied to those samples and the consensus matrices of these partitions are calculated. By comparing these consensus matrices with the consensus matrix of the original data, a figure of merit measure is calculated. The parameters of the clustering algorithm are then changed and the whole process run again until the local maximum of the measure is found. Once the optimal parameters of the clustering algorithm are found, the stable partition is found. Bel Mufti et al. (2005) named the similar sampling procedure proportionate stratified sampling, which selects the number of elements randomly without replacement proportional to the number of elements in each cluster of partition obtained from the original data. This proportion has to be chosen between 0.7 and 0.9 based on experimental analysis. If all partitions obtained from these samples are close in structure to the partition of the original data P, the partition P is claimed as stable. For each *K*, the Loevinger's measure is calculated and the maximum of these indices is taken as the number of the clusters.

Minaei-Bidgoli et al. (2004) proposed a clustering ensemble algorithm, which generates subsamples of the data and obtains partitions by running K-Means clustering algorithm on each of the subsamples. A new partition of the original data is to combine the partition of each subsample so that the entities in the partition of the original data are more similar in same clusters than in different clusters and in order to achieve this, one needs to calculate the consensus matrix, that is, the similarity measure between entities. Monti et al. (2003) also use the subsampling procedure to resample the data, but the way they obtained the partitions is via the consensus distribution area, described in Section 2.3. The subsample size proposed in Minaei-Bidgoli et al. (2004) is within an interval utilized by the total number of entities and Monti et al. (2003) generate the sample from 80% of the original data. The authors of both publications compare bootstrapping and subsampling methods and both methods show similar results but prefer subsampling because of the computer complexity and the possibility of the result inflation of bootstrapping. Mitra et al. (2002) proposed a density-based multi scale data condensation (DBMSDC) algorithm for data subsampling based on a density criterion. Instead of a rather arbitrary subsampling size, this subsampling algorithm is to first pre-specify *K*, and then calculates the distance of each entity of the original data using K-nearest neighbor method. The next two steps are iterated until the original data set is empty: select the entities that have the lowest distance and remove the entities in the original data sets that lie within a disc of radius of two times of centroids of the selected entities. This algorithm along with six other subsampling methods including random sampling has been tested on some well-known real-world data and it has been found that this subsampling method is superior to others. Some publications use subsampling as an initialization of the clustering algorithm, for example, the mixture likelihood approach proposed by Rocke and Dai (2003) and others use subsampling for identifying the tight and stable clusters in data, for example, a sequential approach proposed by Tseng and Wong (2003).

(b) Random splitting

Dudoit and Fridland (2002) proposed a popular procedure named Clest, following the pioneering work by Breckenridge (1989). This method has been tested on both the generated data and four microarray datasets. For each *K*, a number *B* of the following operations is performed: the set is split into non-overlapping training and testing sets, after which the training part is partitioned into *K* parts; then a classifier is trained on the training set clusters and applied for predicting clusters on the testing set entities. The predicted partition of the testing set is compared with that found, with the same procedure, on the testing set. The result of these *B* iterations is the median value t(K) of the index of similarity measure between two partitions of the testing set, that predicted from the training set and that found directly. The reason for using median instead of mean is not stated in Dudoit and Fridland (2002): probably because the median is more robust in the presence of outliers than the mean. After that a number of data sets of the same size is generated randomly and the same procedure applies to each of them producing the average value of the index *t'(K)* under the null hypothesis. The estimated *K* is that maximizing the difference *t(K)-t'(K)* under some additional conditions. This procedure, as well as other resampling schemes, involves a number of important parameters such as the type of classifier (taken to be the linear discriminant analysis with the diagonal covariance matrix in Dudoit and Fridlyand 2002), the training-testing split proportion (taken to be 2:1), numbers of iterations and reference sets generated (taken to be 20), the threshold on *K* values (taken to be 5 or 10), the similarity between partitions index, etc. On the same data generating mechanisms, the approach was outperformed by a model-based statistic as reported by McLachlan and Khan (2004).

(c) Bootstrapping

Bootstrapping is one of the most popular resampling approaches in machine learning. One of its advantages is that the number of items of generated samples is the same as the original data. The identical replicated samples are generated *n* times by replacement from the original data, so the clustering algorithm might claim those *n* entities as a cluster, which are actually *n* replicates of the same item. Some authors prefer other resampling approach, e.g. subsampling, for determining the number of clusters. The bootstrapping method proposed by McLachlan and Khan (2004) is to generate samples under the null hypothesis of K_1 clusters from the parametric mixture model with unknown parameters replaced by its maximum log-likelihood (*log L*) estimate from the original data. The hypothesis set is H_0*:* K=K_1 vs. the alternative hypothesis H_1*:* K=K_2 (K_2>K_1). The likelihood ratio test statistic *-2 log* λ is computed for each sample after fitting the mixture model for K_1 and K_2 clusters and this process is iterated several times. The number of clusters is determined whether the number of clusters is the null hypothesis or not. Wishart (2005) proposed a bootstrap validation method which compares dendrogram, and searches for the partition that manifests the greatest departures from randomness. The dendrogram obtained from the original data is compared with the dendrograms obtained from the sampling data in order to find the biggest departures from randomness.

(d) Adding noise to the data

Kerr and Churchill (2001) proposed a sampling method combining the bootstrapping and adding noise. They first fit the data to a linear model, found parameters and residue, and then obtained the bootstrapping data by randomly sampling with replacement among those parameters and residue using the same linear model. This sampling method is applied to gene expression data and the clustering method they apply is based on correlations between genes, that is, data with high correlations form clusters. A comparison of resampling methods is proposed by Möller and Radke (2006), which apply the subsampling, bootstrapping and adding noise on three gene expression data and four well-known real-world data and found the adding noise resampling method outperforms others. The subsampling rate taken as 80%, within the range of 70%-90%, of the original data, coincides with other published subsampling rates, for example, Monti et al. 2003. They add 1%, 5%, and 10% of the original data set with the same size to be noise and found 10% is the best rate among three experimentally.

In general, the procedure of the resampling approach is as follows: generating copies related to the original data, running a suitable algorithm, for example, K-Means, evaluating and merging the clustering results from the original data and copies. The clustering algorithm is done in the same way as it was on the original data, except for the case of random splitting, that is; the algorithm is only applied to the training sets. This difference is applied to the evaluation procedure, that is, the partition of the training sets are compared with the testing sets while others are compared with the original data.

Most of the publications use similarity measure to compare the partition obtained from the original data and copies, for example, the subsampling case in Minaei-Bidgoli et al. (2004), but the similarity measure is specifically suitable for the splitting case because the two partitions obtained from the testing set, that predicted from the training set and that found directly by applying algorithm is closer the better. For other copies-generating cases, one can use any validation index, for example, the Rand index, described in Section 3.4. McLachlan and Khan (2004) and Wishart (2004) both use test statistic for evaluating the performance between the original data and copies and Levine and Domany (2001) choose average overlap index for evaluation, that is, $\frac{\sum_{k=1}^{K}\sum_{j=1}^{J} N_{kj}^{2} - N}{2}$, where N_{kj} is the co-occurant counts in a confusion table described in Section 3.4 and N is the total number of entities.

The widely applied method to merge the partition results of the original data and copies is to average the evaluation results for each copy and the average evaluated result can be taken as a result of the algorithm (Diday 1971, Diday et al.

1979). Therefore, one can select the best algorithm based on the testing results of algorithms, for example, the method proposed by Levine and Domany (2001). However, Levine and Domany (2001) use the testing result for selecting the parameters within the same algorithm, for example, the number of clusters. Other ways of merging the results is to average and combine the mode$_{L2}$ if the models have same and different formats respectively. If, for example, the hierarchical cluster structures have the same format, these can be averaged into a similar structure with the clusters that are found in most of the structures (Margush and McMorris 1981). If these structures have different formats, one can combine these structures to make a joint structure.

2.6 Summary

K-Means is arguably the most intuitive, computationally easy and the most commonly used clustering method and this is why studying its properties is of interest not only to the classification, data mining and machine learning communities, but also to the increasing numbers of practitioners in marketing research, bioinformatics, customer management, engineering and other application areas. Five different approaches to estimating the “right” number of clusters *K** in K-Means are described in this chapter. Clearly, different clustering methods and criterion for choosing K can suggest different results when applied to the same data sets. The best way for determining the number of clusters is to use several clustering techniques and to analyse all the results in order to have a clearer picture of the data.

Chapter 3

Experiment Setting for Comparison of Methods for Choosing K

The data for experimental comparisons can be taken from real-world applications or generated artificially. In the published literatures, several clustering experiments conducted over real-world data sets only, for example, Casillas et al. (2003) apply the document clustering on a Spanish newspaper with 14,000 news items, Minael-Bidgoli et al (2005) apply the resampling method on five famous datasets, such as Iris, Wine, and etc, Shen et al. (2005) apply the dynamic validity index on the microarray data, and etc. More publications only focus on generated data, for instance, Hand and Krzhanowski (2005), Hardy (2005), Ishioka (2005), Milligan and Cooper (1985), Steinley and Brusco (2007), and etc. Some publications use both the generated data and the real-world data, for example, Chae et al. 2006, Dudoit and Fridland (2002), Feng and Hamerly (2005), Kuncheva and Vetrov (2005), Maulik and Bandyopadhyay (2000) etc. For our K-Means clustering experiments, we consider generated data only, to allow us to control the parameters of the experiments. Having the set of parameter values specified, we generate a number of datasets so that the results reported further on are averaged over these datasets. Initially we generated 20 random datasets for each parameter setting (as did Dudoit and Fridlyand 2002) – these are reflected in Tables 4.1 and 4.2, but then for the sake of time, we reduced the number of generated datasets to 10 (in Tables 4.3, 4.4 and 4.5). The following issues are to be decided upon before a data generator is set:

(A) Data sizes,

(B) Cluster sizes,

(C) Cluster shapes,

(D) Cluster intermix, and

(E) Data standardization.

These are described in Section 3.1.

3.1 Modelling cluster structure

A. Data sizes. First of all, the quantitative parameters of the generated data and cluster structure are specified: the number of entities *N*, the number of generated clusters *K**, and the number of variables *M*. In most publications, these are kept relatively small: *N* ranges from about 50 to 200, *M* is in many cases 2 and, anyway, not greater than 10, and *K** is of the order of 3, 4 or 5 (see, for example, Casillas et al. 2003, Chae et al. 2006, Hand and Krzanowski 2005, Hardy 1996, Kuncheva and Petrov 2005, McLachlan and Khan 2004, Milligan and Cooper 1985). Larger sizes appear in Feng and Hamerly (2006) (*N*= 4000, *M* is up to 16 and *K**=20) and Steinley and Brusco (2007) (*N* is up to 5000, *M*=25, 50 and 125, and *K** =5, 10, 20). Our choice of these parameters is based on the idea that the data should imitate the conditions of real-world data analysis, under the timing constraints of the computational capacity. That means than *N* should be in thousands while limiting *M* within one or two dozens, to mimic the situation in which the data analysts select only features relevant to the problem at hand (“tall” data table cases) rather than using all features or key words available ("wide" data table case); the latter should be treated in a different experiment. Another consideration taken into account is that, according to our real-world clustering experiences, it is not the absolute values of M and K^* but rather their ratios, the average cluster sizes, that affect the clustering results. As the major focus of our experiment is the effects of within and between cluster spreads on the clustering results, we decided to keep the ratio restricted, while maintaining two rather distinct values of K^*. Therefore, two settings for the sizes are: (i) N=1000, M=15, K^*=7 and 9 – about 110 entities in a cluster on average, and (ii) N=3000, M=20, K^*=21 – about 145 entities in a cluster on average. These are obviously at the upper end of the sizes in the published reports (Casillas et al. 2003, Chae et al. 2006, Hand and Krzanowski 2005, Hardy 1996, Kuncheva and Petrov 2005, McLachlan and Khan 2004, Milligan and Cooper 1985).

It is probably worth mentioning that we do not consider the so-called irrelevant, or noisy, features: The presence of features that have nothing to do with the cluster structure was considered by Milligan and Cooper (1985); see also Dudoit and Fridlyand (2002) and Kuncheva and Vetrova (2005). K-Means partitioning can be and has been applied when no visible cluster structure is present, just to dissect the domain into manageable chunks as advocated by Späth 1985, among the others and a similar goal has been pursued by the so-called vector quantization (VQ) and learning vector quantization (LVQ) (Lloyd 1982 and Pollard 1982), the concepts that, basically, resemble the cluster centroids in K-Means. The issue of noisy features, in this perspective, deserves a separate consideration.

B. Cluster sizes. The term “size” is ambiguous in the clustering context, because it may refer to both the number of entities and spatial volume taken by a cluster. We use it here for the number only, in accordance with the practice of Computer Sciences, while utilizing the term “spread” for the geometric size. (Steinley and Brusco 2007 term the cluster size as the “cluster density” – we prefer to utilize this regarding a probabilistic density function.) The difference in cluster sizes can affect the outcome of a clustering process if it is driven by a criterion, such as the point-biserial correlation, that depends on them in a non-linear way. As mentioned in section 3.2, this may have affected some of experimental results in Milligan and Cooper (1985) because of the relatively equal cluster sizes utilized by them. However, criterion (1) (see page 2) always involves the same number N of distances, whichever cluster sizes these are, so that cluster sizes should not much matter. Steinley and Brusco (2007), who maintained three different patterns for cluster size distributions, report no differences in their results regarding the patterns. Therefore, we decided to disregard this aspect of the cluster structure: our generated clusters have uniformly random size distributions. To generate a random distribution of the cluster size proportions $\boldsymbol{p}=(p_1,\dots,p_{K*})$ under the condition that elements of $\boldsymbol{p}$ are positive and sum up to 1, one can randomly generate $K*-1$ real numbers $r_1, r_2, \dots, r_{K*-1}$ in the interval (0,1), sort them in the ascending order so that $r_1 < r_2 < \dots < r_{K*-1}$, set $r_0=0$ and $r_{K*}=1$, after which the uniformly random proportions are computed as $p_k = r_k - r_{k-1}$ $(k=1,\dots,K*)$.

C. Cluster shapes. This property is not typically taken into account as a variable to control, because K-Means is conventionally seen as a method for fitting the

Gaussian mixture model with spherical Gaussians – and this, in fact, is a property which is directly associated with the Minimum distance rule. However, in real-world applications clusters may have more complex and elongated shapes, which can be, to an extent, be caught by the ellipsoidal shape of the Gaussian clusters (see also McLachlan and Khan 2004, p. 92). Thus, we generate data entities in each cluster by independently sampling from a Gaussian distribution. We take the conventional spherical shape of Gaussian clusters versus another one, much more elongated. Since the number of parameters needed to define the covariance matrix of a Gaussian distribution is in hundreds for our size settings, we utilize a version of the covariance matrix defined with a smaller number of control variables in a MatLab toolbox NetLab (see Generation of Gaussian mixture distributed data 2006). According to the so-called Probabilistic Principal Component Analysis (PPCA) model (Tipping and Bishop 1999), the $M\times M$ covariance matrix of a Gaussian distribution in this toolbox is defined by selecting the hidden dimension q as:

$$Cov(\sigma)=W_q*W_q^{'}+\sigma^2 I_{M\times M} \tag{7}$$

where $W_q=\begin{pmatrix} I_{q\times q} \\ 1_{(M-q)\times q} \end{pmatrix}$, $\boldsymbol{I_{n\times n}}$ is an $n\times n$ identity matrix, and $\mathbf{1_{n\times m}}$ a $n\times M$ matrix whose all entries are equal to 1. The PPCA model runs with the manifest number of features M and the hidden dimension q. The hidden factor structure is also advocated in Maclachlan and Peel (2000).

It is easy to show that $Cov(0)=\begin{pmatrix} I_{q\times q} & 1_{q\times(M-q)} \\ 1_{(M-q)\times q} & q1_{(M-q)\times(M-q)} \end{pmatrix}$. Obviously, the eigen-values of *Cov(σ)* are the same as those of *Cov(0)* with σ^2 added to each; the eigen vectors are the same as well.

The structure of eigenvalues of *Cov(0)* has been investigated by Wasito and Mirkin (2006) who found that, of q nonzero eigenvalues, the maximal one is $\lambda=1+(M-q)q$ whereas all the other $q-1$ eigen-values are equal to unity. In order to prove the eigenvalues of *Cov(0)*, let us consider an M-dimensional vector x in the form $x=(x_q, x_{M-q})$ where x_q and x_{M-q} denote subvectors with q and $M-q$ components, respectively. Also denote the sum of elements of x_q by a and the sum of elements of x_{M-q} by b. Obviously, to be an eigenvector of *Cov(0)* corresponding to its eigenvalue λ, x must satisfy the following equations: $x_q+b\mathbf{1}_q=\lambda x_q$ and $(a+qb)\mathbf{1}_{m-q}=\lambda x_{M-q}$. Summing up components of these vector equations leads to (i) $a+bq=\lambda a$ and (ii) $(a+bq)(M-q)= \lambda b$, respectively. Let us see first that $a = 0$ implies $b = 0$ and $\lambda = 1$. Having put $a = 0$ into (i) one obviously gets $b = 0$ as well. This implies that $a + bq = 0$ so that $(a + bq)\mathbf{1}_{m-q} = \lambda x_{M-q}$ can hold only at $x_{M-q}=0$, provided that $\lambda \neq 0$. Similarly, $x_q +b\mathbf{1}_q =\lambda x_q$ can hold only if $x_q = \lambda x_q$, that is, if $\lambda = 1$, which proves that $\lambda = 1$ is an eigenvalue. Moreover, the rank of the subspace of eigenvectors corresponding to $\lambda = 1$ is equal to $q - 1$, because they all are defined by the condition that the sum of their components $a = 0$.

Let us now assume that a is not zero. Eq. (i) implies that λa can be put for $a + qb$ in (ii), leading to $\lambda a(M - q) = \lambda b$. Thus, with $\lambda \neq 0$, $a(M - q) = b$ and $b/a = M - q$. But $\lambda = 1 + qb/a$ according to (i), which leads to $\lambda = 1 + q(M - q)$ and proves the statement. This provides for really elongated shapes, so that we can check whether this change of the shape indeed affects the clustering results.

The actual data generation process is based on the spectral decomposition of matrix *Cov(0)* such as described in Murtagh and Raftery (1984) and Fraley and Raftery (2002). In our experiments q is set to be 6. The variance σ^2 is taken to be 0.1, which is not very important because, in any case, it is multiplied by the within-cluster spread values described in the following item **D**.

Therefore, the generic PPCA covariance matrix generated is defined by formula (7) with q=6 and σ^2=0.1. The generic covariance matrix of the Spherical Gaussian distribution is taken to be the identity matrix. These are multiplied then by different values to model different versions of the distribution of cluster spatial volumes.

D. Clusters intermix. The possibility of controlling cluster intermix is a much-desired property in clustering experiments. Steinley and Henson (2005) noted that this issue had never been satisfactorily addressed in the literature and proposed a mechanism for generating clusters with an explicitly formalized degree of overlap, i.e. set-theoretic intersection. Specifically, their model involves a value of the intersection for each pair of clusters over each single feature, thus having a disadvantage of "restricting the generation of the joint distribution clusters to be the product of the marginal distributions" (Steinley and Henson 2005, p. 245). Another problem with this mechanism is by far too many parameters which are not necessarily directly related to parameters of the generated clusters themselves. There is also an issue of how relevant is the usage of overlapping clusters for evaluation of a partitioning method. We consider that the cluster overlap should be modelled as the spatial intermix rather than intersection, for which parameters of distributions used for modelling individual clusters are convenient to use.

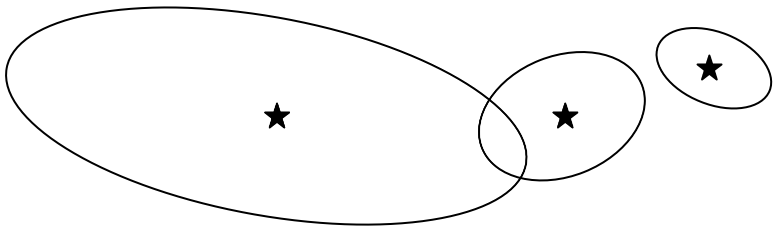

Figure 3.1 An illustration of the cluster intermix depending on the distance between cluster centroids (represented by pentagrams), and their geometric sizes (represented by ellipses): two clusters on the right are close to each other but well separated, whereas the cluster on the left is further away but not separated because of its larger spread.

Since we utilize Gaussian clusters, their intermix are modelled by using the Gaussian characteristics of location, centres, and cluster shape and spread, covariance matrices. In this way, the intermix among Gaussian clusters can be captured as a consequence of the two not necessarily related aspects: the distance between cluster centroids (“between-cluster spread”) and the magnitude of their variance/covariance values (“within-cluster spread”), as illustrated in Figure 3.1, at which the centers of two clusters are close to each other (a small between-cluster spread) but are well separated because of small (co)variances, while another cluster, with its center being much further away, may intermix with either or both of them, because of its large (co)variances.

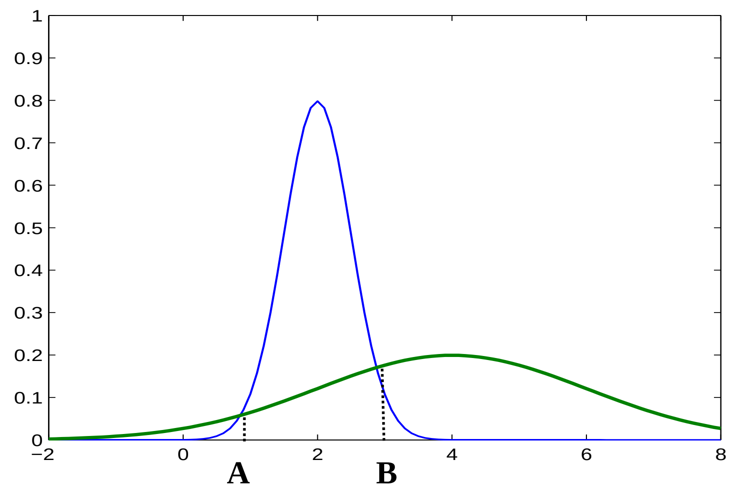

Figure 3.2 Two Gaussian clusters with their density functions drawn using a green and blue line respectively. The interval (A,B) is the only place at which the blue line cluster is more likely than the green line cluster.

Yet Figure 3.1 may introduce some perception bias too, by representing Gaussian clusters as ellipses. When dealing with different within-cluster variances, the perception of Gaussian clusters as being "compact" can be misleading, to an extent. Consider, for example, densities of two one-dimensional Gaussian clusters drawn in Figure 3.2. One, on the left, is centered at 2 with its standard deviation equal to 0.5, the other on the right is centered at 4 and has its standard deviation equal to 2. The clusters are well intermixed, but the cluster on the right is spread not only over the right part, but over the left as well – its density function is greater than that of the left cluster in all points to the left of A in Figure 3.2. This contradicts the compact cluster intuition. This is why, in the setting of cluster generation from probabilistic distributions, we prefer the term intermix rather than overlap.

To control the within-cluster spread, one can multiply the cluster's covariance matrix by a value. The number of these values is equal to the number of generated clusters K*. To keep things simple, one should try to define such a distribution of the within-cluster spreads that can be controlled by a single parameter. One obvious definition comes from the model of spherical clusters – all the spreads are equal to each other, that is, all clusters are represented by spheres with a constant radius. This pattern fits well into the theoretical perspective of K-Means as a maximum likelihood method for fitting a Gaussian distribution mixture model in which all individual distributions are spherical with the same variance (Banfield and Raftery 1993). However, within the data-mining framework, clusters to be found may have different spatial sizes. To fit into this perspective, one may use different settings such as several, two or three or four, different within-cluster spread values – which would lead then to the task of defining the proportions for each of these types, for which we could find no guidance in the literature or our personal experiences. Therefore, we decided to go along a less challenging path by designing two types of the variant within-cluster spread values: the "linear" and "quadratic" ones. Specifically, we take the within-cluster spread value to be proportional to the cluster's index k (the linear, or k-proportional distribution) or k^2 (the quadratic, or k^2-proportional distribution), $k=1, 2, \ldots, K^*$. That is, with the variable within-cluster spreads, the greater the generated cluster index, the greater its spatial size. For example, the within cluster-spread of cluster 7 will be greater than that of cluster 1, by the value of 7 in k-proportional model and by the value of 49 in k^2-proportional model. Since the clusters are generated independently, the within-cluster spread values can be considered as assigned to clusters randomly. Hence, three different models for the within-cluster spread values utilized in our experiments are: (i) constant, (ii) k-proportional, and (iii) k^2-proportional.

To control the distance between clusters with a single parameter, we utilize a special two-step mechanism for the generation of cluster locations. On the first step, all cluster centroids are generated randomly around the origin, so that each centroid entry is independently sampled from a normal distribution *N(0,1)* with the mean 0 and standard deviation 1. On the second step, each of these centroids is shifted away from 0, and from the others, along the line passing through the centroid and space origin, by multiplying it with a positive value: the greater the value, the greater the shift, and the greater the distances between centroids.

The cluster shift value is taken the same for all centroids. In our experiments, we consider two types of the between-cluster spread, “large” and “small” ones. These should be defined in such a way that the clustering algorithms recover the generated clusters well at the large spreads, and less than well at the small spreads. This idea has been implemented experimentally as follows: given the within-cluster spread and shape, put the between-cluster spread value at such a value that the generated clusters are recovered on average on the level of 0.95 of the ARI index of cluster recovery, which is defined by equation (8) in Section 3.4. This value is accepted then as the “large” between-cluster spread value. For a “small” between-cluster spread value, we have chosen a smaller value, such that the best cluster recovery achieved reaches ARI index value of about 0.4. Thus chosen between-cluster spread values at different within-cluster spread and shape models are presented in Table 3.1.

Typical configurations of datasets with K*=9 clusters generated as explained above are illustrated in Figure 3.3. These are just two-dimensional projections of multidimensional spreads, thus hiding many of their spatial interactions, but still bearing some of them and shown here for purely illustrative purposes.

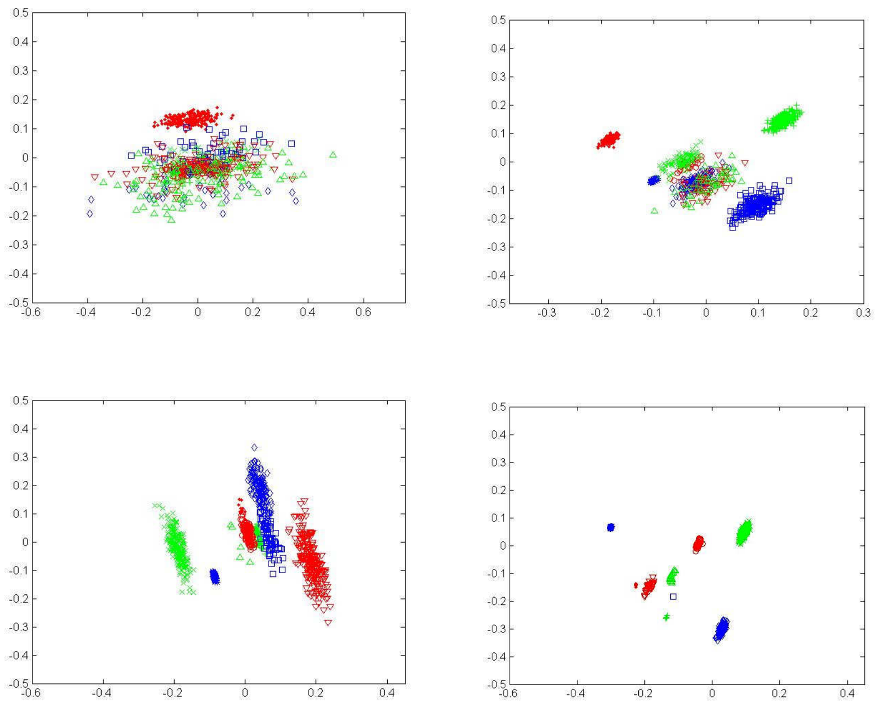
Figure 3.3 Examples of datasets generated at different data models on a plane defined by the two largest principal components, from the most confusing pattern on the left (PPCA clusters with the quadratic within-cluster spread and the between-cluster spread value equal to 2) to a clear-cut pattern on the right (the same cluster model, but the between-cluster spread value grows to 28). The nine clusters are shown with symbols: *,., +, o, x, □, ❋, ▽, △.

Between-cluster spread	Within-cluster spread type		
	Constant	k-proportional	k^2-proportional
Large	1.6	8	8
Small	0.16	0.4	1.6

Table 3.1 Between-cluster spread values depending on the within-cluster spread-shape types in the experiments

E. Feature standardization: In many publications, starting from Milligan and Cooper (1985), the data are generated in such a way that features are comparable and no data standardization is needed, which is very far from the real case scenario. In real-world data, features are usually incomparable so that some form of data standardization is needed. Conventionally, data standardization is conducted as an independent transformation of each individual feature by shifting its origin with the rescaling either using the standard deviation or the range.

In statistics, the most popular standardization is the so-called z-scoring which shifts the origin of each feature to its grand mean and then rescales the feature into the units of its standard deviation. This standardization is rooted in the invariance properties of the one-dimensional Gaussian distribution. In the neural network and support vector machine learning literature, the standardization is conventionally performed in a distribution-free way – by shifting the origin to the midrange and relating the result to the half-range so that the boundary values become -1 and +1, which is very convenient for working with target features that tend to have a range between –1 and 1. (Vapnik 2006).

Published clustering experiments have demonstrated that the mixed standardization in which the origin is shifted to the grand mean and rescaled using the range is better for cluster recovery than that by the standard deviation, for example, in Milligan and Cooper 1988, Steinley 2004, Vesanto 2001, etc. We can contribute to the debate with the following argument. Dividing the feature scale over the standard deviation is counter-intuitive in the following example that involves two features of the same ranges, so that one of them is uni-modal and the other is bi-modal, as shown on Figure 3.4, (a) and (b), respectively. The standard deviation of the former is much smaller than that of the latter so that after dividing by the standard deviations the uni-modal feature's range and, thus, contribution to the distances, will be by far greater than that of the multimodal feature. But intuition tells us that it is rather the bi-modal feature which is more useful for clustering, because the two modes lead to natural subgroups while the uni-modal feature tends to put all, except for the outliers, into the same group.

Figure 3.4 Uni-modal distribution shape on (a) versus a bi-modal distribution shape on (b): the standard deviation of the latter is greater, thus making the latter less significant under the z-scoring standardization, which is odd in the clustering context.

Milligan and Cooper (1988) compare seven standardization methods and found out that range normalization is the best standardization among those (see also a review in Milligan and Cooper 1987). The experiments of Steinley (2004) also support this experimental finding and suggest that normalized by maximum of the data performs quite well (see also a review in Steinley 2006). Vesanto (2001) compare only the range normalization and z-scoring and suggest that the range normalization performs better than z-scoring.

The standardization issue addressed above explicitly relates to established statistics concepts when using mixed scale data (Mirkin 2005), that is, the data table contains quantitative, nominal and categorical features. By doing the data standardization, there are not many constant effects on the data scatter and the feature contributions to the data scatter. The mixed standardization is adopted in our experiments

3.2 Selection of algorithms

Five different approaches to estimating the "right" number of clusters *K** in K-Means are described in the previous section: (i) Variance based, (ii) Structural, (ii) Consensus distribution, (iv) Hierarchical, and (v) Resampling. Of these, we take only three, (i), (ii), and (iii), for our experiments. Each of the other two approaches, both (iv) Hierarchical and (v) Resampling, involves too many diverse parameters that are absent from the variance based, structural based and consensus distribution based approaches. Since the thesis is confined to K-Means related clustering methods only, the hierarchical methods are beyond the scope. The resampling methods involve many parameters, for example, the type of classifier, the training-testing split proportion, number of iterations, reference sets generated, the threshold value on K, etc, and the choices for these parameters are not well-defined or well-specified. As the (i) Variance based approach relates to the criterion of K-Means and has received most theoretical support (Krzanowski and Lai 1985, Sugar and James 2003 Tibshirani et al. 2001), we take all four procedures referred to in section 2.1 – Hartigan's "rule of thumb", Calinski and Harabash criterion, Gap statistic and Jump statistic. We also take in the Silhouette width statistic, as the most versatile procedure, from (ii) Structural approaches, and two procedures from the (iii) Consensus distribution approach. Table 3.2 presents the selection of *K** estimating methods that participate in our experiments, along with their acronyms used in the remainder of the thesis**.**

Method	Acronym
Calinski and Harabasz index	CH
Hartigan rule	HT
Gap statistic	GS
Jump statistic	JS
Silhouette width	SW
Consensus distribution area	CD
Average distance between partitions	DD
Square error iK-Means	L_2
Absolute error iK-Means	L_1

Table 3.2 Set of methods for estimation of the number of clusters in K-Means under comparison

It is probably worth noting that almost all the methods utilize Euclidean square distance throughout, except for two cases: (a) a version of intelligent K-Means L_1 is based on Manhattan metric, and (b) the Jump-statistic utilizes Mahalanobis distance within clusters.

The seven methods from the three selected approaches utilize the same format of computations: they run K-Means at different *K* and then choose "the best" fitting value among the Ks as the estimate of *K**. Thus, we need to specify the range of *K* values for the experiments. Since the data is generated many times for each of the chosen values *K**=7 and 9 and *K**=21, and the between-cluster spread values are large enough to have several of the clusters well separated, we decided, to keep the computations within a reasonable time limit, that the range of tested *K* values should be within an interval of about a dozen with *K** in the middle; thus, the range of tested *K* values is from 4 to 14 at *K**=7 and 9 and from 15 to 25 at *K**=21.

As is well known, the clustering and criterion value produced by K-Means depends on the initialization. The user typically does not have a clear implication about the initial centroids. Several attempts and evaluations have been reported to solve the cluster initialization problem. Babu and Murty (1993) published a near-optimal centroid selection method using genetic programming and the fitness of each centroid selection is assessed by running the K-Means algorithm until convergence and then calculating the distance measures. The fitness solutions will then reproduce to create a second generation of solutions and this process is repeated until a predetermined number of generations have been created. Given if the optimum solution in many cases can be found, however, it becomes infeasible in a large database due to the need for repeated runs of the K-Means algorithm. Thiesson et al. (1997) suggested a rather simple idea: taking the mean of the entire dataset and randomly perturbing it *K* times to produce K centroids. Khan and Ahmad (2004) proposed a cluster center initialization algorithm (CCIA) under the assumption of Gaussian distributed features, which first generates initial clusters for each feature using Euclidean distance between feature values based on the mean, standard deviation, and the percentile of the feature and the entities in that feature and then runs the K-Means algorithm on each feature and the whole data set. The percentile is obtained based on the equal area under the partitions of the Gaussian curve of features. They treated the partitions obtained from each feature as a sampling result; therefore they applied the DBMSDC sampling algorithm to merge these partitions, described in Section 2.5.

Comparisons among several different initialization methods also have been proposed. Pena et al. (1999) presented a comparative study for different initialization methods for the K-Means algorithm and the results of their experiments illustrate that the random and the Kaufman approach (Kaufman and Rousseeuw 1990, 1999) outperforms the rest of the compared methods as they make the K-Means more effective and more independent on initial clustering and on instance order. Steinley and Brusco (2007) evaluated 12 different initializing K-Means options and found that Ward’s (1963) hierarchical cluster analysis suggested by Milligan (1980) performs the best followed closely by a multiple random initialization strategy. The multiple random initialization strategy is highly recommended for most of the situations, except when the size of the data set, the number of variables, or the number of clusters are too large to estimate the distribution of the solution. Kaufman and Rousseeuw (1999) suggested that the first centroid locates on the most central point of the whole data set and then, which of the points in the databases and which when chosen as the next centroid will produce the greatest reduction in the distance measures are examined. Once the second centroid is chosen, the third centroid is selected in the same way and continues until *K* centroids are chosen. If this algorithm is to be considered useful for large databases, a sub-sample of the instances must be used instead when find the centroids (He et al. 2004).

Since several experimental evidences have suggested that the multiple random initialization strategy outperforms other initialization methods in real-world conditions (see Pena et al. 1999, Hand and Krzanowski 2005, Steinley and Brusco 2007), we propose the following initialization methods: at each *K* from the ranges defined above, the Batch K-Means is run *R* times, each time from a random set of entities taken as initial centroids. Of the *R* resulting clusterings, that one minimizing the value of criterion (1) (see page 2) is chosen, and the value of criterion (1) (see page 2) at it is denoted by W_K. We accept *R*=100. This choice is consistent with, first, Hand and Krzanowski (2005) recommendation *R*=20 for *N* of the order of 200 in their experiments, and, second, our desire to simulate the constraints of real-world computations.

It should be noted that there have been suggested many improvements over the Straight K-Means version, leading to deeper minima of the criterion (1) (see page 2) for the same initializations, such as the adaptable change of centroids after each entity's Minimum distance assignment (McQueen 1967). Likas et al. (2003) presented a global K-Means algorithm which aims to gradually increase the number of clusters until *K* are found and this algorithm can be taken as an initialization of other clustering techniques (Steinley and Brusco 2007). This algorithm starts at one cluster and its centroid is the grand mean, and they then run the K-Means clustering algorithm with a gradual increase in the number of clusters N times, where N is the number of entities. They compared their method with multiple runs of the K-Means algorithm and claim that the global K-Means algorithm shows the best quality. Hansen and Mladenovich (2001) proposed a J-Means algorithm, where the centroids of clusters are relocated to entities which have not yet been selected as centroids rather than entities in clusters which may move to other clusters in K-Means and all entities of that cluster are reassigned to their closest centroids. This method along with 3 other methods have been applied on several famous real-world datasets, and J-Means shows very good performance on the cluster quality but the worst on the computational time. Other authors compared different versions of K-Means, for example, bisecting K-Means (Steinbach et al. 2000) clustering method, described in Section 2.4. Hamerly and Elkan (2002) proposed a G-means (G stands for Gaussian) clustering method to ensure the entities in each cluster are Gaussian distributed. It runs Straight K-Means starting from one cluster or a small number of clusters, then if the entities in a cluster are Gaussian under hypothesis test, the cluster centroid remains; otherwise, the centroid of the cluster splits into two by adding two numbers obtained from a principal component based method to the centroid. The advantage of G-means is that only one parameter needs to be specified, that is, the significance level of the hypothesis test, which should be set in a standard way.

Another K-Means related improvement can be done by modifying the summary within cluster squared Euclidean distance, which can be generalized as a cost function (Kothari and Pitts 1999). The added term of the modified cost function further ensures the summary within cluster distance is minimal; therefore if the algorithm starts from a large number of clusters, the centriods are much closer to each other because of this added term. Kothari and Pitts (1999) applied this modification on four data sets and used the Dunn index (Dunn 1974) to validate the clustering results.

Another improvement can be done by applying the genetic algorithm, for example, GA-clustering proposed by Maulik and Bandyopadhyay (2000) (also see Bandyopadhyay and Maulik 2002) a genetic algorithm based clustering technique, from the idea of the evolutionary genetics, which improves K-Means by a process of selection, crossover and mutation until a termination criterion is reached. It creates a population of solutions based on the so-called fitness function and finds the good solution for the next generation according to the process iteratively. The fitness function is the multiplicative inverse of the summary within the cluster absolute Euclidean distance; therefore, to minimize the summary distance is to maximize the fitness function. A different version of the fitness function with the summary within cluster squared Euclidean distance is also tested and similar good performance of GA-clustering has been shown in the paper cited. A comparison of four methods including GA-clustering has been proposed by Paterlini and Krink (2006) and they suggest that the differential evolution method is superior to the other methods, which uses a more complex crossover procedure, because the mutation procedure of other methods is rather random search in the existing solutions.

Some authors propose a centroid-based clustering algorithm, for example, Leisch (2006) generalized the K-centroids method, which finds centroids which average distances between entities to the closest centroids is minimal, and nominal data clustering algorithm K-modes, which have been implemented in R statistical software package. The distance measure of K-modes is to count the number of dimensions of which an entity and its centroid do not have the same value. Modified criteria have been utilized by many (see, for reviews, Steinley 2006 and Bock 2007). These all are left outside of our experiments: only Straight K-Means is being tested since the thesis is confined to be K-Means related methods only.

3.3 Evaluation: distance between centroids

Since the generated data is a collection of entities from K^* Gaussian clusters, the results of a K-Means run can be evaluated by the quality of recovery of the following components of the generated clusters: (1) the number K^*, (2) the cluster centroids, and (3) the clusters themselves. This leads us to using three types of criteria based on comparison of each of these characteristics as produced by the algorithm with those in the generated data. The cluster recovery conventionally is considered of greater importance than the other two.

The recovery of K^* can be evaluated by the difference between K^* and the number of clusters K in the clustering produced with a procedure under consideration. Measuring the distance between found and generated centroids is not quite straightforward even when $K=K^*$. Some would argue that this should be done based on a one-to-one correspondence between centroids in the two sets, hence the best pair-wise distance matching between two sets. Others may consider that such a matching would not necessarily be suitable because of the asymmetry of the situation – one should care only about how well the generated centroids are reproduced by those found ones, so that if two of the found centroids are close to the same generated centroids, both should be considered its empirical representations. We adhere to the latter view, the more so that this becomes even more relevant, both conceptually and computationally, when K differs from K^*.

Another issue that should be taken into account is of the difference in cluster sizes: should the centroid of a smaller cluster bear the same weight as the centroid of a larger cluster? Or, on the contrary, should the relative cluster sizes be involved so that the smaller clusters have less of an effect on the total? To address this issue, we use both weighting schemes in the experiments conducted, to find out which of them is more consistent with cluster recovery than the other.

According to the "asymmetric" perspective above, to score the similarity between the generated centroids, $\boldsymbol{g_1, g_2, \ldots, g_{K^*}}$, and those obtained using one of the chosen algorithms in Table 3.2, $\boldsymbol{e_1, e_2, \ldots, e_K,}$ we utilize a procedure consisting of the following three steps:

(a) pair-wise matching of the obtained centroids to those generated:

For each $k=1,\ldots.K^*$, assign $\boldsymbol{g_k}$ with that $\boldsymbol{e_j}$ ($j=1,\ldots,K$) which is the nearest to it. Any not yet assigned centroid $\boldsymbol{e_i}$ then is matched to its nearest $\boldsymbol{g_k}$.

(b) calculating distances between matching centroids:

Let E_k denote the set of those $\boldsymbol{e_j}$ that have been assigned to $\boldsymbol{g_k}$; and $\alpha_{jk} = q_j/|E_k|$, where q_j is the proportion of entities in j-th found cluster (weighted version) or $\alpha_{jk} = 1$ (unweighted version). Define, for each $k=1,\ldots,K$, $dis(k) = \Sigma_{ej \in Ek}\, d(g_k,e_j)^* \alpha_{jk}$. The weighted distance is the average weighted distance between the generated and the set of matching centroids in the computed clusters; the unweighted distance is just the summary distance between all matching pairs of clusters. (The distance d here is Euclidean squared distance.)

(c) averaging the distances:

Calculate $D=\sum_{k=1}^{K^*} p_k * dis(k)$ where $p_k=N_k= |N_k|$, is the number of entities in the generated k-th cluster (in the weighted version), or $p_k= 1/K^*$ (in the unweighted version).

3.4 Evaluation: confusion between partitions

To measure similarity between two partitions, the contingency (confusion) table of the corresponding partitions of *I* is used. Entries in the contingency table are the co-occurrence frequencies of the generated partition clusters (row categories) and the obtained clusters (column categories): they are the counts of entities that fall simultaneously in both. Four coefficients, that is, adjusted Rand index ARI (Hubert and Arabie 1985, Yeung and Ruzzo 2001), average overlap *A*, the relative distance *M*, and Tchouproff's coefficient *T* (Mirkin 2005), are used for measuring the similarities between two partitions and the four coefficients capture different structural properties of partitions and expose different behaviour in our experiments, but regarding our main conclusions they tend to show the same outcome. This is why in the experimental result tables in Chapter 4 and 5 we present only values of ARI coefficient.

Denote the generated clusters (rows) by *k*, the obtained partition clusters (columns) by *j* and the co-occurrence counts by N_{kj}. The frequencies of row and column categories (cluster sizes) are denoted by N_{k+} and N_{+j}. The relative frequencies are defined accordingly as $p_{kj}=N_{kj}/N$, $p_{k+}=N_{k+}/N$, and $p_{+j}=N_{+j}/N$, where *N* is the total number of entities. We use a conventional similarity measure, the adjusted Rand index ARI defined by the following formula (Hubert and Arabie 1985, Yeung and Ruzzo 2001):

$$ARI = \frac{\sum_{k=1}^{K}\sum_{l=1}^{L}\binom{N_{kl}}{2} - \left[\sum_{k=1}^{K}\binom{N_{k+}}{2}\sum_{l=1}^{L}\binom{N_{+l}}{2}\right]/\binom{N}{2}}{\frac{1}{2}\left[\sum_{k=1}^{K}\binom{N_{k+}}{2} + \sum_{l=1}^{L}\binom{N_{+l}}{2}\right] - \left[\sum_{k=1}^{K}\binom{N_{k+}}{2}\sum_{l=1}^{L}\binom{N_{+l}}{2}\right]/\binom{N}{2}} \qquad \textbf{(8)}$$

where $\binom{N}{2} = \frac{N(N-1)}{2}$. ARI captures the similarities in the contents of pairs of entities belonging to the same clusters. The greater the ARI, the more similar are the partitions.

The relative distance to the real partition *M* and the relative chi-square contingency coefficient *T* are:

$$M = \sum_{k \in T} p_{k+}^2 + \sum_{j \in U} p_{+j}^2 - 2\sum_{k \in T}\sum_{j \in U} p_{kj}^2 \qquad (9)$$

$$T = \frac{\sum_{k \in T}\sum_{j \in U} \frac{p_{kj}^2}{p_{k+}p_{+j}} - 1}{\sqrt{(K-1)(E-1)}} \qquad (10)$$

where *K* is the real number of clusters and *E* is the estimated number of clusters. Average overlap *A* is another criterion related to the contingency table. Two tables are formed as follows: the row of the contingency table is divided by the number of items in the obtained cluster list and the column of the contingency table is divided by the number of items in the real cluster list. A table is obtained by summing up the previous two tables and is divided by 2. The entry in the table is then multiplied by the corresponding probability $p_{kj,}$ which will form a new table. Then the average overlap is calculated across the new table. The average overlap index captures the similarities in the contents of entities, not the pair of entities. The relative distance is quite similar to ARI but not normalized. Tchouproff's coefficient captures the statistical dependence, which goes against the statistical independence.

3.5 Summary

The bulk of the experimental study addresses one of the most controversial issues in clustering: the right number of clusters, which some may view as baseless because in many cases, "clusters are not in data but in the viewing eye." In the experiments, we try to maintain the case when clusters are in the data. The data are generated as sets of entities randomly drawn from Gaussian clusters, with the cluster sizes (proportions) drawn randomly as well. Using Gaussian clusters allows us to address the issue of modelling the cluster intermix in an intuitively appealing way in terms of within- and between-cluster spreads. This also enables us to conduct experiments by confronting two types of situations: well separated clusters (large between-cluster spread) and not well separated clusters (small between-cluster spread). We combine these with three different models of within-cluster spread and shape. One of the models is of conventional spherical cluster with a constant variance; the other two involve elongated clusters and different cluster variances. The twelve combined data settings provide rather different cluster structures for comparing different methods. To be closer to the real-world data analyses, we maintain relatively large data sizes (one or three thousand entities) and cluster numbers (7, 9 and 21). Another feature of our experimental setting is that to evaluate the results, we utilize the centroid recovery performance of a clustering method in addition to the conventional cluster recovery performance.

Chapter 4

Analysis of the Results

4.1 First series of evaluation tables and their analysis

The experiment is conducted in two instalments. The first instalment is, basically, to see whether our assumptions are right, specifically:

(i) If one of the two distance formulas, weighted and unweighted, is any better then the other;

(ii) If the randomness in the generated cluster sizes or initialization of centroids makes a difference, and if it does, what to do about it;

(iii) Are there any patterns in the recovery of the number of generated clusters K*, that go across the lines of the within- and between-cluster spread models accepted for the experiment? If there are, can they be used for enhancing the clustering procedures?

(iv) Are there any patterns in the cluster recovery within or across the within- and between-cluster spread models?

The major parameters of the first instalment of the experiment are six spread-shape models that are the result of combining two types of models: (a) either of the three cluster models according to the distribution of the within-cluster spreads and associated shape formats (the spherical shape for the constant spreads, and the elongated NetLab (see Generation of Gaussian mixture distributed data

2006) covariance for the variant within-cluster spreads), and (b) either of two modes of the between-cluster spreads, the “large” and “small”, according to Table 3.1.

The results of our experiments are presented in Table 4.1, 4.2 and 4.3, for the cases of 7, 9 and 21 Gaussian clusters generated respectively. The entries are averages of the respective evaluation values taken over 20 data sets generated, along with their standard deviations. In the experimental result tables, the standard deviations are divided by the averages, expressed in per cent. The reason for this is the presentational purpose. The cluster shape, spread and spatial sizes are taken according to Table 3.1 in Section 3.2. In Table 4.1, 4.2 and 4.3, we highlight two winners among the nine algorithms under comparison, at each of the six spread patterns (three cluster spread-shape models times two between-cluster spreads), by using the bold font. The two different between-cluster spreads are presented in different columns while the three cluster spread-shape models are reflected within the cells by three rows, as explained in the captions.

Comparison of iK-Means with 7 other methods at cluster=7 and 3 cluster structural models

	Estimated number of clusters		Weighted distance between centroids		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS	LaS	SmS
	8.28/5	4.00/0	48116.80/16	360.91/14*	326.49/14*	380.77/20	0.77/12	0.40/11
CH	10.70/6	4.00/0	1558562.68/15	3621.98/16	9699.31/14*	3279.99/14	0.64/11	0.31/12
	8.30/5	4.30/12	1595574.32/13	**55930.42/12**	10448.37/13*	56453.01/13	0.74/13	0.43/13
	7.39/6	**6.20/10**	128684.97/17	390.98/13	329.66/15	388.65/18	0.75/12	0.39/12
HT	**7.55/5**	8.89/10	1799188.85/16	3030.92/15	**9656.33/15**	3047.52/18	0.76/13	**0.38/12**
	7.55/6	**8.70/10**	1746987.36/14	60371.09/15	**10440.73/12**	58707.33/15	0.72/11	0.50/12
	5.25/7	5.85/11	49584.52/11	475.85/11	338.38/11	425.89/11	0.80/11	0.37/12
GS	5.75/8	5.12 / 9	1492546.32/14	3785.25/11	9642.58/11	3280.65/11	0.81/12	0.31/12
	5.95/7	5.25/11	1458569.52/11	59351.25/12	10589.52/12	54963.74/12	0.79/12	0.44/11
	10.67/6	4.00/0	51148.43/15	**360.90/12**	**325.04/13**	**353.96/19**	0.60/15	0.40/11
JS	10.00/0	4.78/10	1456705.09/14	3441.78/15	9743.94/13*	3018.44/15	0.74/12	0.37/11
	10.40/6	4.80/10	1766608.06/13	72390.75/12	10491.41/14*	58712.23/16	0.6911	0.50/15
	4.89/5	4.65/10	**44560.63/15**	**359.24/12**	**325.59/15**	**379.24/16**	**0.93/12**	**0.41/12**
SW	6.60/6	**5.44/10**	1412019.54/13	3375.02/15	9672.26/14*	**2997.17/16**	**0.90/11**	**0.40/11**
	5.60/5	**5.40/11**	1696914.01/15	62581.11/14	**10408.32/13**	55420.80/14*	**0.94/12**	**0.57/13**
	5.22/6	5.05 / 4	45201.58/17	476.60/15	341.30/14*	379.84/16	0.79/11	0.36/10
CD	5.00/0	5.00 / 0	**1365256.89/12**	3178.91/15	9741.09/14	3283.51/15	0.74/12	0.32/11
	5.00/0	5.00 / 0	**1390176.82/15**	56446.03/13	10476.44/13	56759.32/17	0.77/11	0.45/15
	5.00/0	**5.95/12**	45638.01/16	483.02/15	342.90/15	445.71/17	0.82/14	0.38/12
DD	**6.70/4**	5.11/10	1423139.34/15	3849.27/14	9740.43/14	3307.85/16	0.75/12	0.30/11
	6.20/6	5.30 / 9	1488715.14/14	**56111.21/16**	10486.01/15	56261.32/19	0.71/13	0.45/12
	5.44/5	17.90/18	**44586.72/15**	1142.03/15	328.19/13*	476.86/23	**0.97/13**	**0.41/12**
L_2	5.90/6	10.89/19	**1358256.30/15**	**2869.79/14**	**9658.11/13**	3096.48/20	**0.98/12**	0.33/15
	5.40/6	9.40/18	**1348704.94/14**	60274.25/17	10504.31/13*	**55334.98/20**	**0.95/11**	**0.53/15**
	16.78/7	35.00/21	58992.53/14	439.60/12	340.97/15	647.83/29	0.66/12	0.28/13
L_1	7.70/6	**7.67/18**	1513975.39/14	**2883.21/15**	9739.12/15	**3007.08/21**	0.73/11	0.28/14
	9.10/4	18.10/19	1499187.03/13	64655.17/15	10507.21/14	**55290.32/22**	0.74/15	0.37/15

* within 1% of the best value

Table 4.1 The average values of evaluation criteria at 7-clusters data sets with NetLab Gaussian covariance matrix for the large and small spread values (LaS and SmS, respectively) in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three cluster structure models: the spherical shape with constant cluster sizes on top, the PPCA elliptical shape with k-proportional cluster sizes in the middle, and the PPCA elliptical shape with k^2-proportional cluster sizes in the bottom. Two winners of the eight methods are highlighted using the bold font, for each of the options.

Comparison of iK-Means with 7 other methods at cluster=9 and 3 cluster structural models

	Estimated number of clusters		Weighted distance between centroids		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS	LaS	SmS
	11.55/8	4.00 / 0	53057.85/13	832.87/15	403.85/12	419.27/12	0.82 / 9	0.25/12
CH	12.10/4	5.30 / 5	1462774.95/11	465599.77/14	11788.38/14*	2932.79/19	0.81 / 8	0.21/12
	11.15/8	4.11 / 8	1560337.21/11	50703.90/12	12146.83/13*	53779.46/15	0.79 / 9	0.22/12
	8.27/6	**7.60/10**	47293.32/13	**742.47/13**	412.40/13	386.01/14	0.89 / 9	0.29/10
HT	8.55/7	**9.40 / 9**	1332058.56/15	409831.54/14	11833.21/14*	2965.56/15	0.90 / 9	0.37/11
	9.35/7	**9.12/10**	1495325.18/14	51941.10/15	12154.99/15	55286.55/14	0.84 / 9	0.28/12
	6.25/7	5.75 / 8	47295.85/11	795.52/11	438.33/12	385.25/12	0.77/11	0.28/13
GS	6.75/8	5.95/10	1305125.52/10	394596.52/11	11758.62/12	2991.15/12	0.77/12	0.28/12
	5.95/8	6.25 / 9	1395568.25/11	51845.25/11	12185.62/13	54258.63/13	0.76/12	0.29/12
	12.12/8	4.50 / 0	55417.22/15	798.96/13	403.38/13	419.27/13	0.77/10	0.25/12
JS	12.75/9	6.15 / 8	1548757.47/12	510687.27/15	11785.21/13*	2908.33/15	0.82 / 8	0.24/13
	12.10/8	4.45 / 5	1570361.91/12	50716.82/12	12131.86/12*	53699.24/14	0.80 / 8	0.22/11
	6.29/8	4.54/10	**46046.56/15**	805.30/15	418.26/12	418.66/14	0.92/10	0.26/13
SW	6.95/7	4.95 / 4	**1299190.70/15**	**393227.66/14**	11876.31/13*	2846.31/16	0.92 / 8	0.27/12
	7.15/8	4.28/11	**1462999.91/12**	**50383.53/13**	12203.58/12	**53583.12/16**	**0.85 / 6**	0.22/13
	5.31/7	5.11 / 9	**47122.13/14**	791.76/12	429.96/12	**373.93/12**	0.78/12	0.27/13
CD	5.30/6	5.10/10	**1305051.80/14**	**394572.84/13**	11943.98/13	2897.61/18	0.78/12	0.28/14
	5.20/6	5.31 / 9	**1350841.29/13**	51968.86/12	12265.98/12	55040.86/15	0.75/12	0.25/13
	5.67/3	**6.42 / 8**	47190.83/15	792.15/15	435.37/12	409.97/13	0.75/12	0.27/12
DD	4.90/3	5.60 / 9	1306014.88/13	395524.66/12	11979.30/13	2996.28/18	0.74/12	0.24/12
	5.30/3	**5.83 / 8**	1394892.59/14	50813.28/15	12286.43/12	53912.13/13	0.71/12	0.27/10
	8.67/6	13.00/18	49095.21/15	1110.88/13	**402.47/12**	**335.91/23**	**0.99 / 9**	**0.48/12**
L_2	**8.80/6**	**10.80/16**	1485719.73/12	486979.24/14	**11771.70/12**	**2661.41/20**	**0.99/10**	**0.42/12**
	7.95/7	13.44/18	1444645.99/15	51226.10/12	**12031.13/11**	54026.92/15	**0.90 / 9**	**0.45/12**
	9.33/6	25.00/18	54478.33/13	**705.61/15**	**400.18/12**	381.12/25	**0.92 / 9**	**0.38/12**
L_1	**8.80/7**	16.10/17	1487335.77/13	487940.63/13	**11767.34/13**	**2648.60/20**	**0.99/10**	**0.41/12**
	10.00/6	23.11/18	2092537.57/12	**50506.80/12**	**12114.01/12**	**53507.21/16**	0.84/10	**0.41/12**

* within 1% of the best value

Table 4.2 The average values of evaluation criteria at 9-clusters data sets with NetLab Gaussian covariance matrix for the large and small spread values (LaS and SmS, respectively) in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three cluster structure models: the spherical shape with constant cluster sizes on top, the PPCA elliptical shape with k-proportional cluster sizes in the middle, and the PPCA elliptical shape with k^2-proportional cluster sizes in the bottom. Two winners of the eight methods are highlighted using the bold font, for each of the options.

Comparison of iK-Means with 7 other methods at cluster=21 and 3 cluster structural models

	Estimated number of clusters		Weighted distance between centroids		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS	LaS	SmS
CH	33.25/11	14.52/10	68196.52/11	1052.63/11	578.42/12*	498.42/12	0.81/10	0.35/12
	34.95/10	14.85/10	178529.52/12	24584.52/12	12685.52/11	6574.54/12*	0.82/11	0.34/12
	31.45/10	13.95/12	181648.52/11	61458.63/12	14896.54/11	35145.25/12	0.79/12	0.29/12
HT	**20.72/11**	**19.85/10**	**66524.85/12**	958.46/12	569.12/11*	487.65/11*	0.81/12	0.34/12
	20.45/11*	**20.42/10**	177389.63/12	22548.63/12	12578.12/10*	6585.19/11*	0.82/11	0.33/11
	21.85/10*	**21.79/10**	179526.12/11	**59851.58/12**	14746.49/11	34719.49/11*	0.82/12	0.35/10
GS	17.52/10	14.36/11	67521.95/12	984.65/11	571.45/12*	491.48/12*	0.79/12	0.34/11
	16.85/10	15.42/10	178528.62/12	23758.96/12	12694.28/12	6512.75/11*	0.81/12	0.29/12
	16.45/10	16.52/11	182176.52/13	61953.25/11	14940.63/11	34751.85/12*	0.81/12	0.30/10
JS	32.15/10	13.99/11	67195.52/12	975.27/12	574.45/12*	489.75/10*	0.82/12	0.32/12
	34.12/10	14.85/12	179526.52/13	23579.48/12	12501.27/11*	6541.51/11*	0.81/11	0.33/12
	32.62/11	15.75/12	182274.85/12	61847.52/11	14975.75/12	34275.15/12*	0.83/11	0.34/10
SW	15.42/10	14.18/12	**66745.85/12**	**931.42/12**	562.15/10*	485.42/12*	0.79/11	0.33/12
	16.65/10	15.95/10	**176859.52/12**	**21587.54/12**	12649.57/11	6524.75/11*	0.81/11	0.29/11
	14.85/10	16.85/11	**180493.85/11**	60157.24/11*	14734.15/10	34815.16/12*	0.80/12	0.31/10
CD	17.29/10	15.85/11	67085.12/12*	942.35/12*	571.16/10*	486.52/11*	0.81/12	0.29/10
	16.76/10	15.52/10	**176384.85/11**	**21465.18/12**	12534.75/11*	6518.27/12*	0.82/12	0.32/12
	18.45/12	17.04/12*	**180052.63/11**	**59941.11/12**	14576.67/11	34842.19/11*	0.80/11	0.34/12
DD	16.78/10	**17.85/12**	66975.52/12*	954.25/12	572.42/10*	482.45/11*	0.79/12	0.34/12
	18.65/10	**16.49/10**	179416.85/12	22951.54/11	12549.42/11*	6547.73/11*	0.81/12	0.33/12
	17.95/12	**17.42/10**	181756.85/12	60175.52/12*	14594.12/11	34768.42/10*	0.82/11	0.29/12
L_2	20.12/11*	26.85/20	69015.52/11	942.16/12*	**571.48/12**	**479.48/11**	**0.99/11**	**0.42/12**
	20.85/12	28.45/22	179526.75/12	22568.42/12	**12347.57/11**	**6498.15/10**	**0.99/12**	**0.44/11**
	21.32/10	30.42/10	181085.63/12	59975.54/10*	**14259.54/12**	**34152.57/12**	**0.99/10**	**0.43/10**
L_1	**20.35/10**	39.45/18	68759.52/12	**934.16/10**	**570.85/12**	**480.45/12**	**0.98/12**	**0.45/12**
	21.42/11	38.63/19	179528.53/12	21984.85/11*	**12468.27/11**	**6501.57/12**	**0.99/11**	**0.44/11**
	20.95/12	39.52/21	182163.52/13	60846.18/12*	**14375.25/12**	**34271.45/12**	**0.95/12**	**0.43/10**

* within 1% of the best value

Table 4.3 The average values of evaluation criteria at 21-clusters data sets with NetLab Gaussian covariance matrix for the large and small spread values (LaS and SmS, respectively) in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three cluster structure models: the spherical shape with constant cluster sizes on top, the PPCA elliptical shape with k-proportional cluster sizes in the middle, and the PPCA elliptical shape with k^2-proportional cluster sizes in the bottom. Two winners of the eight methods are highlighted using the bold font, for each of the options.

With respect to the issues (i)-(iv) raised for this batch of experiments, one can notice the following:

(i) The orderings of estimates according to the weighted and unweighted distances between centroids differ considerably. The winners with respect to the centroid recovery closely match the winners with respect to the cluster recovery when the unweighted distance is used, and do not match at all, when the weighted distance is used. This goes in line with the view that K-Means clustering results can be interpreted as a form of typology at which centroids represent its so-called intensional, that is, conceptual, part. According to this view, the distances should not be weighted by the cluster sizes. The results in the table support this view and make us use only the unweighted distance in the further experiments.

(ii) The averages reported in Table 4.1, 4.2 and 4.3 are rather stable: all the standard deviations lie within 15% of the average values (except for L_1 and L_2 at the small between-cluster spread associated with very high numbers of clusters found – these two will be modified later on). That means that the randomness of the choice of initial centroids and the randomness in cluster sizes do not affect the results that much, and can be considered justifiable.

(iii) With regard to the number K^* recovery, one can easily notice that the differences in within-cluster shape/spread do not appear to affect the outcomes. However, with respect to between-cluster spread differences, there can be discerned four different patterns: (a) HT consistently chooses K values that are very close to K^*=7, 9 and 21; (b) L_1 and L_2 closely follow K^*=7, 9 and 21 at the large spread and lead to much larger Ks at the small spread – this especially concerns L_1; (c) when K^*=7, 9 and 21, both CH and JS overestimate K^* at the large spread and underestimate it at the small spread, and (d) when K^*=7, 9 and 21, GS, SW, CD, and DD underestimate K^* at both between-cluster spreads, though when K^*=9, SW is close at the large spread and DD at the small spread, but when K^*=7, SW is close at the small spread and DD at the large spread.

(iv) With respect to the cluster recovery, the common pattern is that the larger spreads lead to better reproduction of clusters than the small spreads for all of the algorithms. Overall, the algorithm L_2 outperforms other methods but when $K^*=7$, the algorithms SW and HT join; when $K^*=9$, the algorithms SW and L_1 join and when $K^*=21$, the algorithm L_1 joins.

4.2 Adjusted intelligent K-Means

According to the experiment, iK-Means methods L_2 and L_1 may lead to excessive numbers of clusters, while HT, on the other hand, makes a very good recovery of the number of clusters. This leads us to suggest that the HT number-of-cluster results should be taken as a reference to adjust the threshold for removing small AP clusters for the initial setting in iK-Means. So far, only AP singletons are removed from the initial setting. If other "smaller" AP clusters are removed, the chosen K will be smaller and, thus, closer to K^*. A straightforward option would just remove all AP clusters whose sizes are less than or equal to a pre-specified discarding threshold DT. Given K_h, found with the Hartigan rule, a suitable discarding threshold DT can be found in such a way that the number of clusters K_{DT} identified with DT, taken as the discarding threshold, is close enough to K_h. This can be done by gradually increasing DT from the default value $DT=1$. A typical sequence of steps, at a given K_h, say $K_h=9$, could be like this: at $DT=1$, the number of AP clusters is $K_{DT}=32$; at DT=2, still $K_{DT}=32$, that is, no doubletons among the AP clusters; then $K_3=29$, $K_4=24$, $K_8=20$, $K_{11}=14$, $K_{12}=11$, and $K_{14}=8$ (the omitted

DT values give no reduction in K_{DT} values). Therefore, *DT* should be taken as *DT=14*. Since K_h value is not necessarily correct but rather indicative, *DT=12*, leading to 11 clusters, is also acceptable, especially if K*=10 or 11. Thus, one can use a computational routine of increasing DT one by one until K_{DT} becomes less than θK_h. When we put θ=1.1, the next K_{DT} value is typically less than K_h, whereas θ=1.2 leaves K_{DT} rather large, but θ=1.15 produces reasonable approximations of K_h. We refer to thus HT conditioned versions of L_2 and L_1 as AL_2 and AL_1.

HT-adjusted iK-Means

0. HT-number: Find the number of clusters K_h by using R runs of Straight K-Means at each *K* with the Hartigan rule.
1. iK-Means number: Find the number of clusters by using iK-Means with the discarding threshold *DT=1*. Let it be K_{ls} for L_2 and K_{lm} for L_1.
2. Adjust: If K_{ls} (or K_{lm}) is 1.15 times greater than K_h, increase the discarding threshold by 1 and go to step 1 with the updated *DT*. Otherwise, halt. (The adjustment factor value of 1.15 has been found experimentally.)

4.3 Second series of the experiment and their analysis

The second series of our experiments differs from the first one in three aspects:

(1) The adjusted versions of iK-Means clustering, AL_2 and AL_1, are included in the list of methods;

(2) Data sets with the number of clusters K* in three versions, 7, 9 and 21 clusters, are generated as described in section 3.1;

(3) The cluster shapes and cluster distances are fully crossed.

Therefore, the set of data structures generated here is expanded to 24 models by fully crossing the following four values:

(a) Three versions of the number of clusters K*, 7, 9 and 21 clusters;

(b) Two versions of the cluster shape, either spherical or elliptical, as described in section 3.1.C;

(c) Three versions of the within-cluster spread – constant, linear and quadratic, as described in section 3.1.D;

(d) Two versions of the between-cluster spread, large and small, as described in section 3.1.D with the spread values presented in Table 3.1.

The issues to be addressed in these experiments are those (ii)-(iv) above, and, additionally, as follows:

(i) Is there any pattern of (dis)similarity between the two data size formats;

(ii) Are the HT-adjusted iK-Means methods better than the original ones;

(iii) Are the algorithms’ recovery properties at the constant spherical within-cluster-spread model any better than those at the elongated not-constant spread clusters?

The averaged, over ten to twenty data sets generated at each of the 24 patterns, evaluation criteria values are presented in Tables 4.4 to 4.9. Each of the four tables corresponds to one of the four combinations of the size (a) and shape (b), whereas the six combinations of spread (c) and (d) are presented within each of the Tables 4.4 to 4.9.

The cluster centroid recovery results in Tables 4.4 to 4.9 are presented with a change in reporting: the weighted distance case is removed so that only the unweighted distances are left. Moreover, the distances are rescaled to achieve comparability across the between-cluster spread models, so that issue (vii) can be addressed with just visual inspection by a naked eye. The distance between centroids recovery is calculated in a Euclidean space, not in a squared Euclidean space. When we move the centroids by multiplying a value, for example 30, the squared distance becomes the square of the value greater, in this case, 30^2=900.

The rescaling is conducted according to the inter-cluster spread values in Table 3.1 and takes into account that, at the small within-cluster spreads, the spread value at k^2-proportional model, 2, is four times greater than that at k-proportional model, 0.5, and 10 times greater than that at the equal spread model, 0.2. By multiplying the distances between centroids at the equal spread model by $100=10^2$ and at the k-proportional model by $16=4^2$, they are made comparable with those at the k^2-proportional model. (Note that the distance between centroids is squared Euclidean, which implies the quadratic adjustment of the values.) Similarly, at the large spreads, the within-cluster spread values at the variant spread models are the same while that at the constant spread model is 5 times smaller, so we multiply the distances between centroids at the equal spread model by $5^2=25$.

Here are the findings related to each of the issues above:

(v) Tables 4.4 to 4.9 show a remarkable degree of similarity regarding the main findings of the first series of experiments:

a. The relatively small standard deviations;

b. The same four groupings of the procedures with regard to the number of clusters K* recovery, with the obvious AL_2 and AL_1 effects;

c. The same winners over a bulk of the experimental conditions, though HT at K*=21 shows winning performances over some of the conditions too.

(vi) The HT-adjusted iK-Means methods are not better than the original ik-Means with respect to the cluster recovery; they, however, are better with respect to the number of clusters. It is somewhat surprising that the absolute error based method L_1 is on par with the square error based method L_2, in spite of the fact that the data is generated according to Gaussian distributions favouring squared distances.

(vii) The algorithms' recovery properties at the equal within-cluster-spread model are not much better than those at the elongated not-constant spread clusters, whichever measure is used – the centroid or cluster recovery. Yet most methods perform better when the cluster spatial sizes are less different: at the constant sizes the best, and at the

k^2-proportional sizes the worst. However, the effects of differences in within-cluster spread-shape patterns are rather minor.

Comparison of adjusted iK-Means with 9 other methods at cluster=7, cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
	8.28/5	4.00 / 0	8162.25/14*	38077.00/20	0.77/12	0.40/11
CH	9.56/7	5.00 / 0	9218.27/12	42578.25/12	0.58/12	0.35/11
	9.25/7	5.84 / 8	9957.24/13	48249.26/12	0.72/12	0.48/12
	7.39/6	**6.20/10**	8241.50/15	38865.00/18	0.75/12	0.39/12
HT	7.65/7	8.75 / 7	9217.57/12	42671.48/10	0.65/12	0.34/11
	7.12/7	9.15 / 9	9910.24/12	47527.75/10*	0.73/12	0.42/11
	5.25/7	5.85/11	8459.29/11	42589.52/11	0.80/11	0.37/12
GS	5.14/6	6.14/11	9312.59/11	43057.85/12	0.75/11	0.32/11
	5.18/7	5.79/10	9917.24/10	47562.75/10*	0.68/10	0.44/11
	10.67/6	4.00 / 0	**8126.00/13**	**35396.00/19**	0.60/15	0.40/11
JS	9.75 / 7	9.49 / 9	9327.45/11	42759.42/10	0.68/12	0.35/15
	10.71/7	10.24 / 8	10048.18 / 9	48217.35/12	0.69/12	0.42/12
	4.89/5	4.65/10	8139.74/15*	37983.00/16	0.93/12	0.41/12
SW	7.75/6	6.54/11	9299.48/10	42873.15/10	0.78/12	0.31/12
	6.49/7	5.75/10	10057.26/10	47657.85/12	0.85/13	0.47/12
	5.22/6	5.05 / 4	8532.50/14	37984.00/16	0.79/11	0.36/10
CD	4.50/0	5.27 / 8	9314.67/10	42496.18/12	0.75/10	0.37/12
	4.50/0	4.85/11	9957.15/10	48018.72/11	0.76/10	0.45/10
	5.00/0	5.95/12	8572.50/15	44571.00/17	0.82/14	0.38/12
DD	5.85/7	6.27 / 8	9327.18/10	42579.27/12	0.68/12	0.38/10
	6.37/7	5.85/11	9948.26/10	**47524.52/12**	0.71/13	0.43/15
	5.44/5	17.90/18	8240.75/13*	47686.00/23	**0.97/13**	0.41/12
L_2	4.96/7	12.75/25	9248.52/11	42279.52/18	**0.96/10**	0.37/14
	5.17/6	11.49/18	9968.85/12	48078.21/20	**0.94/10**	0.42/13
	16.78/7	35.00/21	8524.25/15	64783.00/29	0.66/12	0.28/13
L_1	6.95/7	14.48/18	9247.35/12	**41917.35/12**	0.68/12	0.34/12
	7.28/7	17.48/21	9952.49/11	**47495.57/21**	0.72/13	0.45/10
	6.44/5	**6.10 / 7**	**8129.75/13**	37780.00 / 9	**0.97/13**	**0.60/10**
AL_2	**7.15/7**	**7.24 / 8**	**9095.45/12**	**42175.25/12**	**0.94/11**	**0.57/10**
	7.25/8	**7.75 / 7**	**9745.18/12**	48256.52/10	**0.95/10**	**0.52/13**
	16.78/7	**6.10 / 9**	8224.25/15	**29727.00/11**	0.76/14	**0.60/12**
AL_1	**7.49/6**	**7.42 / 6**	**9125.75/13**	42834.27/12	0.71/13	**0.56/12**
	8.47/8	**7.37 / 7**	**9713.25/13**	48527.17/13	0.72/13	**0.49/11**

* within 1% of the best value

Table 4.4 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL_2 (adjusted L_2) and AL_1 (adjusted L_1) . Two winners of 10 in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

Comparison of adjusted iK-Means with 9 other methods at cluster=9, cluster shape=elliptical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
CH	5.25/7	3.82 / 8	9375.18/11	47957.19/12	0.58/11	0.35/12
	10.70/6	4.00 / 0	9699.31/14	52479.84/14	0.64/11	0.31/12
	8.30/5	4.30/12	10448.37/13*	56453.01/13	0.74/13	0.43/13
HT	**8.07/8**	7.87 / 9	9317.26/12	47967.52/13	0.69/11	0.38/11
	7.55/5	8.89/10	9656.33/15*	48760.32/18	0.76/13	0.38/12
	7.55/6	8.70/10	10440.73/12*	58707.33/15	0.72/11	0.50/12
GS	5.17/8	4.01 / 8	9297.47/12*	47634.48/12*	0.74/11	0.39/11
	5.75/8	5.12 / 9	9642.58/11	52489.52/11	0.81/12	0.31/12
	5.95/7	5.25/11	10589.52/12	54963.74/12	0.79/12	0.44/11
JS	9.74/9	4.93 / 9	**9275.81/12**	48921.75/13	0.64/13	0.40/11
	10.00/0	8.78/10	9743.94/13	48295.04/15	0.74/12	0.37/11
	10.40/6	10.80/10	10491.41/14	58712.23/16	0.69/11	0.50/15
SW	4.73/8	5.49 / 8	9301.75/12	48276.96/12	0.89/13	0.40/11
	6.60/6	4.78/10	9672.26/14*	47954.72/16	0.90/11	0.40/11
	5.60/5	5.40/11	10408.32/13*	55420.80/14	0.94/12	**0.57/13**
CD	5.11/7	5.17 / 8	9395.17/11	48672.45/12	0.80/10	0.38/11
	5.00/0	5.00 / 0	9741.09/14	52536.16/15	0.74/12	0.32/11
	5.00/0	5.00 / 0	10476.44/13	56759.32/17	0.77/11	0.45/15
DD	5.88/7	5.23 / 9	9401.57/12	49019.46/12	0.81/13	0.39/13
	6.70/4	5.11/10	9740.43/14	52925.60/16	0.75/12	0.30/11
	6.20/6	5.30 / 9	10486.01/15	56261.32/19	0.71/13	0.45/12
L_2	4.84/7	15.48/21	9297.15/12*	**46218.12/18**	0.91/12*	0.40/11
	5.90/6	10.89/19	9658.11/13*	49543.68/20	**0.98/12**	0.33/15
	5.40/6	9.40/18	10504.31/13	**55334.98/20**	**0.95/11**	0.53/15
L_1	9.23/7	17.64/18	9285.49/12*	48567.52/19	**0.92/11**	0.34/12
	7.70/6	7.67/18	9739.12/15	48122.88/21	0.73/11	0.28/14
	9.10/4	18.10/19	10507.21/14	**55290.32/22**	0.74/15	0.37/15
AL_2	**7.24/8**	**7.35 / 9**	**9276.49/11**	**47349.75/12**	**0.93/13**	**0.63/13**
	6.90/6	**7.24 / 8**	**9595.11/13**	**46592.16 / 9**	**0.97/13**	**0.53/12**
	6.40/6	**7.75 / 7**	**10369.31/13**	56806.21/10	**0.95/11**	**0.55/11**
AL_1	8.49/7	**7.29 / 9**	9308.46/11	48086.75/13	0.74/13	**0.65/11**
	8.70/6	**7.42 / 6**	**9635.12/15**	**47203.20/11**	0.88/11	**0.53/12**
	9.10/4	**7.37 / 7**	**10386.21/14**	57908.32/13	0.74/15	0.41/11

* within 1% of the best value

Table 4.5 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL_2 (adjusted L_2) and AL_1 (adjusted L_1). Two winners of 10 in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

Comparison of adjusted iK-Means with 9 other methods at cluster=9, cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
	11.55/8	4.00 / 0	10096.25/12*	41927.00/12	0.82 / 9	0.25/12
CH	10.76/9	5.17 / 8	13859.21/12	48657.52/13	0.78/10	0.28/11
	9.98/9	5.49 / 7	19247.63/12	49657.52/11*	0.82 / 9	0.25/11
	8.27/6	**7.60/10**	10310.00/13	38601.00/14	0.89 / 9	0.29/10
HT	8.06/8	9.77/11	13795.45/12	49349.42/13	0.89/10	0.23/11
	9.07/9	9.85/12	19067.85/12	50348.52/12	0.84 / 9	0.27/12
	6.25/7	5.75 / 8	10958.25/12	38526.96/12	0.77/11	0.28/13
GS	6.47/8	4.35/12	13957.32/13	48963.75/12	0.79/12	0.27/11
	7.34/8	5.67/11	19123.52/12	49446.52/13*	0.79/12	0.30/13
	12.12/8	4.50 / 0	**10084.50/13**	41927.00/13	0.77/10	0.25/12
JS	11.95/7	5.19 / 8	13967.52/12	49052.75/14	0.79/10	0.27/14
	12.07/6	5.75 / 8	19635.75/12	50217.53/12	0.80 / 8	0.25/12
	6.29/8	4.54/10	10456.50/12	41866.00/14	0.92/10	0.26/13
SW	5.85/7	6.96/10	**13769.75/12**	49135.86/14	0.89/10	0.28/13
	6.07/7	5.08/11	19452.49/11	49834.47/13	0.85 / 8	0.25/11
	5.31/7	5.11 / 9	10749.00/12	37393.00/12	0.78/12	0.27/13
CD	5.18/8	6.49/12	13994.63/12	49235.36/12	0.77/11	0.30/11
	4.75/7	4.98 / 8	19379.85/13	49576.74/13*	0.79/11	0.27/11
	5.67/3	6.42 / 8	10884.25/12	40997.00/13	0.75/12	0.27/12
DD	4.76/7	5.79 / 8	14027.67/12	49726.45/13	0.78/11	0.26/11
	6.85/8	6.98 / 8	19459.63/11	50176.35/12	0.71/12	0.28 / 9
	8.67/6	13.00/18	10061.75/12	**33591.00/23**	**0.99 / 9**	**0.48/12**
L$_2$	**8.76/8**	15.79/19	13867.63/11	**45367.16/18**	0.98/10	**0.45/11**
	8.92/7	25.46/21	19196.85/12*	**49174.37/17**	0.91/10	**0.42/11**
	9.33/6	25.00/18	**10004.50/12**	38112.00/25	0.92 / 9	0.38/12
L$_1$	**8.74/7**	17.69/19	13982.52/12	**46397.53/21**	**0.99/10**	**0.43/11**
	9.86/9	21.64/21	19237.45/13	**49324.52/21**	0.89/11	**0.47/12**
	8.50/5	**7.60 / 6**	10086.75/12*	33849.00/12*	**0.99/11**	**0.50/11**
AL$_2$	8.36/8	**9.25/10**	13846.38/11	47219.56/13	**0.99/10**	**0.43/12**
	9.24/8	**9.77/11**	**18963.52/11**	49734.54/12*	**0.94/11**	0.40/11
	8.70/6	7.50 / 6	10504.50/12	**30556.00/12**	**0.99/12**	0.44/10
AL$_1$	9.98/8	**8.95/12**	**13725.19/12**	47652.36/12	**0.99/11**	0.41/12
	9.37/8	**9.38/11**	**19035.16/13**	49652.46/13*	**0.93/12**	0.38/10

* within 1% of the best value

Table 4.6 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL$_2$ (adjusted L$_2$) and AL$_1$ (adjusted L$_1$) . Two winners of 10 in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

Comparison of adjusted iK-Means with 9 other methods at cluster=9, cluster shape=elliptical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
	9.43/12	6.52/11	11969.34/12	45793.48/12	0.81/10	0.27/11
CH	12.10/4	5.30 / 5	11788.38/14*	46924.64/19	0.81 / 8	0.21/12
	11.15/8	4.11 / 8	12146.83/13	53779.46/15	0.79 / 9	0.22/12
	8.57/11*	**8.97/12**	11785.34/12*	43967.25/13	0.85/10	0.28/11
HT	8.55 / 7	**9.40 / 9**	11833.21/14*	47448.96/15	0.90 / 9	0.37/11
	9.35 / 7	**9.12/10**	12154.99/15	55286.55/14	0.84 / 9	0.28/12
	5.00 / 0	6.95/12	11795.36/12*	44369.27/12	0.79/10	0.28/13
GS	6.75 / 8	5.95/10	11758.62/12	47857.52/12	0.77/12	0.28/12
	5.95 / 8	6.25 / 9	12185.62/13	54258.63/13	0.76/12	0.29/12
	11.75/10	5.29/12	12084.37/11	45736.18/12	0.77/11	0.26/11
JS	12.75 / 9	6.15 / 8	11785.21/13*	46533.28/15	0.82 / 8	0.24/13
	12.10 / 8	4.45 / 5	12131.86/12	53699.24/14	0.80 / 8	0.22/11
	7.65 / 9	5.08/11	11936.47/12	45739.27/11	0.91/11	0.28/12
SW	6.95 / 7	4.95 / 4	11876.31/13*	45540.96/16	0.92 / 8	0.27/12
	7.15 / 8	4.28/11	12203.58/12	53583.12/16	0.85 / 6	0.22/13
	5.19 / 9	4.00 / 0	11997.52/12	45691.34/11	0.77/11	0.25/14
CD	5.30 / 6	5.10/10	11943.98/13	46361.76/18	0.78/12	0.28/14
	5.20 / 6	5.31 / 9	12265.98/12	55040.86/15	0.75/12	0.25/13
	4.00 / 0	6.39/12	11857.20/12*	44637.18/11	0.77/11	0.27/11
DD	4.90 / 3	5.60 / 9	11979.30/13	47940.48/18	0.74/12	0.24/12
	5.30 / 3	5.83 / 8	12286.43/12	53912.13/13	0.71/12	0.27/10
	8.95 / 9	11.69/19	**11753.19/12**	**43593.14/18**	**0.99 / 9**	**0.51/13**
L_2	**8.80 / 6**	10.80/16	**11771.70/12**	**42582.56/20**	**0.99/10**	**0.42/12**
	7.95 / 7	13.44/18	12031.13/11	54026.92/15	0.90 / 9	**0.45/12**
	8.47 / 9	17.96/18	11896.49/12	**43829.76/17**	0.91/10	0.40/11
L_1	**8.80 / 7**	16.10/17	**11767.34/13**	**42377.60/20**	**0.99/10**	0.41/12
	10.00 / 6	23.11/18	12114.01/12	53507.21/16	0.84/10	**0.41/12**
	8.69/10	**9.31/12**	**11763.52/12**	45324.76/13	**0.99/10**	**0.50/13**
AL_2	8.70 / 7*	**9.90 / 7**	11871.70/15*	43536.32/11	**0.99/11**	**0.42/12**
	8.70 / 9	**9.40 / 9**	**11031.13/12**	**52098.21/12**	**0.95/11**	0.38/12
	9.64 / 9	9.81/11	11967.54/13	44679.52/13	**0.99/13**	0.48/11
AL_1	8.70 / 7*	10.60 / 9	11867.34/15*	44298.88/11	**0.99/10**	0.38/11
	9.50 / 9	9.60 / 9	**10114.01/13**	**53057.21/11**	**0.92/13**	0.35 / 9

* within 1% of the best value

Table 4.7 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL_2 (adjusted L_2) and AL_1 (adjusted L_1). Two winners of 10 in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

Comparison of adjusted iK-Means with 9 other methods at cluster=21, cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
CH	24.56 / 9	15.00 / 0	14598.62/10*	24583.26/10	0.79 / 8	0.24/11
	23.49/11	16.83 / 8	16658.37/13	25869.74/13	0.82 / 9	0.28/12
	24.63 / 9	17.09/10	19254.52/12	28563.64/13	0.79 / 9	0.25/12
HT	20.45/8	18.50 / 8	14378.52/11*	26164.25/13	0.90 / 9	0.21/11
	20.95 / 8	**20.39/10**	16764.96/13	26946.37/13	0.88/11	0.39/12
	22.85 / 7	22.79/10	19246.34/13	28837.96/13	0.88/10	0.30/12
GS	18.32 / 9	15.32/10	15489.65/10	24354.25/11	0.81/11	0.25/11
	18.75/10	17.91/10	16431.05/13	25736.48/13	0.81/11	0.26/12
	18.19/11	18.04/10	19113452/13	28543.65/13	0.78/11	0.29/11
JS	25.58 / 7	15.00 / 0	14478.96/12*	24583.26/10	0.76/10	0.24/11
	23.27 / 8	17.63 / 9	16776.14/12	25960.81/13	0.83 / 8	0.27/13
	24.08/11	16.74 / 8	19248.52/13	28619.57/11	0.82/10	0.25/11
SW	19.35 / 8	17.50/10	15895.52/11	22267.25/12*	0.93/10	0.26/12
	17.87/10	17.38 / 8	16737.57/13	25842.51/13	0.90/10	0.28/11
	18.65 / 8	18.29 / 9	19376.19/11	28736.11/13	0.83 / 8	0.25/12
CD	17.52 / 9	17.00 / 0	15254.95/11	27154.26/12	0.79/12	0.30/11
	18.17 / 7	17.82 / 9	16493.24/13	25964.75/13	0.78/11	0.29/12
	17.51 / 9	18.07/10	19237.82/13	28893.12/14	0.77/10	0.31/13
DD	17.84 / 9	17.25 / 8	15269.52/11	26458.25/10	0.79/12	0.35/11
	16.38 / 8	17.64/10	16793.52/13	26019.57/13	0.74/10	0.29/12
	17.74/10	18.19/12	19436.42/11	28631.75/13	0.70/11	0.31/10
L_2	**20.85 / 7**	25.85 / 8	**14254.85/11**	26954.23/12	**0.99 / 9**	0.36/10
	21.43 / 8	29.42/12	**16237.10/13**	**25234.27/13**	0.98/10*	**0.46/12**
	20.74 / 9	31.48/15	**18934.26/13**	**28443.59/13**	0.94/10	**0.41/12**
L_1	21.56/8	37.45/18	15254.85/11	24586.23/12	0.96 / 9*	0.40/11
	21.96/10	34.63/17	16634.91/13	**25336.52/19**	**0.99/10**	0.43/11*
	22.16/10	39.67/19	**18896.17/13**	28651.63/16	0.90/11	**0.39/11**
AL_2	20.32/8	**19.85 / 6**	**14358.95/11**	**22145.85/10**	**0.99/11**	**0.50/11**
	21.76/9	**20.79 / 7**	**16349.27/13**	25729.15/13	**0.99/11**	**0.45/11**
	21.87 / 9	**22.10 / 9**	19234.71/12	28931.25/10	**0.98/11**	0.37/12*
AL_1	**21.25 / 9**	**22.52 / 6**	15254.95/11	**21856.32/12**	**0.99/12**	**0.45/11**
	21.07 / 9	22.61/10	16836.49/12	26167.29/13	**0.99/11**	0.39/11
	22.13 / 8	**21.63 / 8**	19273.85/13	**28392.24/11**	**0.95/12**	0.36/10*

* within 1% of the best value

Table 4.8 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL_2 (adjusted L_2) and AL_1 (adjusted L_1). Two winners of ten in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

Comparison of adjusted iK-Means with 9 other methods at cluster=21, cluster shape=elliptical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
CH	25.67/7 24.96/8 23.45/9	18.59 / 9 17.35 / 9 16.45 / 9	16948.49/12 18789.25/11* 20984.62/10	21654.51/13 37856.25/19 55145.89/12	0.81 / 9 0.80 / 8 0.82/10	0.27/10 0.23/12 0.21/12
HT	**20.97/8** **21.12/8** **21.52/7**	22.08 / 9 **21.45/10** **21.12/11**	15949.52/11 18457.52/10* **20761.95/10**	**20369.85/13** 38152.52/15 59254.56/11	0.91 / 7 0.89 / 8 0.87/10	0.20/11 **0.45/12** 0.35/12
GS	19.57/8 17.56/9 17.52/8	17.64/12 16.52/11 18.32/10	16495.49/13 21278.32/11 21859.32/11	21549.18/13 37524.21/11 55328.45/11	0.82/11 0.77/10 0.79/11	0.26/12 0.27/11 0.26/10
JS	19.24/8 24.65/8 25.25/7	17.67/13 18.75 / 9 15.85 / 7	16627.49/13 18546.32/11* 21254.74/10	21687.13/12 37526.25/15 56254.85/14	0.78/11 0.82 / 8 0.81 / 9	0.23/12 0.25/13 0.25/11
SW	18.00/0 18.35/8 18.52/8	18.26/10 16.85 / 7 17.38 / 7	16762.56/12 21587.85/10 22459.45/12	21026.84/13 37859.26/16 56859.25/16	0.91/11 0.91 / 8 0.85 / 8	0.26/12 0.26/12 0.23/13
CD	17.97/8 18.52/7 16.45/9	17.32/13 17.25 / 8 18.52 / 9	16596.19/13 21148.52/11 22984.52/11	21738.16/11 37152.56/18 55492.17/15	0.81/11 0.78/11 0.77/10	0.26/12 0.27/14 0.29/13
DD	18.46/7 15.95/8 17.52/9	16.85/12 16.52 / 9 17.25 / 9	16815.24/13 20365.14/11 21523.65/11	21267.19/13 38185.54/18 56874.82/13	0.81/11 0.76/10 0.76/12	0.33/12 0.25/12 0.27/10
L_2	21.96/7 20.75/7 18.96/7	25.49/13 27.65 / 9 30.45 / 9	**15536.28/13** **18254.65/11** 22351.85/11	**20035.15/13** **31459.25/18** 53462.52/15	**0.99 / 9** **0.99/10** 0.90/10	0.33/11 0.40/12* **0.42/12**
L_1	22.07/9 **20.65/9** 22.45/9*	27.10/15 36.25/16 38.12/17	15863.87/13 18754.25/11 22145.88/11	20469.25/15* **29025.52/17** 52854.21/16	0.98 / 9* **0.99/10** 0.86/10	0.43/11 0.42/12* **0.40/12**
AL_2	**21.72/7** 21.85/9* **21.42/8**	**21.95/13** **21.87 / 8** **21.85 / 9**	**15532.45/12** 19658.52/11 **20542.65/11**	21354.56/13 38452.95/10 **51954.65/12**	**0.99/11** **0.99/11** **0.97/11**	**0.54/11** **0.45/12** 0.39/12*
AL_1	22.49/9 20.12/8* 21.85/7*	**21.72/12** 23.45 / 9 21.45 / 8*	15767.63/13 **18236.12/11** 22956.25/11	20861.57/13 37529.52/10 **52018.85/11**	**0.99/12** **0.99/10** **0.95/13**	**0.52/11** 0.38/11 0.38 / 9*

* within 1% of the best value

Table 4.9 The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are divided by the averages, expressed after slash in per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom. The rows correspond to ten K-Means methods (eight listed in Table 3.2 plus AL_2 (adjusted L_2) and AL_1 (adjusted L_1). Two winners of ten in each category are highlighted using the bold font. Distances between centroids are rescaled as described above according to factors in Table 3.1.

4.4 Summary

In this chapter, we compare the iK-Means related methods with seven other methods described in Chapter 2 and 3. In the first section of this chapter, we compare the two versions of iK-Means with seven methods by checking how well the generated clusters can be reproduced by the found ones. The two versions of iK-Means perform well apart from the number of clusters recovery. HT index performs well on the number of clusters recovery. This leads to the adjusted version of iK-Means algorithm. In the second series of the experiments, we compare the adjusted version of iK-Means methods with nine other methods including the two versions of iK-Means methods. It shows that HT-adjusted iK-means methods perform the best among the eleven methods.

Chapter 5

Relationship between L_1 & L_2 Versions

5.1 The difference of the methods

Another issue that remains unanswered is whether there is a difference between the two versions of the iK-Means method. Therefore, we conduct a series of similar experiments as above but only at the two versions of the iK-Means method, where the centroids and cluster recovery are evaluated between these two versions, rather than with the generated partition. In this set of experiments, the unweighted distance between centroids is applied because it shows that the weighted distance between centroids has no correlation with cluster recovery and number of cluster recovery in the experimental results shown in Table 4.1 to 4.3. The cluster shape is the conventional spherical shape of Gaussian clusters because the spherical Gaussian clusters are one of the simplest data structures. The between-cluster and within-cluster spread values are taken from Table 3.1. The two settings for the data sizes are: (i) N=1000, M=15, K*=7 and 9 – about 110 entities in a cluster on average, and (ii) N=3000, M=20, K*=21 – about 145 entities in a cluster on average.

Tables 5.1 to 5.3 show the experimental results of the comparison. The averages reported in Tables 5.1 to 5.3 are rather stable: all the standard deviations lie within 15% of the average values, except when the between-cluster spread is small and this match with the findings of the previous experiments. The values of the ARI index are rather small compared to those in Tables 4.1 to 4.9 in both large and small between-cluster spreads. The ARI index for L_2 method in large between-cluster spreads in Table 4.1 is 0.97; whereas the ARI index in Table 5.1 for L_2 method in large between-cluster spreads is 0.62. On average, the ARI index for L_2 and L_1 methods in Table 4.1 to 4.9 is 0.99; whereas the ARI index in Table 5.1 to 5.3 is 0.65. This indicates that the two versions of iK-Means may produce very different results.

Comparison of L_2 and L_1 at clusters=7 and cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	6.12 / 9 7.85/10 6.75/11	15.42/25 17.45/25 14.85/21	1524.75/11 27451.85/14 38254.52/13	345.85/23 2719.52/24 48529.12/27	0.62/11 0.67/10 0.61 / 9	0.29/12 0.31/13 0.31/12
L_1	6.45/11 7.12/11 8.25/11	16.45/21 24.52/24 19.52/24	1425.74/12 28519.52/11 39219.14/12	349.52/25 2465.85/20 41296.12/25	0.59 / 9 0.61/11 0.64/12	0.31/12 0.29/12 0.30/13

* within 1% of the best value

Table 5.1 The data entities in each cluster are sampled from Gaussian distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom.

Comparison of L_2 and L_1 at clusters=9 and cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	8.27/10 8.75/11 8.45/12	12.85/20 11.75/25 14.75/27	1242.65/12 24785.12/13 32478.95/14	236.52/24 2513.25/20 37589.52/17	0.64 / 9 0.63/10 0.67 / 9	0.32/12 0.24/12 0.34/12
L_1	8.95/12 8.12/14 9.45/13	27.45/21 18.45/18 20.45/18	1469.02/12 27458.96/15 37859.12/13	374.52/23 2614.56/20 47851.36/18	0.65 / 9 0.66/10 0.67/10	0.31/12 0.29/12 0.30/12

* within 1% of the best value

Table 5.2 The data entities in each cluster are sampled from Gaussian distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom.

Comparison of L_2 and L_1 at clusters=21 and cluster shape=spherical

	Estimated number of clusters		Adjusted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	20.74/11 21.85/12 22.15/11	25.15/20 28.15/28 27.56/12	1745.52/12 29483.12/11 30158.52/10	358.12/25 3015.52/20 39581.26/21	0.65/10 0.63/10 0.60/11	0.32/10 0.31/10 0.32/11
L_1	20.95/11 21.15/12 21.65/11	40.51/30 38.05/12 33.15/12	1625.42/11 26859.12/10 36152.85/10	294.52/20 3125.65/24 39415.12/23	0.61/11 0.62/10 0.60/12	0.32/11 0.31/10 0.30/10

* within 1% of the best value

Table 5.3 The data entities in each cluster are sampled from Gaussian distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent. The three values in a cell refer to the three within-cluster spread models: the constant on top, the k-proportional cluster sizes in the middle, and the k^2-proportional cluster sizes in the bottom.

5.2 Suitable data structures

The experimental results in Chapter 4 show that in general L_2 always performs better than L_1. The cluster shapes in those experiments are Gaussian clusters in spherical and ellipsoidal and the data entities in each cluster are generated independently sampling from a Gaussian distribution. The experimental results in Chapter 4 are in line with the view that L_2 version of K-Means is a method for fitting with Gaussian mixture model. Given n independent Gaussian distributed random numbers $x_1, x_2, \ldots, x_n$, in order to find the maximum likelihood estimation of the parameters mean μ of the continuous Gaussian joint probability density function $f(x_1, x_2, ..., x_n) = \frac{1}{(\sigma\sqrt{2\pi})^n} \prod_{i=1}^{n} \exp(-\frac{1}{2}(\frac{x_i - \mu}{\sigma})^2)$, one needs to minimize the sum $\sum_{i=1}^{n} (x_i - \mu)^2$. This sum is exactly the square Euclidean distance, also known as the least square criterion in K-Means clustering.

The above calculation can simply apply to the continuous exponential distribution. Therefore, in order to find the maximum likelihood parameter estimation of its continuous joint probability density function given n independent exponentially distributed numbers x_1, $x_2,\ldots x_n$, $f(x_1, x_2, ..., x_n) = \frac{1}{(\beta)^n}\prod_{i=1}^{n}\exp(\frac{x_i - \mu}{\beta})$, the sum that needs to be minimized becomes $\sum_{i=1}^{n}|x_i - \mu|$. This is the Manhattan distance, also known as the least moduli criterion in K-Means clustering.

The experiment settings are similar with the experiments in Section 5.1 of the spherical cluster structure with 7, 9 and 21 generated clusters. The two settings for the data sizes are: (i) *N*=1000, *M*=15, *K**=7 and 9 – about 110 entities in a cluster on average, and (ii) *N*=3000, *M*=20, *K**=21 – about 145 entities in a cluster on average. The clustering results of L_1 and L_2 methods are compared with the generated clusters to see how well the generated clusters can be reproduced by the two versions of iK-Means. The constant within-cluster spread values are taken in this set of experiments. The simulation results are shown in Table 5.4, 5.5 and 5.6. The results clearly show that L_1 outperforms L_2, which proves the above implication.

Comparison of L_2, L_1, AL_2, and AL_1 with the generated clusters at clusters=7 and cluster shape=spherical

	Estimated number of clusters		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	7.55/10	9.20/10	860857.45/10	133587.52/10	0.75/10	0.53/11
L_1	**7.30 / 9**	**7.55/10**	**852579.52/10**	**128945.76/11**	**0.79 / 9**	**0.65/10**
AL_2	6.25/11	8.25/11	861075.85/11	131269.85/11	0.77/11	0.54/11
AL_1	**7.08/11**	**7.15/10**	**850798.85/10**	**129125.19/11**	**0.80/10**	**0.69/10**

Table 5.4 The data entities in each cluster are sampled from exponential distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent.

Comparison of L_2, L_1, AL_2, and AL_1 with the generated clusters at clusters=9 and cluster shape=spherical

	Estimated number of clusters		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	7.70/12	8.20/13	7640.98/11	1044.68/10	0.83 / 8	0.39 / 8
L_1	**9.60 / 7**	**8.20/13**	**7618.69/10**	**1039.43 / 9**	**0.93 / 8**	**0.55 / 9**
AL_2	8.12/10	8.15/10	7630.48/12	1045.36 / 8	0.82 / 9	0.41/12
AL_1	**9.54 / 7**	**8.80/12**	**7615.85/10**	**1038.54 / 8**	**0.96 / 8**	**0.56/10**

Table 5.5 The data entities in each cluster are sampled from exponential distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent.

Comparison of L_2, L_1, AL_2, and AL_1 with the generated clusters at clusters=21 and cluster shape=spherical

	Estimated number of clusters		Unweighted distance between centroids		Adjusted Rand Index	
	LaS	SmS	LaS	SmS	LaS	SmS
L_2	15.74/11	16.54/10	26503.32/10	3021.45/10	0.87/11	0.38/11
L_1	**20.52/10**	**21.12/10**	**26401.96/12**	**2941..52/10**	**0.91/11**	**0.41/11**
AL_2	20.34/10	20.45/11	26543.85/11	3104.52/10	0.89/11	0.40/10
AL_1	**21.29/10**	**20.95/10**	**26429.54/11**	**3012.85/11**	**0.95/10**	**0.45/10**

Table 5.6 The data entities in each cluster are sampled from exponential distribution. The average values of evaluation criteria for the large and small between-cluster spread factors (in columns LaS and SmS, respectively) as presented in Table 3.1. The standard deviations are after slash, per cent.

5.3 Summary

In this chapter, we compare L_1 version with L_2 version of iK-Means method to see (1) whether if there is any difference between them and (2) which data structure is more suitable for L_2 version. We run a series of experiments by comparing the L_1 version partitions with L_2 version partitions, not comparing those partitions with the generated ones. It clearly shows the two versions of iK-Means are different by comparing the cluster recovery. In order to answer the second question, we run a series of experiments where the data entries in each cluster are independently generated from exponential distributions. We compare the clustering of L_1 and L_2 versions of iK-Means with the generated clusters and it shows that L_1 performs the best on all evaluation criteria.

Chapter 6

Application of L_1 and L_2 K-Means to Gene Expression Data

From the experimental results described above, iK-Means methods outperform other methods in terms of the centroids and cluster recovery but not on the number of clusters, which can be solved by the HT-adjusted version iK-Means methods. To find the patterns of gene expression data has become one of the most popular research fields and many authors have applied various clustering techniques on gene expression data, for example, Dudoit and Fridlyand (2002), Shen et al. (2005), etc. Obviously, iK-Means can be applied for clustering gene expression data too, but this is not exactly our goal. We are interested in utilizing the discrepancies between L_1 and L_2 methods for a biological meaningful problem. Such a problem emerged on research of Prof. B. Chain, Virology Department, UCL. Their team has produced two data sets based on the same genes and gene fragments, one related to gene expression in dendritic cells and the other in cancerous dendritic cells. We are indeed interested in finding which genes differ between DC and Mutz3. This is because many people would like to use dendritic cells derived from leaukaemias to stimulate an immune response which could potentially control the leukemia itself. However, it doesn’t work. One reason may be that for some reason dendritic cells which are derived from leukaemic cells are different (and not as good as) dendritic cells from normal monocytes (i.e. normal blood cells which are not cancerous). So we want to know, at a molecular level, what differences there are between a "normal" dendritic cell (which we call DC) and a cancerous dendritic cell (which we call Mutz33). To answer this question computationally, one first needs to select a data pre-processing option, because the gene expression data is subject to many potential failings of the expression process. As proven in Section 5.1, the L_1 and L_2 methods may produce different results --- we are going to exploit this by using those parts of the found clusters that are stable between the two and it can be used for analyzing the difference in gene activity across gene expression data in different cells. Since our gene expression data contains highly correlated signals, we develop a special normalization method for separation of the physical condition of the gene expression experiment from its biological part, the pivot-based normalization (PBR). The following section is organized as follows: we briefly described the existing literature of pre-processing in Section 6.1, pre-processing, clustering dendritic and tumor cells gene expression data using iK-Means methods and we compare the results with three different pre-processing methods in Section 6.2.

6.1 The issue of gene expression data pre-processing

DNA microarrays are a technology to investigate the expression levels of thousands of genes simultaneously, which is a great improvement over the traditional genomic research that has focused on the study of single gene, single protein, or a single reaction at a time. The thousands of affixed DNA sequences known as probes can be placed on a single DNA microarray. The probes are normally oligonucleotides or complementary DNA (cDNA) in spotted microarrays and the probes are short oligonucleotides sequence in oligonucleotide microarrays for matching part of the sequences to known or predicted genome portions. A well-known DNA microarray manufacturer Affymetrix produces the sequence for oligonucleotide array and now the term “Affymetrix” is not only a company name but also the data obtained from an oligonucleotide array. Each spotted microarray has been hybridized with cDNA from two samples (e.g. disease tissue vs. healthy tissue or experimental data vs. synthetic data in our analysis) labeled with different fluorescent dyes. A two-colored dye is used for each sample so that we can tell the two samples apart on the array. Fluorescent dyes include Cy5 and Cy3, referred to by convention as red and green, accordingly. The data is represented as a matrix with rows (genes) and columns (different conditions or time). The data which measures the expression level of genes in a certain condition at different instances of time is called temporal data (versus non-temporal data).

In order to reduce the ill-effects of various data corruption circumstances, data pre-processing is necessary for the effective analysis of gene expression data. However, there may be some inconsistencies after pre-processing, for example, different scales among different conditions of gene expression data, gene expression data obtained from different arrays, replicated gene expression data or etc. Many techniques and approaches have been presented to tackle the above inconsistencies. They have been reviewed and, partly, compared in a number of articles (Yang et al. 2002, Bolstad et al. 2003, Park et al. 2003, Pandey et al. 2007, etc). However, as we limit ourselves with more specific data types akin to those developed in the Virology Department of University College London (UCL), we found convenient to categorize the pre-processing procedures being applied in the gene expression data analysis as follows:

A. Standardization, that is, to minimize various variations between rows and/or columns of the data

B. Functional transformation, that is, to perform a functional transformation, for example, logarithmic transformation (Yang et al. 2002), sigmoid-based normalization (Pandey et al. 2007), etc ; and

C. Filtering outliers, that is, to filter the differently expressed genes (Saviozzi and Calogero 2003, Jiang et al. 2004);

Each of these admits different approaches that can be systematized, based on the body of published literature, as follows:

A. Standardization:

A.1 Column normalization:

A.1.1 by itself: dividing by scaling values, for example, Z score based normalization (Jiang et al. 2004, Tamayo et al. 1999, Cheadle et al. 2003, Pandey et al. 2007), optimization-based genetic algorithm (Shmulevich and Zhang 2002), etc;

A.1.2 by green from the same column: LOWESS/LOESS normalization for cDNA arrays (Yang et al. 2001, Quackenbush 2002), or etc;

A.1.3 by replicates comparison: Combining replicates (Draghici 2003), parametric normalization (Liggett 2006), pivot-based with removals normalization method (Chiang and Mirkin 2008), etc;

A.1.4 by samples comparison: centralization (Zien 2001), etc

A.2 Row normalization: quantile normalization (Bolstad et al. 2003), etc;

One of the most commonly used column normalization methods is called Z score (Tamayo et al. 1999, Cheadle et al. 2003, Pandey et al. 2007), that is, to shift the red or green signal values in a vector by the mean of their values and scale them by the standard deviation. Z score is popular for the transformation of temporal gene expression data (Tamayo et al. 1999). Bergmann et al. (2003) proposed an algorithm for analyzing the gene expression data, that is, to iteratively refine the genes and conditions until they match pre-defined transcription modules and the normalization method they used is mathematically equivalent to Z score. Shmulevich and Zhang (2002) proposed a normalization procedure: apply the optimization-based genetic algorithm then binarize the data. The optimization-based genetic algorithm chooses scaling parameters so that the sample mean and standard deviation are minimized and ensure that the maximum gene expression levels after normalized is larger than both the maximum un-normalized gene expression levels and one. Those genes with high expression levels are binarized to 1, otherwise 0, where the threshold is the first difference between sorted gene expression levels exceeding a pre-specified value.

However for cDNA arrays, Z score adjusts the overall intensities of the gene expression data but does not address the dye non-linearity (Draghici 2003). Therefore, there are several normalization techniques which are specifically for the cDNA or Affymetrix data, for example, use LOWESS/LOESS normalization (Yang et al. 2001, Quackenbush 2002) to eliminate the intensity dependent bias for cDNA data, the detection calls (Draghici 2003) implemented in several Affymetrix analysis software, and etc. LOWESS normalization stands for locally weighted scatter plot smoothing and LOESS stands for locally weighted polynomial regression. Both of them use the linear polynomial function to normalize the red signals by the green signals, but LOESS uses a quadratic polynomial to solve the over-fitting and the excessive twisting and turning problems. Berger et al. (2004) proposed a LOWESS-based method, which aims for choosing the best fraction used in the local regression so that the mean-squared difference function between LOWESS estimates and normalization reference level is minimized. This fraction is between 0 and 1, and in general, the smaller the value, the more that the LOWESS curves follow data points. Piece-wise normalization (Draghici 2001) is another LOWESS-similar normalization method, which improves the computational efficiency of LOWESS. Detection calls characterize genes into three states using a non-parametric hypothesis testing approach as either present (P), absent (A), or marginal (M), which means that the expression level is higher, lower, or similar to the minimum detection level, respectively. Other various normalization methods specifically either for cDNA data or Affymetrix data are presented in several publications, such as Li and Wong 2001, Li and Wong 2001a, Wang et al. 2002, Finkelstein et al. 2002, etc.

Because of the large amount of noise with the microarray data, scientists tend to repeat the microarray experiments. It is convenient to combine all the replicates to a unique value in some circumstances; however, the loss of information may happen by using the above normalization and transformation methods. Two approaches described in the book by Draghici (2003) may be attempted to solve the loss of information problem. The first approach is to store the parameters of the distribution of the original values and the second approach is to filter out the outliers. Both of these approaches need to calculate the mean, the standard deviation, and other parameters of the distribution of the original data. Data points outside some given interval are considered as outliers and will be eliminated. The parameters of the remaining data will be re-calculated and the process is iterated until no outliers can be found.

In some cases, the variability in the red signals may be partly attributed to the physical conditions of the experiments, and this can be captured in the green signals. If this hypothesis is true, then we could improve the reproducibility of red signals, by taking into account the physical conditions as caught up on the green signals. Liggett (2006) proposed a parametric normalization approach by normalizing the red by the physical condition, but Liggett used the results of factor analysis of greens to normalize the reds, which requires user-specified parameters. If the replicates show highly linearly correlated, the parametric normalization approach may not be suitable. Therefore we proposed a pivot-based normalization method, which can capture the differences among replicates by using linear regression analysis[1]. We consider one replicate green signal, g_p, as a pivot and express others as linear functions of g_p. Specifically, if a replicate green signal g can be expressed as $g=ag_p+b$, where a and b are constants, then it is reasonable to assume that these constants take into account the difference in physical conditions that produced signals g_p and g. If the conditions would have been the same, the

[1] A linear regression line has an equation of the form Y=aX+b, where X is the explanatory variable and Y is the dependent variable, a is the slope and b is the intercept. In order to obtain a and b, the mean and standard deviation of X and Y are computed, denoted as $\overline{X}$, $\overline{Y}$, σ_X, σ_Y and the equations for calculating a and b are as follows: $a=\theta * \sigma_Y / \sigma_X$ and $b=\overline{Y} - a * \overline{X}$, where θ is the correlation coefficient between X and Y.

signals should be the same, g_p=g. Therefore, to take into account the difference in the physical conditions over replicas g_p and g, one should normalize g into g'=(g-b)/a. The same normalization should be applied to the red signal in g-replica to make it compatible, over physical conditions, over the red in g_p-replica. This pivot-based linear regression normalization algorithm with removals is implemented as shown in the box below, along with some data cleaning steps, so that the regressions found in the green can be applied to the red.

Pivot-based linear regression normalization algorithm

A. Check the correlations between the green replicas and select that one that makes the highest summary correlation with the other as the pivot, g_p, and make regressions of each of the others, g_o, over the pivot

B. The red signals are pivot-adjusted according to the regressions: the red corresponding to the green pivot remains as is, and subtracting the intercept and dividing by the slope adjust two other red replicas.

The Quantile normalization method proposed by Bolstad et al. (2003) is a popular normalization method commonly used for Affymetrix data, which is available in the MATLAB bioinformatics toolbox[2]. It takes the means across rows (gene) of a column-sorted gene expression data matrix and assigns the mean to each element in the row to get a quantile equalized matrix. Then the quantile equalized matrix is rearranged to have the same order of the original gene

[2] http://www.mathworks.com/products/bioinfo

expression data matrix. This procedure makes the distribution of the expression level of each array identical and the drawback might be some loss of information during the normalization process. A number of papers have been presented for row (gene) normalization, such as Workman et al. (2002), Lepre et al. (2004), etc.

Zein et al. (2001) proposed a sample-based normalization method, centralization, which is applied after the replicates normalization methods. They use a maximum likelihood approach to find a scaling factor after computing the probability distributions for the pairwise scaling for every pair of the sample measurement. Based on their assumption that most genes are not or only moderately regulated or the numbers of genes which up-regulated or down-regulated are approximately the same, centralization reproduces the results of other normalization methods.

B. Functional transformation

The logarithmic transformation has been widely used in microarray data pre-processing (Yang et al. 2002) because it is convenient for later data analysis. Ease of interpretation is another well-known reason for the logarithmic transformation and the log transformed data will be more meaningful to biologists. Sigmoid-based normalization methods (Pandey et al. 2007) are based on the sigmoid function and double sigmoid function which take into account not only the outliers but also the distribution of the gene expression data. Pandey et al. (2007) modify the sigmoid function so that it considers the gene expression value distribution and smooths the center of the distribution. The main parameters of the modified sigmoid function are the mean and variance of the normal distribution. These functions have been applied to other studies but firstly applied to bioinformatics domain for gene expression analysis and they perform very well for non-temporal data (Pandey et al. 2007).

C. Filtering outliers

A number of papers have been published to establish the importance of filtering the genes that have significantly different expression patterns between two data sets; for example, Nimgaonkar et al. (2003) report 27% of negative correlation between the two data sets they use. Therefore, several filtering methods have been published. Saviozzi and Calogero (2003) firstly remove the genes that have a similar expression level with the background and then further remove the genes that show low hybridization quality, generated from the dCHIP software (Li and Wong 2001a). Jiang et al. (2004) apply the student's t test to filter the outliers and the p-value is set to be 0.00001.

Several evaluations of normalization methods have been published, for example, Park et al. (2003) compared seven normalization methods for replicates and found that the normalization methods perform similarly when the original data has a high linear correlation, the intensity-dependent normalization method performs better among others and the performance of intensity-dependent linear and non-linear methods are quite similar. Steinhoff and Vingron (2006) summarized several normalization approaches and found that the choice of the normalization methods depends on the gene expression. Ma and Qin (2006) evaluated different normalization strategies on heritability estimation, which are implemented in software such as MAS 5.0 (Mircroarray Analysis Suite), dChip, or RMA (Robust Multi-array Average), for oligonucleotide data and found that the RMA method performs well in cross-chip normalization with the highest heritability among the three methods. RMA creates an expression matrix from Affymetrix data. The raw intensity values are standardized, log2 transformed and then quantile normalized. Next a linear model is fit to the normalized data to obtain an expression measure for each gene set. Pandey et al. (2007) recently presented an evaluation over several normalization methods, such as Z score, quantile and sigmoid normalization, and their results show that different data sets and the type of functional information being predicted can significantly affect the performance of different normalization methods.

6.2 L_1/L_2 consistent genes analysis

The experiments in Chapter 5 have shown that L_1 and L_2 methods are different; therefore, it would be reliable to utilize the difference of these two methods in a meaningful problem: given these gene expression data, find those genes that re weak and those that are strong. The biologists tend to replicate the experiment several times. The problem is extracting consistent patterns from the data. To do so, we propose the following method:

1. Normalization on gene expression data
2. Clustering on one set of the normalized gene expression data
3. Selecting the weak and active genes according to the centroids of the genes consistent between L_2 and L_1 methods

4. Clustering on another set of the normalized gene expression data
5. Selecting the weak and active genes according to the centroid sets of genes consistent over L_1 and L_2 methods

We apply this method to two gene sets: one is DC (dendritic cells) and another one is Mutz3 (cancerous dendritic cells). Each of the DC and Mutz3 gene sets is represented by data in three different versions. Experimental data (red vs. green signal) is contained in each version and there are 37358 genes in each version.

The hypothesis of these gene sets is that the variability between the versions may be partly attributed to the physical conditions of the experiments, and this can be captured by analyzing differences in the green signal, which is supposed to be independent of the substantive variability. For these data sets, the high level of correlation (on the average level of 0.95) between green signals is observed both in Mutz3 and DC, which ensures that there is a substantial linear component in the relations between the signals. Because of this feature, we would like to apply the pivot-based linear regression normalization method described in Section 6.1. We consider one of the three green signals, g_p, as a pivot and express two others as linear functions of g_p. Specifically, if green signal g in one data set can be expressed as $g=ag_p+b$, where a and b are constants, then it is reasonable to assume that these constants take into account the difference in physical conditions that produced the signals g_p and g. If the conditions would have been the same, the signals should be the same, $g_p=g$. Therefore, to take into account the difference in the physical conditions over replicas g_p and g, and make g comparable to g_p, one should normalize g into $g'=(g-b)/a$. The same normalization should be applied to the red signal in g-version to make it comparable, over physical conditions, over the red in g_p-version. This pivot-based linear regression normalization algorithm is implemented as shown in the box below, along with some noise cleaning steps, named as pivot-based with removals (PBR) normalization method, so that the regressions found in the green signal can be applied to the red signal.

After normalization, we utilize the discrepancies of L_1 and L_2 methods to find out the cluster consistent genes, that is, the cluster contents in the clusters that are present among the results of both methods. The normalization and clustering results are presented in Section 6.2.1. We then compare the clustering results of pivot-based linear regression normalization method with the clustering results of three pre-processing methods. This is carried out in Section 6.2.2.

Pivot-based regression normalization algorithm with removals

A. Remove all the genes where the expression level reaches 100000 or more at least on one replicate

B. Check the correlations between the green replicas and select that one that makes the highest summary correlation with the other as the pivot, g_p, and make regressions of each of the others, g_o, over the pivot, after cleaning the 2.5% of the high-value outliers in the distribution of $\max(g_p/g_o, g_o/g_p)$

C. The red signals are pivot-adjusted according to the regressions: the red corresponding to the green pivot remains as is, and subtracting the intercept and dividing by the slope adjust two other red replicas.

6.2.1 Pivot-based with removal normalization results

For DC, we found that version 2 of green, g2, should be the pivot, after the high expression level is removed, and the corresponding regressions after a further 2.5% removal are: g_3=1.29*g_2-2389.1 and g_1=1.55*g_2-4228.8. For Mutz3, it is version 3 of green, g_3, that should be the pivot, after removal of genes with the high expression level, and the corresponding regressions, after a further 2.5% removal, are: g_2=1.28*g_3-1838 and g_1=1.21*g_3-1617.8.

After the double removals described above, there remain 35452 genes in DC and 35510 genes in Mutz3.

Our clustering methods iK-Means Least Square (L_2) and Least Moduli (L_1) are applied to cluster these data sets (over three features corresponding to the pivot-regression normalized red signals). For DC, L_2 produces only two clusters, of 35097 and 355 genes, respectively, and L_1 produces three clusters, containing 34865, 577 and 10 genes, respectively. For Mutz3, a similar story: three clusters for L_2 algorithm (30862, 3638, and 1010 genes) and two clusters for L_1 algorithm (33069 and 2441 genes). This means that the distributions of the signals are so much skewed to the left that even intelligent K-Means cannot properly separate the genes according to this data.

Therefore, we take logarithms of all the original signals, and carry on the same procedure as described above, with thus transformed data. The only difference is in the cleaning of noise: because of using the logarithm transformation, one needs to subtract rather than divide to perform the 2.5% removal operation. The correlations between green signals are very high (but not higher, as it should be expected) again. For DC, we take version 3 of green, g_3, as the pivot after removing the higher expression level genes, and the corresponding regressions after further 2.5% removals are: $g_2=0.93*g_3+0.73$ and $g_1=0.93*g_{39}+1.12$ (Note, these are for logarithms!). For Mutz3, we take version 3 of green, g_3, as the pivot after removing the higher expression level genes, and the corresponding regressions after further 2.5% removals are: $g_1=0.98*g_3+0.31$ and $g_2=1.02*g_3+0.11$. There are 35539 genes of DC and 35628 genes of Mutz3 left after the cleaning.

In spite of the logarithm transformation, Mutz3 remains tight against our clustering methods and gives us again only two clusters. Thus we concentrate on clustering DC, which makes more clusters. Since our L_1 and L_2 methods tend to produce rather different results, we consider those clusters valid that are present among the results of both methods – we refer to their contents as cluster consistent genes.

There are 35539 genes of DC and 35628 genes of Mutz3 left after double-removal of the log-transformed gene sets and Table 6.1 and 6.2 present centroids (averages) of clusters of DC found with L_2 and L_1 methods, respectfully. According to Table 6.1 and the averages, L_2 clusters 2 and 5 contain active genes, cluster 1 weak and cluster 3 medium expression level, which are of interests to us. Similarly, the L_1 centroids in Table 6.2 give us clusters 2 and 6 of active genes, cluster 1 weak, and 3 medium expression level.

Cluster centroids of log-transformed DC data obtained using the L_2 method

Cluster number	R1	R2	R3
1	5.18	4.68	4.86
2	10.03	11.06	10.79
3	7.02	7.67	7.44
4	8.14	9.08	8.65
5	9.19	10.34	9.72
6	8.76	9.76	9.00
7	9.24	8.85	10.06
8	8.06	9.85	9.59
9	8.46	9.12	9.33
10	8.90	9.87	9.78
11	8.56	9.60	9.90
12	9.18	9.42	9.32
13	8.50	9.88	9.39
14	8.72	9.57	9.36
15	8.76	9.37	9.64
16	8.43	9.48	9.50
17	8.71	9.57	9.62

Table 6.1 The values are the averages of clusters of DC found with L_2 method. Three columns represent three versions.

Cluster centroids of log-transformed DC data obtained using the L_1 method

Cluster number	R1	R2	R3
1	5.03	4.69	4.84
2	9.99	11.15	10.73
3	6.70	7.63	7.41
4	8.14	9.05	8.66
5	8.78	9.16	9.31
6	9.20	10.30	9.74
7	8.17	9.70	9.27
8	9.50	9.66	9.18
9	8.78	9.75	9.02
10	9.10	9.65	9.81
11	8.61	10.07	9.78
12	8.34	9.54	9.72
13	8.57	9.40	9.27
14	8.74	9.87	9.42
15	8.65	9.37	9.73
16	8.87	9.68	9.67
17	8.48	9.72	9.58
18	8.61	9.62	9.83
19	8.26	9.39	9.54
20	8.74	9.61	9.39
21	8.64	9.69	9.52
22	8.73	9.54	9.59

Table 6.2 The values are the averages of clusters of DC found with L_1 method. Three columns represent three versions

To compare these genes we derive the confusion matrix of the overlaps between them (Table 6.3). This table shows that the active, weak and medium genes over L_1 and L_2 are almost identical. In the follow steps of the experiment, we will use the cluster intersection of cluster number 1 of L_2 and cluster number 1 of L_1, cluster number 2 of L_2 and cluster number 2 of L_1, cluster number 5 of L_2 and cluster number 6 of L_1 and cluster number 3 of L_2 and cluster number 3 of L_1 for the further analysis, that is, the cluster-consistent genes of DC data.

Confusion matrix between the results of L_2 and L_1 methods of DC data

		L_1					
		1	3	2	6	others	Total
L_2	1	20345	180	0	0	0	20525
	3	39	9326	0	0	120	9485
	2	0	0	1257	24	0	1281
	5	0	0	12	750	109	871
	Others	0	18	5	14	3340	3377
	Total	20384	9524	1274	788	3569	35539

Table 6.3 The number of cluster-consistent genes of DC data. The number of genes of cluster number 1 of L_2 method and cluster number 1 of L_1 method are 20345 (weak), cluster number 2 of L_2 method and cluster number 2 of L_1 method are 1257 (very active), cluster number 5 of L_2 method and cluster number 6 of L_1 method are 750 (active) and cluster number 3 of L_2 method and cluster number 3 of L_1 method are 9326 (medium).

Now we are going to take a look at the distributions of Mutz3 expression levels within each of these cluster-consistent genes. In order to find out which gene differs between DC and Mutz3, we take the cluster-consistent genes in DC, which are 20345 (weak), 1257 (very active) and 750 (active), and 9326 (medium) genes and do clustering on the corresponding 35628 pivot-based linear regression normalized logarithmic Mutz3 data. The numbers of genes for clustering are 18962, 1246, 748, and 9243 genes, accordingly (since some of the genes have been cleaned out before).

The following tables, Table 6.4 to 6.9, show the clustering centroids (averages) and partitions (number of genes) of L_2 and L_1 method in each cluster of the 18962, 1246, 748 corresponding Mutz3. Since the extreme cases are of interests to us, clustering results of 9243 medium genes are not listed here. From the tables, these two methods show very similar results, so we take the cluster intersections, that is, the cluster-consistent genes.

The 18962 genes are the cluster-consistent genes which shown weak in DC clusters, and the centroids (averages) in clusters 2 and 5 shown in Table 6.4 and 6.5 are very high, which means the genes in cluster 2 and 5 are very weak in DC but very active in Mutz3. The sizes of these genes are 26 and 10 genes respectively; these 36 genes are listed in Appendix A as genes that are weak in DC and active in Mutz3. A similar analysis is done with the very active 1246 cluster-consistent active genes in DC, where the clustering results are shown in Table 6.6 and 6.7, and found out that the centroids of clusters 2 and 5 are quite low and the number of cluster-consistent genes in clusters 2 and 5 is 8 and 6 genes respectively. These genes are listed in Appendix A, as genes that are very active in DC and weak in Mutz3. The same procedure is done with the active 748 cluster consistent genes in DC, where the clustering results are shown in Table 6.8 and 6.9, and found the centroids in cluster 1 of both clustering methods are very low. This means the cluster-consistent genes of cluster 1 are active in DC and weak in Mutz3, which are listed in Appendix A as active in DC and weak in Mutz3 and the number of the genes is 6.

Cluster centroids obtained using the L_2 method of the 18962 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	4.57	4.62	4.63	15624
2	10.33	9.64	9.70	27
3	6.67	6.44	6.43	3191
4	8.38	8.23	7.55	76
5	9.17	8.85	8.92	17
6	8.17	7.61	8.24	27

Table 6.4 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

Cluster centroids obtained using the L_1 method of the 18962 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	4.56	4.61	4.52	15314
2	10.17	9.55	9.55	26
3	6.52	6.28	6.35	3462
4	8.14	7.93	7.58	120
5	9.23	9.11	8.61	15
6	8.93	8.56	8.73	10
7	8.40	7.88	8.44	15

Table 6.5 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

Cluster centroids obtained using the L_2 method of the 1246 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	10.72	10.23	10.73	1107
2	5.89	5.64	6.31	8
3	9.13	8.58	9.58	108
4	8.96	8.45	6.36	8
5	7.17	6.79	7.20	7
6	8.50	8.25	7.73	8

Table 6.6 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

Cluster centroids obtained using the L_1 method of the 1246 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	10.76	10.28	10.75	1084
2	6.42	6.04	6.69	9
3	9.35	8.83	9.71	130
4	8.93	8.46	6.27	8
5	7.19	6.88	7.13	8
6	8.21	8.02	7.57	7

Table 6.7 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

Cluster centroids obtained using the L_2 method of the 748 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	5.00	5.13	5.29	9
2	9.90	9.48	9.88	602
3	8.77	8.34	9.07	110
4	7.41	7.20	7.32	17
5	8.14	8.04	8.11	10

Table 6.8 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

Cluster centroids obtained using the L_1 method of the 748 corresponding Mutz3

Cluster Number	R1	R2	R3	Number of genes
1	4.27	4.50	4.77	6
2	9.88	9.46	9.89	557
3	6.76	6.60	6.50	9
4	7.81	7.44	8.32	15
5	8.91	8.58	9.20	148
6	7.60	7.57	7.37	6
7	8.31	8.15	8.14	7

Table 6.9 R1, R2, and R3 represent three versions of the data. The centroids are log transformed.

6.2.2 Comparing clustering results with LOESS normalization method

Since there are many methods for normalizing red signals over green signals, we would like to compare the clustering results at different normalization methods. Therefore, we applied LOESS normalization method to the same DC and Mutz3: pivot-based normalization without removals (PB), one of the most popular normalization methods: intensity dependent normalization (LOESS) method (Yang et al. 2001). The data for normalization is log transformed because the cluster analysis shown in the previous section suggests that the distribution of the signals is left skewed and in order to make these results comparable, there will be no removals on the pivot-based normalization whereas the rest of the algorithm is the same as described in Section 6.2.

The intensity dependent normalization (Yang et al. 2001) is done by the following equation:

$$\log R/G \rightarrow \log R/G - c(A) \qquad \textbf{(11)}$$

where R and G are considered as the red and green signal respectively and c(A) is the LOESS fit to the M=log(R/G) vs. A=log($\sqrt{R*G}$) plot. This normalization is one of the most popular methods for gene expression data normalization; the software is freely available in the Matarray software (Venet 2003). Many publications and books have suggested that the MA plot should be used for solving the dye bias which depends on the spot intensity. Figure 5.1 shows the MA plots with LOESS fit of the three replicates of DC datasets from the left to the right respectively, and it clearly shows that a linear normalization is required because these curves are around a horizontal line near 0, which matches the fact found in the previous section that the data are highly linearly correlated. The MA plots of the Mutz3 data are quite similar to the MA plots of DC data. In this case, for each entity in the dataset we apply the linear LOESS curve to a subset of the data. This parameter usually lies between 0.2 and 0.5 for most LOESS applications and is set to be 0.5 as default in the Matarray software.

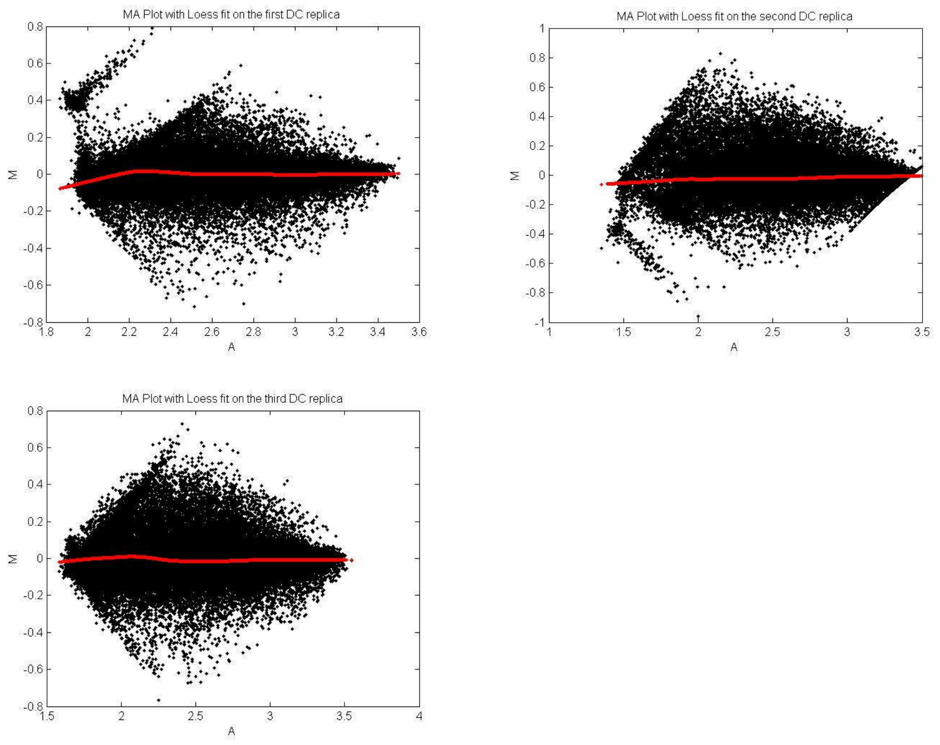

Figure 5.1 MA plots with loess fit on the three replicates of DC datasets. The red line is the loess curve and the plots show the first, second and third replicate of the DC datasets from left to right, respectively.

In order to find the genes that are different between Mutz3 and DC, we apply the same procedure as described in the previous section:

1. Normalization of both log-transformed DC and Mutz3
2. Clustering on the normalized DC data
3. Selecting weak and active DC genes according to the centroids of sets of genes consistent between L_2 and L_1 methods
4. Clustering on the normalized Mutz3 data of the corresponding sets of weak and active DC genes
5. Selecting the weak and active Mutz3 genes according to the centroids of sets of genes consistent over L_2 and L_1 methods

The following table presents the numbers of weak and very active DC genes that have been selected based on the overlaps according to the centroids of both L_2

and L_1 methods because of the identical results found by using of both methods. The other genes are not listed or shown because we are only interested in the extreme cases.

The number of weak and active DC genes found at three normalization methods

	Weak	Very active
Pivot-based without removals (PB)	21648	1185
Pivot-based with removals (PBR)	20345	1257
Intensity dependent normalization (LOESS)	22161	1284

Table 6.10 The weak and active is based on the cluster centroids of L_2 and L_1 method.

We then do clustering on the corresponding normalized Mutz3 data sets according to each of the normalization methods and Table 6.11 shows the numbers of very active and weak Mutz3 genes based on the sets of consistent genes according to the centroids of both clustering methods. These genes are of interest to us because these are the genes that differ between DC and Mutz3, that is, take PB method as an example, there are 28 genes that are weak in DC but very active in Mutz3 and 5 genes that are very active in DC but weak in Mutz3, according to PB normalized data.

The number of very active and weak Mutz3 genes for three normalization methods

	Very active	weak
Pivot-based without removals (PB)	28	5
Pivot-based with removals (PBR)	26	8
Intensity dependent normalization (LOESS)	33	12

Table 6.11 The weak and active is based on the cluster centroids of L_2 and L_1 method.

Among the genes that are weak in DC but very active in Mutz3, there are 22 genes in common among the three normalization methods. We compare the uncommon genes of each method with those at other two methods and can easily see the reasons why these genes are selected in one method but not the other. All the different reasons for a gene having not been selected at a normalization method categorized into four:

(1) medium expression level on DC or the corresponding Mutz3;

(2) cluster-inconsistence of the gene on DC data;

(3) cluster-inconsistence of the gene on the corresponding Mutz3 data;

(4) removed as an outlier.

Table 6.12 shows the number of genes that are not selected for different reasons, for example, there are 6 and 3 genes that are not selected in PB method but selected in other methods because these genes are classified as medium expression level on DC or Mutz3 and cluster-inconsistent genes on Mutz3 data using PB method respectively. Table 6.13 shows the gene numbers of those genes categorized in Table 6.12 and detailed lists can be found in Appendix B.

Consider, for example, the differences between PB and PBR methods: we found that 3 genes, that have been removed as outliers at PBR method are selected at the PB method. Of these, 2 genes are not selected at the PBR method because these genes are cluster-inconsistent on Mutz3 data normalized by PB method.

The number of genes that are not selected for different reasons in weak DC and very active Mutz3 case

	Medium DC or Mutz3	Cluster-inconsistent genes on DC data	Cluster-inconsistent genes on Mutz3 data	Removed as outliers
PB	6	0	3	0
PBR	3	0	1	3
LOESS	2	1	0	0

Table 6.12 There are total 11 genes that are not selected due to the medium expression level of DC or Mutz3. There is only 1 gene that is not selected due to the cluster inconsistency on DC data. There are 4 genes that are not selected due to the cluster inconsistency on Mutz3 data. 3 genes are removed as outliers.

The corresponding gene numbers according to Table 6.12

	Medium DC or Mutz3	Cluster-inconsistent genes on DC data	Cluster-inconsistent genes on Mutz3 data	Removed as outliers
PB	6121 17501 19105 25435 34293 42349	N/A	262 21187 42205	N/A
PBR	25435 34293 42349	N/A	262	16620 21827 44198
LOESS	6121 19105	42349	N/A	N/A

Table 6.13 The gene number lists that are not selected due to four reasons.

A similar analysis can be done for the very active DC and weak Mutz3 genes: there is just 1 gene common among all four normalization methods. Table 6.14 shows the numbers of genes that have not been selected. For example, there are 2 and 4 genes that are not selected at PB method but selected in other methods because these genes are classified as cluster inconsistent on DC and Mutz3 data respectively. Table 6.15 shows the gene labels of those genes in Table 6.14; their detailed lists can be found in Appendix B. Comparing between PB and PBR methods, one can see that there are 4 genes that are inconsistent because 3 of them are cluster inconsistent genes on Mutz3 data and 1 of them is cluster-inconsistent on DC data for PB method.

The gene lists of weak DC/very active Mutz3 and very active DC/weak Mutz3 of the two normalization methods are in Appendix B. Two comparison tables of these genes with normalized expression levels obtained from different normalization methods and log-transformed original values are available online[3]. The tables show the common part of all methods followed by the genes obtained under different normalization methods.

The number of genes that are not selected for different reasons in active DC and weak Mutz3 case

	Medium DC or Mutz3	Cluster-inconsistent genes on DC data	Cluster-inconsistent genes on Mutz3 data	Removed as outliers
PB	0	2	4	0
PBR	0	1	1	2
LOESS	0	0	0	0

Table 6.14 There are 3genes that is not selected due to the cluster inconsistency on DC data. There are 5 genes that are not selected due to the cluster inconsistency on Mutz3 data. 2 genes are removed as outliers.

The corresponding gene numbers according to Table 6.14

	Medium DC or Mutz3	Cluster-inconsistent genes on DC data	Cluster-inconsistent genes on Mutz3 data	Removed as outliers
PB	N/A	28352 41755	2344 8538 16552 29926	N/A
PBR	N/A	41755	16552	23361 33774
LOESS	N/A	N/A	N/A	N/A

Table 6.15 The gene number lists that are not selected due to four reasons.

[3] http//www.dcs.bbk.ac.uk/~mingtsochiang/gene/

6.3 Summary

In this chapter, the difference between two versions of iK-Means is utilized in order to analyze real-world data sets. We proposed the two versions of the pivot-based normalization method due to the hypothesis and the high correlation of the gene expression data. After normalization, two versions of iK-Means are applied to the normalized data and an algorithm for finding sets of genes differing in gene activity over difference cells by using L_1/L_2 consistency is proposed in Section 6.2. We would like to compare the differently expressed genes found by using the two versions of the pivot-based normalization method with LOESS normalization method. We found that the genes found by the LOESS method cover other methods, whereas the pivot-based methods only captures the extreme cases, thus leading to a rather conservative estimate

Chapter 7

Conclusion and Future Work

Overall, the impact of this work to the body of knowledge can be summarised as follows. A computational model for generation of data with a Gaussian cluster structure controlled by just two “spread” parameters is proposed. It is experimentally shown that popular methods for choosing the number of clusters in K-Means clustering, such as Gap statistic, are inferior to intelligent K-Means method on data of this structure, in either iK-Means version considered, L_1 and L_2. Based on the experimental results, a new version of iK-Means, combining it with the Hartigan’s rule, is proposed and verified. We also show, additionally using exponential cluster structures, that the two versions of iK-Means, L_1 and L_2, may lead to rather differing results. This has been utilised in application to a problem in bioinformatics by using only L_1 and L_2 consistent cluster parts. The problem concerns analysis of differences in gene activities in different types of condition (cancer or not) over the gene expression data. To normalise the data, a systematic review of the methods has been conducted and a novel normalisation method suitable to the task was proposed and utilised.

The subject of interest is the intelligent K-Means method, iK-Means, that determines the number of clusters by sequentially extracting “anomalous patterns”, in two versions: least squares (L_2) and least moduli (L_1). We are interested to see whether there are any differences between these two versions and if there are, then what are the specific data structures in which one version is better than another.

In designing the experiments, one needs not only a good data generation model but also comprehensive evaluation criteria. Three evaluation measures, including the number of clusters, centroids recovery and four cluster recovery coefficients, are implemented. We found that the unweighted distance between centroids much better correlates with the cluster recovery than its cluster-size-weighted version, which leads us to rejection of the latter as an evaluation index.

Our experimental results indicate that:

(a) In general, all tested methods are not sensitive to the relative cluster sizes. Both the cluster recovery and centroid recovery are better at the large between-cluster spreads. The centroid recovery of all methods slightly improves when moving from elongated clusters of different variances to spherical clusters of a constant variance; the cluster recovery follows this pattern too, but the effects are minor on this aspect;

(b) L_1 and L_2 version of iK-Means method do lead to different results, and in general, L_2 is favoured by Gaussian clusters whereas L_1 is favoured by exponential clusters;

(c) Hartigan's rule "of thumb" HT outperforms the others, in most tests, in terms of the number of clusters, and it is good in terms of cluster recovery at the large between-cluster spreads; the other methods under consideration form consistent patterns of, typically, under-estimating the number of clusters;

(d) iK-Means, in most cases, outperforms the others in terms of both centroid and cluster recovery, but it overestimates the number of clusters, especially at the small between-cluster spreads, which can be cured by using a specially designed, “HT-adjusted”, version by increasing the size of anomalous patterns being disregarded before running K-Means.

Since L_1 and L_2 iK-Means methods may lead to very different results as follows from our experiments, we are interested in utilizing the differences between the two methods for a biologically meaningful problem. Such is the problem of finding genes that differently express under different conditions. The gene expression data are in two different types of cell, dendritic (DC) and cancerous dendritic cells (Mutz3). Before applying the clustering method, one needs to normalize the data. This issue has attracted a lot of different proposals – we provide a systematic review of the normalization methods. Since L_1 and L_2 versions tend to produce rather different results, we consider those clusters valid that are present among the results of both methods – we refer to their contents as cluster-consistent genes. The L_1-L_2 cluster consistency can be used for analyzing the difference in gene activity across gene expression data in different cells. We utilized the property of our gene expression data that they contain highly correlated signals to develop a special normalization method for separation of the physical condition of the gene expression experiment from its biological part, the pivot-based normalization (PBR), which is compared with other normalization methods. Our results indicate that:

(a) By using only L_1-L_2 consistent gene sets, two sets of genes have been found: those consistently weak in DC and active in Mutz3, and those consistently active in DC and weak in Mutz3;

(b) PBR normalization method finds most conservative cases of the difference between DC and Mutz3 signals.

Among the issues left unexplored one should mention the following. There are many data structures, not covered in this project, that deserve consideration as a medium for comparing clustering methods. Further research should deeper investigate the entire issue of modelling various data structures and see how methods compare on different data structures. Our attempt in this direction, related to the exponential distribution, indicates that there can emerge different patterns in choosing the right number of clusters. Other future work should include the two approaches to choosing K^* that we reviewed but not covered in our experiments: those resampling based on and those utilizing hierarchical clustering approaches. Another direction should include more search-intensive versions of K-Means, such as for example, involving the genetic and other evolutionary minimization algorithms.

References

Adriaans, P. and Zantinge, D. (1996), *Data Mining*, Addison-Wesley Professional

Aldenderfer, Mark S., and Blashfield Roger K. (1984), *Cluster Analysis*, Sage Publications, Inc.

Anderberg, M. R. (1973), *Cluster Analysis for Applications*, Academic Press, New York

Babu G. P., Murty, M. N., (1993), A near-optimal initial seed value selection in K-Means algorithm using a genetic algorithm, *Pattern Recognition Lett.* Vol. 14(10), pp. 763-769

Baker F. B., and Hubert L J., (1975), Measuring the power of hierarchical cluster analysis, *Journal American Statistical Association*, 70, pp. 31-38

Ball, G. H., and Hall, D. J. (1965), *ISODATA: A novel method for data analysis and pattern classification*, Menlo Park, CA: Stanford Research Insitute

Bandyopadhyay, S. and Maulik,U., (2002) An evolutionary technique based on K-Means algorithm for optimal clustering in R^N, *Inf. Sci. 146*, 221-237

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

Beale E. M. L. (1969), Euclidean cluster analysis, *Proceedings ISA* 43, pp.92-94

Bel Mufti, G, Bertrand, P., and El Moubarki, L. (2005), Determining the number of groups from measures of cluster stability, In: *Proceedings of International Symposium on Applied Stochastic Models and Data Analysi*, 404-412

Berger J A., Hautaniemi S., Jarvinen A K., Edgren H., Mitra S K., and Astola J. (2004), Optimized LOWESS normalization parameter selection for DNA microarray data, *BMC Bioinformatics*, 5 Article 194

Bergmann, S., Ihmels, J., and Barkai, N. (2003), Iterative signature algorithm for the analysis of large-scale gene expression data, *Phys Rev E*, 67: 031902

Berry M. J. A., Linoff G., (1997), *Data Mining Techniques: For Marketing, Sale and Customer Support*, Wiley Computer

Bischof, H., Leonards, A., and Selb A., (1999), MDL principle for robust vector quantization, *Pattern Analysis and Application*, Psychometrikla, 42, 429-431

Bock, H.-H. (2007), Clustering methods: a history of K-Means algorithms, *In: P. Brito, P. d, G. Cucumel, F. de Carvalho (eds.): Selected contributions in data analysis and classification*. Springer Verlag, Heidelberg, 161-172

Bolshakova N. and Azuaje F. (2003), Cluster validation techniques for genome expression data, *Signal Processing*, Volume 83, Issue 4, April 2003, 825-833

Bolshakova N., Azuaje F., and Cunningham P. (2005), An integrated tool for microarray data clustering and cluster validity assessment, *Bioinformatics*, 21(4):451-455

Bolstad, B. M., Irizarry, R. A., Astrand, M. and Speed, T. P. (2003), A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* 19, 185-193

Boutin F. and Hascoët M. (2004), Cluster Validity Indices for Graph Partitioning. *Proceedings of the conference on Information Visualization IV 2004, IEEE*, 376-381

Breckenridge, J. (1989) Replicating cluster analysis: method, consistency and validity, *Multivariate Behavioral Research*, 24, 147-161

Cabena P., Hadjinian P., Stadler R.,Verhees J., and Zanasi A., (1997), *Discovering Data Mining from Concept to Implementation*, Prentice-Hall PTR

Calinski T. and Harabasz J. (1974), A Dendrite method for cluster analysis, *Communications in Statistics*, 3(1), 1-27.

Casillas, A., Gonzales de Lena, M.T. and Martinez, H. (2003) Document clustering into an unknown number of clusters using a Genetic algorithm, *Text, Speech and Dialogue: 6th International Conference*, 43-49

Chae, S.S., DuBien, J.L. and Warde, W.D. (2006) A method of predicting the number of clusters using Rand's statistic, *Computational Statistics and Data Analysis*, 50 (12), 3531-3546

Cheadle, C., Vawter, M. P., Freed, William J., Becker, and Kevin G. (2003), Analysis of microarray data using Z score transformation, *J Mol Diagn*, 5(2): 73-81, 2003

Chiang M. M.T. and Mirkin B. (2008), Gene expression data analysis using iK-Means methods, Unpublished manuscript.

Davis, D. L. and Bouldin, D. W. (1979), A cluster separation measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1, 224-227

Diday E. (1971), La méthode des nuées dynamiques. *Revue Statist. Appl.* 19, No 2, 19-34

Diday, E. et al. (1979), *Optimisation en classification automatique*, Vol. I, II. Institut National der Recherche en Informatique et en Automatique (INRIA), Le Chesnay, France.

Draghici, S. (2001), Piecewise linearization method for the normalization for cDNA and protein microarrays in multi-channel experiments, Technical report, Biodiscovery Inc., Patent application

Draghici, S. (2003), *Data Analysis Tools for DNA Microarrays*, CRC Press

Duda R. O., and Hart P. E. (1973), *Pattern Classification and Scene Analysis*, Wiley, New York

Dudoit, S. and Fridlyand, J. (2002), A prediction-based resampling method for estimating the number of clusters in a dataset, *Genome Biology*, 3(7): research0036.1-0036.21.

Dunn J. C. (1974), Well separated clusters and optimal fuzzy partitions, *Journal of Cybernetics*, 4, 95-104.

Edelstein, Herbert A (1999), *Introduction to Data Mining and Knowledge Discovery*, Third edition, Two Crows Corporation

Efron B. and Tibshirani R. J. (1993) *An Introduction to the Bootstrap*, Chapman and Hall.

Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. (Eds.) (1996) *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press, Menlo Park, Ca.

Feng, Y. and Hamerly, G. (2006) PG-means: learning the number of clusters in data, *Advances in Neural Information Processing Systems*, 19 (NIPS Proceedings), MIT Press, 393-400.

Finkelstein, D. B., Ewing, R., Gollub, J., Sterky, F., Somerville, S. and Cherry, J. M., (2002), Iterative linear regression by sector. In S. M. Lin and K. F. Johnson, editors, *Methods of Microarray Data Analysis*, pp. 57-68, Kluwer Academic

Fraley, C. and Raftery, A.E. (2002) Model-based clustering, discriminant analysis, and density estimation, *Journal of the American Statistical Association*, 97 (458), 611-631

Frawley W. J., Piatetsky-Shapiro G., Matheus C. J. (1992), Knowledge discovery in databases: an overview, *AI Magazine*, v.13 n.3, p.57-70

Frey, T., and Van Groenewoud, H. (1972), A cluster analysis of the D-squared matrix of white spruce stands in Saskatchewan based on the maximum-minimum principle, *Journal of Ecology*, 60, 873-886

Generation of Gaussian mixture distributed data (2006), *NETLAB neural network software*, http://www.ncrg.aston.ac.uk/netlab.

Grupe F. H., Owrang M. M. (1995), Database mining discovering new knowledge and cooperative advantage, *Information System Management*, Vol. 12, No 4, pp. 26-31

Hair, J. F., Anderson, R. E., Tatham, R. L., & Black, W. C. (1995). *Multivariate data analysis* (4th ed.). Upper Saddle River, NJ: Prentice Hall

Halkidi M., Batistakis Y., and Vazirgiannis M., (2001), Clustering algorithms and validity measure, in: *Proceedings of SSDBM Conference*, USA

Hamerly, G., and Elkan, C., (2002), *Learning the K in K-Means* (Tech. Rep. CS20020716), La Jolla, CA: University of California at San Diego

Hand, D.J. and Krzhanowski, W.J. (2005) Optimising K-Means clustering results with standard software packages, *Computational Statistics and Data Analysis*, 49, 969-973.

Hansen, P. and Mladenovic, N. (2001) J-MEANS: a new local search heuristic for minimum sum of squares clustering, *Pattern Recognition*, 34, 405-413

Hardy A. (1996), On the number of clusters, *Computational Statistics and Data Analysis 23*, 83-96

Hartigan J. A. (1975). *Clustering Algorithms*, New York: J. Wiley & Sons.

He, J., Lan M., Tan, C.-L., Sung S.-Y., Low H.-B., (2004), Initialization of cluster refinement algorithms: A review and comparative study, in: *Proc. of International Joint Conf. on Neural Networks (IJCNN)*, Hungary, Vol. 1, pp. 297-302

Huang C. and Harris R. (1993), A comparison of several codebook generation approaches, *IEEE Trans. Image Processing*, 2(1), pp. 108-112

Hubert L.J. and Arabie P. (1985), Comparing partitions, *Journal of Classification*, 2, 193-218

Hubert L. J. and Levin J. R. (1976), A general statistical framework for assessing categorical clustering in free recall, *Psychological Bulletin* 83, 1072-1080

Kothari, R. and Pitts, D., (1999), On finding the number of clusters, *Pattern Recognition Letters*, 20, 405-416

Kuncheva, L.I. and Vetrov, D. P. (2005) Evaluation of stability of K-Means cluster ensembles with respect to random initialization, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 28, n. 11, 1798-1808

Ishioka, T. (2005) An expansion of X-means for automatically determining the optimal number of clusters, *Proceedings of International Conference on Computational Intelligence* (Calgary, Canada), 91-96.

Jain, A. K. and Dubes, R.C. (1988). *Algorithms for Clustering Data*, Prentice Hall.

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

Jiang, D., Tang, C., and Zhang, A.. (2004), Cluster analysis for gene expression data: A survey. In *IEEE Transactions on Knowledge and Data Engineering (TKDE).*, Vol. 16, pp. 1370-1386

Jiang, H., Deng, Y., Chen, H-S., Tao, L., Sha, Q., Chen, Jun, Tsai, Chung-Jui and Zhang, Shuanglin (2004), Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes, *BMC Bioinformatics*, 5:81

Jiwei H, Micheline K (2001), *Data Mining: Concepts and Techniques*, Morgan Kaufmann, New York

Kaufman L. and Rousseeuw P. (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*, New York: J. Wiley & Son.

Kerr, M. K. and Churchill, G. A. (2001), Bootstrapping cluster analysis: assessing the reliability of conclusions from microarray experiments, *Proc Natl Acad Sci USA*, 98(16), 8961-8965

Khan, S. S. and, Ahmad A., (2004), Cluster center initialization algorithm for K-Means clustering, *Pattern Recognition Lett.* 25(11), pp. 1293-1302

Kleissner C. (1998), Data mining for the enterprise, *IEEE Proc. 31st Annual Hawaii International Conference on System Sciences*, Vol. 7, pp. 295-304

Krzanowski W. and Lai Y. (1985), A criterion for determining the number of groups in a dataset using sum of squares clustering, *Biometrics*, 44, 23-34.

Leisch, F. (2006) A toolbox for K-centroids cluster analysis, *Computational Statistics and Data Analysis*, 51, 526-544

Lepre, Jprge, Rice, John Jeremy, Tu, Yuhai, and Stolovitzky, Gustavo (2004), Genes@work: an efficient algorithm for pattern discovery and multivariate feature selection in gene expression data, *Bioinformatics*, 20(7):1033-1044

Levine E. and Domany E. (2001), Resampling method for unsupervised estimation of cluster validity, *Neural Computation*, 13, 2573-2593

Li, C., Wong W. H. (2001), Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection, *Proc. Natl. Acad. Sci.*, 98(1):31-36

Li, C., Wong W. H. (2001a), Model-based analysis of oligonucleotide arrays: Model validation, design issues and standard error application, *Genome Biology*, 2(8):1-11

Liggett, W. (2006), Normalization and technical variation in gene expression measurements, *J. Res. Natl. Inst. Stand. Technol.*, 111, 361-372

Likas A., Vlassis N., Verbeek J. J. (2003), The Global K-Means Clustering Algorithm, *Pattern Recognition 36(2)*: 451-461

Liu H. and MotodaH. (1998), *Feature selection for knowledge discovery and data mining*, Kluwer Academic Publisher

Lloyd S. P. (1982), Least squares quantization in pcm, *IEEE Trans. Inf. Theory 28 (2)*, 129–137

Ma J. and Qin Z. (2007), Different normalization strategies for microarray gene expression traits affect the heritability estimation, *BMC Proceedings*, I (Suppl I):S154

Margush T. and McMorris F. R., (1981), Consensus n-trees, *Bulletin of Mathematical Biology*, 43, 239-244

Maulik,U. and, Bandyopadhyay, S. (2000) Genetic algorithm-based clustering technique, *Pattern Recognition*, 33, 1455-1465

Maulik,U. and, Bandyopadhyay, S. (2002a) Performance evaluation of some clustering algorithms and validity indices, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 24, No 12, pp. 1650-1654

McClain J. O. and Rao V. R. (1975), CLUSTISZ: A program to test for the quality of cluster of a set of objects, *Journal of Marketing Research*, 12, 456-460

McLachlan G.J. and Basford K. (1988), *Mixture Models: Inference and Applications to Clustering*, New York: Marcel Dekker.

McLachlan, G.J. and Khan, N. (2004), On a resampling approach for tests on the number of clusters with mixture model-based clustering of tissue samples, *Journal of Multivariate Analysis*, 90, 90-1005

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

McQueen J. (1967) Some methods for classification and analysis of multivariate observations. *In Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. II*, pages 281–297.

Milligan G. W. (1980), The validation of four ultrametric clustering algorithm, *Pattern Recognition*, Vol. 12, pp. 41-50

Milligan G. W. and Cooper M. C. (1985), An examination of procedures for determining the number of clusters in a data set, *Psychometrika*, 50, 159-179.

Milligan G. W. and Cooper M. C. (1987), Methodology review: clustering methods, *Applied Psychological Measurement*, Vol. 11, pp. 329-354

Milligan, G. W. and Cooper, M. C. (1988) A study of standardization of variables in cluster analysis, *Journal of Classification*, 5, 181-204.

Minaei-Bidgoli, B., Topchy, A. and Punch, W.F. (2004) A comparison of resampling methods for clustering ensembles, *International conference on Machine Learning; Models, Technologies and Application (MLMTA04)*, Las

Vegas, Nevada, pp. 939-945

Mirkin B. (2001) Eleven ways to look at the Pearson chi squares coefficient at contingency tables, The American Statistician, 55, no. 2, 111-120.

Mirkin B. (2005) *Clustering for Data Mining: A Data Recovery Approach*, Boca Raton Fl., Chapman and Hall/CRC.

Mitra P., Murthy C. A. Pal S. K., (2002), Density based Multiscale data condensation, *IEEE Trans. Pattern Anal. Machine Intell.* 24 (6), pp. 734-747

Mojena R. (1977), Hierarchical grouping methods and stopping rules: an evaluation, *Computer Journal 20*, pp. 359-363

Möller U. and Radke D. (2006), Performance of data resampling methods for robust class discovery based on clustering, *Intelligent Data Analysis*, Vol. 10, Number 2, 139-162

Monti S., Tamayo P., Mesirov J. and, Golub T. (2003). Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data, *Machine Learning*, 52, 91-118.

Murtagh, F. and Raftery, A.E. (1984) Fitting straight lines to point patterns, *Pattern Recognition*, 17, 479-483

Nimgaonkar A., Sanoudou D., Butte AJ, Haslett JN, Kunkel LM, Beggs AH, and Kohane IS (2003), Reproducibility of gene expression across generations of Affymetrix microarrays, *BMC Bioinformatics*, 4:27

Pandey, G., Ramakrishnan, L., Naarayanan, S., Michael, and Kumar, V. (2007), Systematic evaluation of normalization methods for gene expression data, Technical Report 07-015, Department of Computer Science and University of Minnesota

Paterlini S. and Krink T. (2006), Differential evolution and particle swarm optimization in partitional clustering, *Computational Statistics & Data Analysis*, 50, pp. 1220-1247

Park, T., Yi, S., Kang, S., Lee, S., Lee, Y., and Simon R., (2003), Evaluation of normalization methods for microarray data, *BMC Bioinformatics*, 4-33

Pedersen T. and Kulkarni A. (2006), Automatic Cluster Stopping with Criterion Functions and the Gap Statistic, *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology***,** 276 - 279

Pelleg, D., & Moore, A. (2000), X-means: Extending K-Means with efficient estimation of the number of clusters, *In Proceedings of the Seventeenth International Conference on Machine Learning (pp. 727-734)*, San Francisco:

Morgan Kaufmann

Pena, J. M., Lozano, J. A., Larranaga P., (1999), An empirical comparison of four initialization methods for K-Means algorithm, *Pattern Recognition Lett. 20(10)*, pp. 1027-1040

Petrović S. (2006), A Comparison Between the Silhouette Index and the Davies-Bouldin Index in Labelling IDS Clusters, *Proceedings of the 11th Nordic Workshop on Secure IT-systems, NORDSEC 2006*, pp. 53-64, Linkoping, Sweden

Pollard D. (1982), Quantization and the method of K-means, *IEEE Trans. Inf. Theory 28 (2)*, 199–205

Pollard, K.S. and van der Laan, M.J. (2002) A method to identify significant clusters in gene expression data, *U.C. Berkeley Division of Biostatistics Working Paper* Series, 107

Quackenbush, J., (2002), Microarray data normalization and transformation, *Nature Genetics*, 32:496-501

Ray S., Turi R. H., (1999), Determination of number of clusters in K-Means clustering and application in colour image segmentation, in: *Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques (ICAPRDT'99)*, Calcutta, India

Redmond S. J. and Heneghan C. (2007), A method for initializing the K-Means clustering algorithm using kd-tree, *Pattern Recognition Lett.* 28, pp. 965-973

Rocke D.M. and Dai J. (2003), Sampling and subsampling for cluster analysis in data mining: With applications to sky survey data, *Data mining and Knowledge Discovery*, 7, 215-232

Saviozzi, S. and Calogero, R. A., (2003), Microarray probe expression measures, data normalization and statistical validation, *Comp Funct Genom*, 4: 442-446

Shen J. Chang S. I., Lee E. S., Deng Y., Brown S. J. (2005), Determination of cluster number in clustering microarray data, *Applied Mathematics and Computation*, Vol 169, Issue 2, pp. 1172-1185

Shmulevich I. and Zhang W. (2002), Binary analysis and optimization-based normalization of gene expression data, *Bioinformatics*, Vol 18 no. 4, pp. 555-565

Späth, H. (1985), Cluster Dissection and Analysis, Ellis Horwood, Chichester

Steinbach, M., Karypis, G. and Kumar, V. (2000), A comparison of document clustering techniques, KDD-2000 Workshop on Text Mining

Steinhoff C. and Vingron M. (2006), Normalization and quantification of differential expression in gene expression microarrays, *Briefings in Bioinformatics*, Vol 7, No 2, pp. 166-177

Steinley D. (2003), Local optima in K-Means clustering: What you don’t know may hurt you, *Psychological Methods*, Vol. 8, pp. 294-304

Steinley, D. (2004) Standardizing variables in K-Means clustering. In D. Banks, L. House, F.R. McMorris, P. Arabie and W. Gaul (Eds.) *Classification, Clustering, and Data Mining Applications*, New York, Springer, 53-60

Steinley D. (2006), K-Means clustering: A half-century synthesis, *British Journal of Mathematical and statistical Psychology*, 59, 1-34

Steinley, D. and, Brusco M., (2007), Initializing K-Means batch clustering: A critical evaluation of several techniques, *Journal of Classification*, Vol. 24, pp. 99-121

Steinley, D. and Henson, R. (2005) OCLUS: An analytic method for generating clusters with known overlap, *Journal of Classification*, 22, 221-250

Sugar C.A. and James G.M. (2003), Finding the number of clusters in a data set: An information-theoretic approach, *Journal of American Statistical Association*, 98, n. 463, 750-778.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E., S., Golub, and Todd R. (1999), Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation, *PNAS*, 96(6): 2907-2912

Thiesson, B., Meck B., Chickering, C. and Heckerman D (1997), Learning mixtures of Bayesian networks, *Microsoft Technical Report TR-97-30*, WA

Tian J., Zhou I., Zhang S., Liu L (2005), Improvement and parallelism of K-Means clustering algorithm, Tsinghua Science and Technology, Volum 10, Number 3, pp277-281

Tibshirani R., Walther G. and Hastie T. (2001), Estimating the number of clusters in a dataset via the Gap statistics, *Journal of the Royal Statistical Society B*, 63, 411-423.

Tipping M.E. and Bishop C.M., (1999). Probabilistic principal component analysis, *Journal of the Royal Statistical Society*. B 61, 611–622.

Tou J., Gouzales R., (1974), *Pattern Recognition Principles*, Addison-Wesley, Reading, MA

Tseng G.C. and Wong W.H. (2003), Tight clustering: a resampling-based approach for identifying stable and tight patterns in data, Biometrics, 61, 10-16

Vapnik, V. (2006) *Estimation of Dependences Based on Empirical Data*, Springer

Science + Business Media Inc., 2nd edition.

Venet D. (2003), MatArray: a Matlab toolbox for microarray data. *Bioinformatics*; 19(5):659-660.

Vesentao J. (2001), Importance of Individual Variables in the k−Means Algorithm, *In Proceedings of the Pacific−Asia Conference Advances in Knowledge Discovery and Data Mining (PAKDD2001),* Springer−Verlag, pp. 513−518, 2001.

Wang, W. Lu, J., Lee, R., Gu, Z., and Clarke R., (2002), Iterative normalization of cDNA microarray data, *IEEE Transactions on Information Technology in Biomedicine*, 6(1):29-37

Ward J. H.., (1963). Hierarchical Grouping to optimize an objective function. *Journal of American Statistical Association, 58*(301), pp. 236-244.

Wasito I., Mirkin B. (2006) Nearest neighbours in least-squares data imputation algorithms with different missing patterns, *Computational Statistics & Data Analysis*, 50, 926-949.

Wishart D. (1969), *FORTRAN II programs for 8 methods of cluster analysis (CLUSTAN I)*, Computing Contributions, 38th State Geological Survey, Lawarence, KS: University of Kansas

Wishart, D. (2005), Number of Clusters, in: *Encylopedia of Statistics in Behavioral Science*, Everitt, B S, and Howell, D C (eds), Wiley, Chichester, UK, **3**, 1442-1446

Workman, C., Jensen, L., Jarmer, H., Berka, R., Gautier, L. N., Henrik, S., Hans-Henrik, N. C., Brunak, S. and Knudsen, S. (2002). A new non-linear normalization method for reducing variability in dna microarray experiments. Genome Biology, 3(9): research0048. 1–research0048.16

Yang, Y. H., Dudoit, S., Luu, P., and Speed, T. P. (2001), Normalization for cDNA microarray data, *In Microarrays: optical technologies and informatics volume 4266*, Bittner, M., Chen, Y., Dorsel, A., and Dougherty, E. R. (eds.), San Jose, CA, USA: SPIE, 141−152.

Yang, Y. H., Buckley, M. J., Dudoit, S., and Speed, T. P. (2002), Comparison of methods for image analysis on cDNA microarray data, *Journal of Computational and Graphic Statistics*, 11:108-136

Yeung K. Y. and Ruzzo W. L. (2001), Details of the Adjusted Rand index and clustering algorithms, *Bioinformatics*, Vol. 17, 763--774.

Zein A., Aigner T., Zimmer R., and Lengauer T. (2001), Centralization: A new method for the normalization of gene expression data, *Bioinformatics*, Vol. 1 no1, 1-9

Appendix A Lists of genes in Tumour/Dendrite gene expression data using Pivot-based with the removal normalization method

A.1 Genes that are weak DC but very active in Mutz3

gene number	probename	gene name	gene description
3722	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
4082	A_23_P96158	KRT17	Homo sapiens keratin 17 (KRT17), mRNA [NM_000422]
4198	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
11366	A_32_P62963	ENST00000332402	Unknown
15339	A_24_P8371	LOC124976	Homo sapiens, Similar to spinster-like protein, clone IMAGE:4814561, mRNA, partial cds. [BC041772]
15533	A_24_P882732	ENST00000311208	Unknown
16044	A_23_P121596	PPBP	Homo sapiens pro-platelet basic protein (chemokine (C-X-C motif) ligand 7) (PPBP), mRNA [NM_002704]
19594	A_24_P94222	FBLP-1	Homo sapiens filamin-binding LIM protein-1 (FBLP-1), mRNA [NM_017556]
20272	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
21187	A_24_P153035	ENST00000311208	Unknown
22327	A_24_P887857	LOC440421	PREDICTED: Homo sapiens similar to keratin 17 (LOC440421), mRNA [XM_496202]
22943	A_23_P49136	LOC161931	Homo sapiens testis nuclear RNA-binding protein-like (LOC161931), mRNA [NM_139174]
25019	A_23_P154849	OLIG1	Homo sapiens oligodendrocyte transcription factor 1 (OLIG1), mRNA [NM_138983]
25133	A_32_P76627	ENST00000322533	full-length cDNA clone CS0DI013YN06 of Placenta Cot 25-normalized of Homo sapiens (human). [CR597597]
27258	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
30434	A_24_P265346	KRT14	Homo sapiens keratin 14 (epidermolysis bullosa simplex, Dowling-Meara, Koebner) (KRT14), mRNA [NM_000526]
31036	A_23_P38537	KRT16	Homo sapiens keratin 16 (focal non-epidermolytic palmoplantar keratoderma) (KRT16), mRNA [NM_005557]
31165	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
32578	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
35681	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
35731	A_23_P56050	TNNT1	Homo sapiens troponin T1, skeletal, slow, mRNA (cDNA clone IMAGE:3531880), partial cds. [BC022086]
37900	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
39452	A_32_P53524	THC2132626	Q6IGP7 (Q6IGP7) HDC05721, partial (12%) [THC2132626]
39676	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
42205	A_24_P392991	KRT16	Homo sapiens keratin 16 (focal non-epidermolytic palmoplantar keratoderma) (KRT16), mRNA [NM_005557]
43512	A_24_P261734	SLC38A1	Homo sapiens cDNA FLJ14201 fis, clone NT2RP3002955. [AK024263]
1318	A_24_P610945	ENST00000311197	Unknown
4029	A_23_P152047	SCAMP5	Homo sapiens secretory carrier membrane protein 5 (SCAMP5), mRNA [NM_138967]
6286	A_23_P206280	GPR56	Homo sapiens G protein-coupled receptor 56 (GPR56), transcript variant 1, mRNA [NM_005682]
13186	A_23_P204751	ACCN2	Homo sapiens amiloride-sensitive cation channel 2, neuronal (ACCN2), transcript variant 1, mRNA [NM_020039]
17095	A_23_P336198	GLCCI1	Homo sapiens cDNA FLJ36336 fis, clone THYMU2006303. [AK093655]
17501	A_23_P39647	SLC4A3	Homo sapiens solute carrier family 4, anion exchanger, member 3 (SLC4A3), mRNA [NM_005070]
24861	A_24_P923676	X15674	Human pTR5 mRNA for repetitive sequence. [X15674]
25435	A_23_P396765	PGM2LM	Homo sapiens phosphoglucomutase 2-like 1 (PGM2LM), mRNA [NM_173582]
34293	A_23_P122863	GRB10	Homo sapiens growth factor receptor-bound protein 10 (GRB10), transcript variant 4, mRNA [NM_001001555]

35417	A_23_P390504	FOXC1	Homo sapiens forkhead box C1 (FOXC1), mRNA [NM_001453]

A.2 Genes that are very active in DC but weak in Mutz3

gene number	probename	gene name	gene description
2344	A_24_P12573	CCLS6	Homo sapiens chemokine (C-C motif) ligand 26 (CCLS6), mRNA [NM_006072]
8538	A_23_P109143	PRNP	Homo sapiens prion protein (p27-30) (Creutzfeld-Jakob disease, Gerstmann-Strausler-Scheinker syndrome, fatal familial insomnia) (PRNP), transcript variant 1, mRNA [NM_000311]
10465	A_23_P209625	CYP1B1	Homo sapiens cytochrome P450, family 1, subfamily B, polypeptide 1 (CYP1B1), mRNA [NM_000104]
15420	A_23_P134347	CPVL	Homo sapiens carboxypeptidase, vitellogenic-like (CPVL), transcript variant 1, mRNA [NM_031311]
17957	A_23_P36745	ALDH2	Homo sapiens aldehyde dehydrogenase 2 family (mitochondrial) (ALDH2), nuclear gene encoding mitochondrial protein, mRNA [NM_000690]
28352	A_23_P7827	AF086130	Homo sapiens full length insert cDNA clone ZA84A12. [AF086130]
29926	A_23_P16915	QPCT	Homo sapiens glutaminyl-peptide cyclotransferase (glutaminyl cyclase) (QPCT), mRNA [NM_012413]
40888	A_24_P319088	CCLS3	Homo sapiens chemokine (C-C motif) ligand 23 (CCLS3), transcript variant CKbeta8-1, mRNA [NM_005064]
2927	A_24_P380734	SDC2	Homo sapiens syndecan 2 (heparan sulfate proteoglycan 1, cell surface-associated, fibroglycan) (SDC2), mRNA [NM_002998]
13355	A_24_P673063	FABP5	Homo sapiens fatty acid binding protein 5 (psoriasis-associated) (FABP5), mRNA [NM_001444]
19729	A_24_P71468	QPCT	Homo sapiens glutaminyl-peptide cyclotransferase (glutaminyl cyclase) (QPCT), mRNA [NM_012413]
27927	A_23_P116898	A2M	Homo sapiens alpha-2-macroglobulin (A2M), mRNA [NM_000014]
40263	A_23_P39265	C4.4A	Homo sapiens GPI-anchored metastasis-associated protein homolog (C4.4A), mRNA [NM_014400]
42282	A_23_P94533	CTSL	Homo sapiens cathepsin L (CTSL), transcript variant 1, mRNA [NM_001912]

A.3 Genes that are active in DC but weak in Mutz3

gene number	probename	gene name	gene description
9104	A_23_P31755	CRH	Homo sapiens corticotropin releasing hormone (CRH), mRNA [NM_000756]
19209	A_23_P87709	FLJ22662	Homo sapiens hypothetical protein FLJ22662 (FLJ22662), mRNA [NM_024829]
21084	A_23_P147025	RAB33A	Homo sapiens RAB33A, member RAS oncogene family (RAB33A), mRNA [NM_004794]
28028	A_23_P63209	HSD11B1	Homo sapiens hydroxysteroid (11-beta) dehydrogenase 1 (HSD11B1), transcript variant 2, mRNA [NM_181755]
31460	A_23_P89799	ACAA2	Homo sapiens acetyl-Coenzyme A acyltransferase 2 (mitochondrial 3-oxoacyl-Coenzyme A thiolase) (ACAA2), nuclear gene encoding mitochondrial protein, mRNA [NM_006111]

Appendix B Lists of genes in Tumour/Dendrite gene expression data at two normalization methods

A. Pivot-based without removals normalization method

A.1 Genes that are weak DC but active in Mutz3

gene number	probename	gene name	gene description
262	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
3722	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
4082	A_23_P96158	KRT17	Homo sapiens keratin 17 (KRT17), mRNA [NM_000422]
4198	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
11366	A_32_P62963	ENST00000332402	Unknown
15339	A_24_P8371	LOC124976	Homo sapiens, Similar to spinster-like protein, clone IMAGE:4814561, mRNA, partial cds. [BC041772]
15533	A_24_P882732	ENST00000311208	Unknown
16044	A_23_P121596	PPBP	Homo sapiens pro-platelet basic protein (chemokine (C-X-C motif) ligand 7) (PPBP), mRNA [NM_002704]
16620	A_23_P314101	SUSD2	Homo sapiens sushi domain containing 2 (SUSD2), mRNA [NM_019601]
19594	A_24_P94222	FBLP-1	Homo sapiens filamin-binding LIM protein-1 (FBLP-1), mRNA [NM_017556]
20272	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
21827	A_23_P360754	ADAMTS4	Homo sapiens a disintegrin-like and metalloprotease (reprolysin type) with thrombospondin type 1 motif, 4 (ADAMTS4), mRNA [NM_005099]
22327	A_24_P887857	LOC440421	PREDICTED: Homo sapiens similar to keratin 17 (LOC440421), mRNA [XM_496202]
22943	A_23_P49136	LOC161931	Homo sapiens testis nuclear RNA-binding protein-like (LOC161931), mRNA [NM_139174]
25019	A_23_P154849	OLIG1	Homo sapiens oligodendrocyte transcription factor 1 (OLIG1), mRNA [NM_138983]
25133	A_32_P76627	ENST00000322533	full-length cDNA clone CS0DI013YN06 of Placenta Cot 25-normalized of Homo sapiens (human). [CR597597]
27258	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
30434	A_24_P265346	KRT14	Homo sapiens keratin 14 (epidermolysis bullosa simplex, Dowling-Meara, Koebner) (KRT14), mRNA [NM_000526]
31036	A_23_P38537	KRT16	Homo sapiens keratin 16 (focal non-epidermolytic palmoplantar keratoderma) (KRT16), mRNA [NM_005557]
31165	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
32578	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
35681	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
35731	A_23_P56050	TNNT1	Homo sapiens troponin T1, skeletal, slow, mRNA (cDNA clone IMAGE:3531880), partial cds. [BC022086]
37900	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
39452	A_32_P53524	THC2132626	Q6IGP7 (Q6IGP7) HDC05721, partial (12%) [THC2132626]
39676	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
43512	A_24_P261734	SLC38A1	Homo sapiens cDNA FLJ14201 fis, clone NT2RP3002955. [AK024263]
44198	A_23_P79769	BIRC7	Homo sapiens baculoviral IAP repeat-containing 7 (livin) (BIRC7), transcript variant 2, mRNA [NM_022161]

A.2 Genes that are active in DC but weak in Mutz3

gene number	probename	gene name	gene description
10465	A_23_P209625	CYP1B1	Homo sapiens cytochrome P450, family 1, subfamily B, polypeptide 1 (CYP1B1), mRNA [NM_000104]
15420	A_23_P134347	CPVL	Homo sapiens carboxypeptidase, vitellogenic-like (CPVL), transcript variant 1, mRNA [NM_031311]
17957	A_23_P36745	ALDH2	Homo sapiens aldehyde dehydrogenase 2 family (mitochondrial) (ALDH2), nuclear gene encoding mitochondrial protein, mRNA [NM_000690]
33774	A_24_P133905	CCLS3	Homo sapiens chemokine (C-C motif) ligand 23 (CCLS3), transcript variant CKbeta8-1, mRNA [NM_005064]

40888	A_24_P319088	CCLS3	Homo sapiens chemokine (C-C motif) ligand 23 (CCLS3), transcript variant CKbeta8-1, mRNA [NM_005064]

B. Intensity dependent normalization

B.1 Genes that are weak DC but active in Mutz3

gene number	probename	gene name	gene description
262	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
3722	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
4082	A_23_P96158	KRT17	Homo sapiens keratin 17 (KRT17), mRNA [NM_000422]
4198	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
11366	A_32_P62963	ENST00000332402	Unknown
15339	A_24_P8371	LOC124976	Homo sapiens, Similar to spinster-like protein, clone IMAGE:4814561, mRNA, partial cds. [BC041772]
15533	A_24_P882732	ENST00000311208	Unknown
16044	A_23_P121596	PPBP	Homo sapiens pro-platelet basic protein (chemokine (C-X-C motif) ligand 7) (PPBP), mRNA [NM_002704]
16620	A_23_P314101	SUSD2	Homo sapiens sushi domain containing 2 (SUSD2), mRNA [NM_019601]
17501	A_23_P39647	SLC4A3	Homo sapiens solute carrier family 4, anion exchanger, member 3 (SLC4A3), mRNA [NM_005070]
19594	A_24_P94222	FBLP-1	Homo sapiens filamin-binding LIM protein-1 (FBLP-1), mRNA [NM_017556]
20272	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
21187	A_24_P153035	ENST00000311208	Unknown
21827	A_23_P360754	ADAMTS4	Homo sapiens a disintegrin-like and metalloprotease (reprolysin type) with thrombospondin type 1 motif, 4 (ADAMTS4), mRNA [NM_005099]
22327	A_24_P887857	LOC440421	PREDICTED: Homo sapiens similar to keratin 17 (LOC440421), mRNA [XM_496202]
22943	A_23_P49136	LOC161931	Homo sapiens testis nuclear RNA-binding protein-like (LOC161931), mRNA [NM_139174]
25019	A_23_P154849	OLIG1	Homo sapiens oligodendrocyte transcription factor 1 (OLIG1), mRNA [NM_138983]
25133	A_32_P76627	ENST00000322533	full-length cDNA clone CS0DI013YN06 of Placenta Cot 25-normalized of Homo sapiens (human). [CR597597]
25435	A_23_P396765	PGM2LM	Homo sapiens phosphoglucomutase 2-like 1 (PGM2LM), mRNA [NM_173582]
27258	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
30434	A_24_P265346	KRT14	Homo sapiens keratin 14 (epidermolysis bullosa simplex, Dowling-Meara, Koebner) (KRT14), mRNA [NM_000526]
31036	A_23_P38537	KRT16	Homo sapiens keratin 16 (focal non-epidermolytic palmoplantar keratoderma) (KRT16), mRNA [NM_005557]
31165	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
32578	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
34293	A_23_P122863	GRB10	Homo sapiens growth factor receptor-bound protein 10 (GRB10), transcript variant 4, mRNA [NM_001001555]
35681	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
35731	A_23_P56050	TNNT1	Homo sapiens troponin T1, skeletal, slow, mRNA (cDNA clone IMAGE:3531880), partial cds. [BC022086]
37900	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
39452	A_32_P53524	THC2132626	Q6IGP7 (Q6IGP7) HDC05721, partial (12%) [THC2132626]
39676	A_23_P87879	CD69	Homo sapiens CD69 antigen (p60, early T-cell activation antigen) (CD69), mRNA [NM_001781]
42205	A_24_P392991	KRT16	Homo sapiens keratin 16 (focal non-epidermolytic palmoplantar keratoderma) (KRT16), mRNA [NM_005557]
43512	A_24_P261734	SLC38A1	Homo sapiens cDNA FLJ14201 fis, clone NT2RP3002955. [AK024263]
44198	A_23_P79769	BIRC7	Homo sapiens baculoviral IAP repeat-containing 7 (livin) (BIRC7), transcript variant 2, mRNA [NM_022161]

B.2 Genes that are active in DC but weak in Mutz3

gene number	probename	gene name	gene description
2344	A_24_P12573	CCLS6	Homo sapiens chemokine (C-C motif) ligand 26 (CCLS6), mRNA [NM_006072]
8538	A_23_P109143	PRNP	Homo sapiens prion protein (p27-30) (Creutzfeld-Jakob disease, Gerstmann-Strausler-Scheinker syndrome, fatal familial insomnia) (PRNP), transcript variant 1, mRNA [NM_000311]
10465	A_23_P209625	CYP1B1	Homo sapiens cytochrome P450, family 1, subfamily B, polypeptide 1 (CYP1B1), mRNA [NM_000104]
15420	A_23_P134347	CPVL	Homo sapiens carboxypeptidase, vitellogenic-like (CPVL), transcript variant 1, mRNA [NM_031311]
16552	A_23_P214222	MARCKS	Homo sapiens myristoylated alanine-rich protein kinase C substrate (MARCKS), mRNA [NM_002356]
17957	A_23_P36745	ALDH2	Homo sapiens aldehyde dehydrogenase 2 family (mitochondrial) (ALDH2), nuclear gene encoding mitochondrial protein, mRNA [NM_000690]
23361	A_23_P215484	CCLS6	Homo sapiens chemokine (C-C motif) ligand 26 (CCLS6), mRNA [NM_006072]
28352	A_23_P7827	AF086130	Homo sapiens full length insert cDNA clone ZA84A12. [AF086130]
29926	A_23_P16915	QPCT	Homo sapiens glutaminyl-peptide cyclotransferase (glutaminyl cyclase) (QPCT), mRNA [NM_012413]
33774	A_24_P133905	CCLS3	Homo sapiens chemokine (C-C motif) ligand 23 (CCLS3), transcript variant CKbeta8-1, mRNA [NM_005064]
40888	A_24_P319088	CCLS3	Homo sapiens chemokine (C-C motif) ligand 23 (CCLS3), transcript variant CKbeta8-1, mRNA [NM_005064]
41755	A_23_P29773	LAMP3	Homo sapiens lysosomal-associated membrane protein 3 (LAMP3), mRNA [NM_014398]