APPROXIMATION TO K-MEANS-TYPE CLUSTERING

Approximation to K-means-type Clustering

By

YU WEI, B.Sc.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

McMaster University

MASTER OF SCIENCE(2005) McMaster University

COMPUTING AND SOFTWARE Hamilton, Ontario

TITLE: Approximation Methods to Clustering Analysis

AUTHOR: Yu Wei, B.Sc. (Beijing University)

SUPERVISOR: Dr. Jiming Peng

NUMBER OF PAGERS: xvii, 110

Abstract

Clustering involves partitioning a given data set into several groups based on some similarity/dissimilarity measurements. Cluster analysis has been widely used in information retrieval, text and web mining, pattern recognition, image segmentation and software reverse engineering.

K-means is the most intuitive and popular clustering algorithm and the working horse for clustering. However, the classical K-means suffers from several flaws. First, the algorithm is very sensitive to the initialization method and can be easily trapped at a local minimum regarding to the measurement (the sum of squared errors) used in the model. On the other hand, it has been proved that finding a global minimal sum of the squared errors is NP-hard even when $k = 2$. In the present model for K-means clustering, all the variables are required to be discrete and the objective is nonlinear and nonconvex.

In the first part of the thesis, we consider the issue of how to derive an optimization model to the minimum sum of squared errors for a given data set based on continuous convex optimization. For this, we first transfer the K-means clustering into a novel optimization model, 0-1 semidefinite programming where the eigenvalues of involved matrix argument must be 0 or 1. This provides an unified way for many other clustering approaches such as spectral clustering and normalized cut. Moreover, the new optimization model also allows us to attack the original problem based on the relaxed linear and semidefinite programming.

Moreover, we consider the issue of how to get a feasible solution of the original clustering from an approximate solution of the relaxed problem. By using principal component analysis, we construct a rounding procedure to extract a feasible clustering and show that our algorithm can provide a 2-approximation to the global solution of the original problem. The complexity of our rounding procedure is $O(n^{k^2(k-1)/2})$, which improves substantially a similar rounding procedure in the literature with a complexity $O(n^{k^3/2})$. In particular, when $k = 2$, our rounding procedure runs in $O(n \log n)$ time. To the best of our knowledge, this is the lowest complexity that has been reported in the literature to find a solution to K-means clustering with guaranteed quality.

In the second part of the thesis, we consider approximation methods for the so-called balanced bi-clustering. By using a simple heuristics, we prove that we can improve slightly the constrained K-means for bi-clustering. For the special case where the size of each cluster is fixed, we develop a new algorithm, called Q-means, to find a 2-approximation solution to the balanced bi-clustering. We prove that the Q-means has a complexity $O(n^2)$.

Numerical results based our approaches will be reported in the thesis as well.

Acknowledgements

I would like to gratefully appreciate my supervisor Dr. Jiming Peng for his valuable comments and enthusiastic support. Without his support, this work would not have been done. I also thank him for his very careful reading and insightful suggestions during the writing of this thesis, and his thoughtful guidance during my graduate study.

I am indebted to Dr. Tim Davidson for the optimization knowledge he gave to me. I appreciate Professor Terlaky, and Professor Peng, for the inspirational optimization seminars they organized and for the great facility the laboratory they provided.

I heartfully acknowledge Dr. Jiming Peng, Dr. Antoine Deza and Dr. Kamran Sartipi for their agreement to be a committee member and for their careful reading of, and helpful suggestions on, my thesis.

I thank all members of the Advanced Optimization Lab (AdvOL) for their friendly help during my master studies and for the pleasant and exciting working environment they built.

At last, my special thanks go to also my parents for everything they gave me, their love, understanding, encouragement, and support.

Contents

List of Notations

$\mathcal{S}$:	given set of entities
$\|x\|_p$:	p norm of vector x
x^T, X^T :	transpose of the vector x, matrix X
X^{-1} :	inverse of the matrix X
$\mathbf{R}^d$:	n-dimension Euclidian vector space
$\Re^{n\times k}$:	$n \times k$ real matrix
Tr(X) :	trace of matrix X
e^k :	all 1's vector in k dimension
I :	identity matrix
diag(X, Y, ...) :	block diagonal matrix with diagonal blocks $X, Y, ...$
$\mathcal{I}; \mathcal{J}$:	index set

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

In this chapter, we first give a overview of the clustering analysis and the prevailing clustering methods, and then we introduce the so-called constrained clustering, which is one of the focuses of this thesis.

1.1 What is Clustering Analysis?

Clustering analysis is an important technique in the rapidly growing field known as exploratory data analysis and is being applied in a variety of engineering and science disciplines. Typically classified as a subfield of data mining, as data mining itself, clustering analysis has its own multidisciplinary nature, it has been widely studied by experts in a number of research communities, including machine learning, statistics, social science, optimization and computational geometry [21].

More formally, the clustering problem describes the problem where the goal is to partition a given set of n entities (also known as patterns or points etc.) $\mathcal{S} = \{s_1, \cdots .s_n\}$ into several groups based on how similar/dissimilar they are, such that entities within the same group are similar to each other and entities that belong to two different groups are dissimilar to some extent. An example of clustering is depicted in Figure 1.1. The input entities are shown in Figure 1.1(a), and the desired clusters are shown in Figure 1.1(b). Here, entities belong to the same cluster are given the same label.

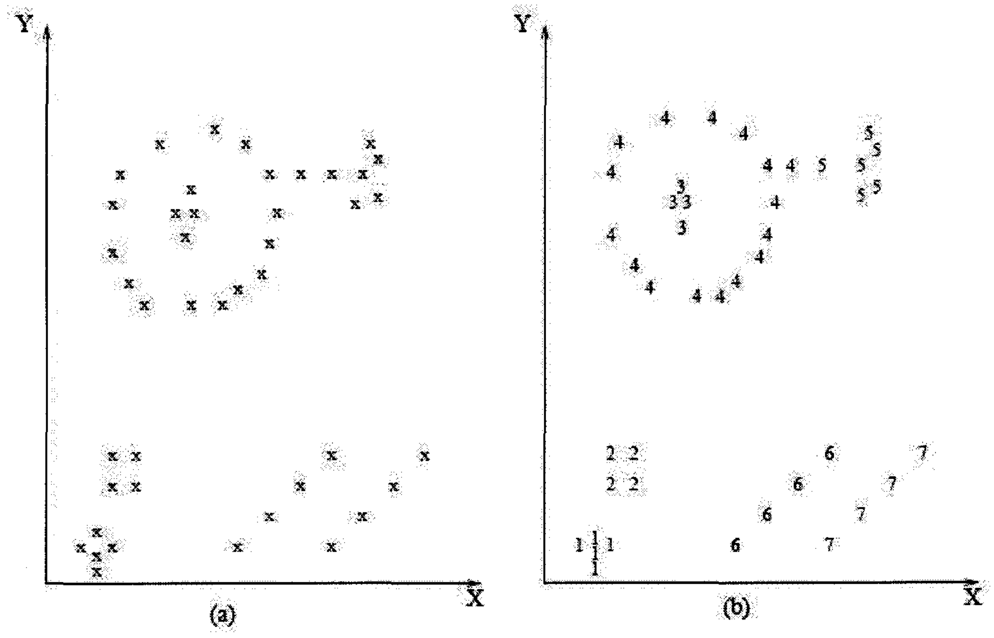

Figure 1.1: Data Clustering.

It is important to stress the difference between clustering (unsupervised classification) and discriminant analysis (supervised classification). In supervised classification, some predefined labels are available for the data sets, and a collection of already labeled entities are given (known as training set); the task is to classify a newly encountered, yet unlabeled, pattern into one of the predefined groups associated with the labels. Typically, the training set is used to learn a classification scheme which in turn is used to label or predict a new pattern's group. However, in the case of clustering, the task is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters too, but these category labels are data driven; i.e., they are learned solely from the data.

Clustering techniques have been applied to a wide variety of research problems. It also plays an important role in solving many engineering and practical problems. We will list some of them following, it is by no means exhaustive though.

- **Medicine**: Clustering diseases, cures for diseases, or symptoms of diseases can lead to very useful taxonomies [60].
- **Archeology**: Researchers have attempted to establish taxonomies of stone tools, funeral objects, etc. by applying cluster analytic techniques [60].

- **Image segmentation**: Different clustering algorithms are used to obtain segment labels for each pixel of a image, in order to identify potential classifications image pixels [49].

- **Information retrieval**: A knowledge-based clustering scheme can usually enhance the efficiency of information retrieval greatly [21].

- **Market analysis**: In market analysis, clustering analysis is often used in Market Segmentation, which describes the division of a market into homogeneous groups which will respond differently to promotions, communications, advertising and other marketing mix variables [28].

- **Software reverse engineering**: Software system evolve over time and their original design is constantly being modified to reflect the results of a series of corrective, perfective, or enhancing maintenance activities. in this context, architectural recovery is a key activity in supporting maintenance tasks such as re-engineering, objectification or restructuring. This discipline is the so-called software reverse engineering. Clustering techniques could be applied to extract the components or subsystems of the legacy system, it is believed in [47] that partition clustering is the suitable way for the recovery of cohesive subsystems in order to obtain higher quality design for the system. A more comprehensive review about this field could be found in [58].

Since, in all the above-listed cases, there is little prior information (e.g., statistical models) available about the data, and the decision-maker must make as few assumptions as possible about the data. It is under these restrictions that clustering methodology is particularly appropriate for the exploration of interrelationships among the data entities to extract an assessment of their structure. [28] gives an excellent summary about the various circumstances where clustering analysis could be applied.

1.2 Similarity Measures

Similarity is fundamental to the definition of a cluster, together with feature selection, it is the first step in a clustering analysis task. It is of essence that how the distance between two patterns are defined in the feature space. Therefore, we list some typical similarity measures that are commonly used in practice, it can roughly be divided to two categories: distance and measure.

1.2.1 Distance

The similarity of two patterns can simply be measured by distance between them, there are a number of distances which could be used to measure the similarity of the patterns:

- **Minkowski Distance**

  Minkowski Distance is defined by

$$\begin{aligned} d_p(s_i, s_j) &= (\sum_{t=1}^{d}(s_{it} - s_{jt})^p)^{1/p} \\ &= \|s_i - s_j\|_p, \end{aligned}$$

  where the s_i and s_j represent the entities in the data set.

  Minkowski distance is the generalization of several well-known distances, if $p = 1$, it is the 'City block' distance; If $p = \infty$, it is also know as Chebyshev distance. When $p = 2$, it is just *Euclidean Distance*:

$$\begin{aligned} d_2(s_i, s_j) &= (\sum_{t=1}^{d}(s_{it} - s_{jt})^2)^{1/2} \\ &= \|s_i - s_j\|_2. \end{aligned}$$

- **Lance Distance**

$$d_L^*(s_i, s_j) = \frac{1}{m}\sum_{t=1}^{d}\frac{|s_{it} - s_{jt}|}{s_{it} + s_{jt}}.$$

- **Mahalanobis Distance**

  Since the linear correlation among entities can distort distance measures, it could be alleviated by applying a whitening transformation to the data or by using the squared Mahalanobis distance

$$d_M(s_i, s_j) = (s_i - s_j)\Sigma^{-1}(s_i - s_j)^T$$

where the entities s_i and s_j are assumed to be row vectors, and Σ is the sample covariance matrix of the patterns or the known covariance matrix of the pattern generation process; d_M assigns different weights to different feature based on their variances and pairwise linear correlations.

1.2.2 Measure

The similarity of two patterns can also be measured by measures instead of distance. In what follows, we will give an example of the measures.

- **Czekanowski Coefficient**

$$m(s_i, s_j) = 1 - \frac{2\sum_{t=1}^{d} \min(s_{it}, s_{jt})}{\sum_{t=1}^{d}(s_{it} + s_{jt})}.$$

Another common measure is association-based similarity, emerging in relationship-based Clustering, one typical example is Jaccard similarity, for an introduction in more details, we refer to [52].

Among the various distance or measure metrics above, *Euclidean Distance* is nevertheless the one that is most widely adopted in practice, especially in the so-called minimal sum-of-squared criterion (MSSC), where the sum of squared Euclidean distance from each entity to its assigned cluster center is minimized. MSSC is the most intuitive and broadly used criterion for numerous clustering algorithms [28].

1.3 Clustering Algorithms

The main task of clustering analysis is to organize data by abstracting underlying structure either as a grouping of individuals or as a hierarchy of groups. It follows naturally that clustering methods can be grouped into one of two main categories: partitioning methods or hierarchical methods, depending on the strategies used in clustering.

In hierarchical clustering, an objective function is used locally as the merging or splitting criterion. In general, hierarchical algorithms can not provide optimal partitions for their criterion. To the contrast, partitional methods assume given the number of clusters to be found and then look for the optimal partition based on the objective function. The most important distinction between hierarchical and partitional approach is that hierarchical methods produce a nested series of partitions while partitional methods produce only one. Most partitional methods can be further classified as deterministic or stochastic, depending on whether the traditional optimization technique or a random search of the state space is used in the process. Figure 1.2 from [28] gives a good taxonomy for the different methods for clustering analysis.

We will give a brief review about the these two main approaches in the

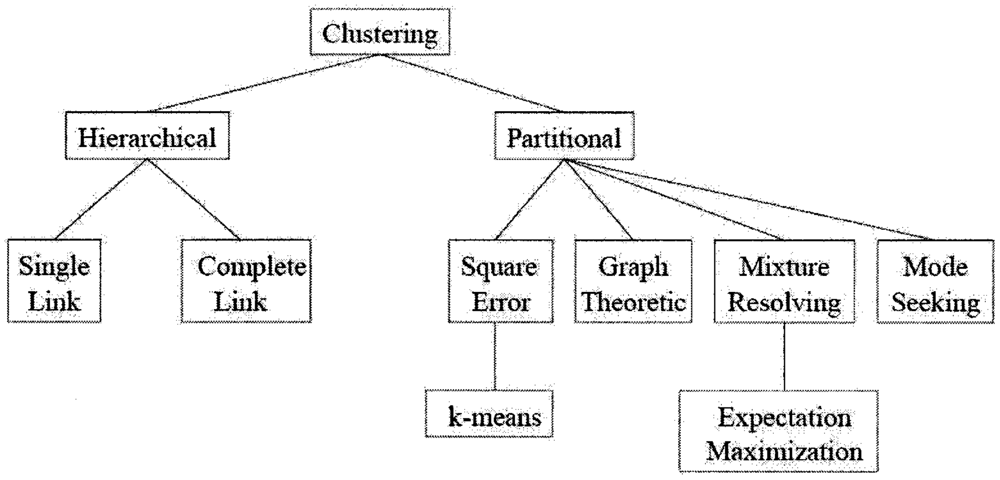

Figure 1.2: A taxonomy of clustering approaches.

remaining part of this section. To make it concise, the review focuses on the existing approaches that are related to the work in this thesis. For a more comprehensive survey for all these clustering methods in Figure 1.2, we refer to [28].

1.3.1 Hierarchical Clustering Methods

A hierarchical method yields a *dendrogram* representing the nested grouping of the pattern and similarity levels at which grouping changes. When there is a natural hierarchical structure underlying in the data, gene expression data in bioinfomatics for instance, hierarchical method is spontaneously the right algorithm to resort to.

There are mainly two categories of the hierarchical methods in terms of the way to construct the *dendrogram*: agglomerative or divisive. For the agglomerative approach, every pattern is treat as a single cluster in the beginning and it is agglomerated with other patterns to form a higher level cluster, until some stopping criterions are met. The divisive approach works in the reverse way, the whole data set are perceived as on cluster at first and is divided into two clusters, namely bi-clustering at each step to from a hierarchical structure.

There are two local criterions for merging or splitting in the hierarchical clustering algorithms: single-link [50] or complete-link [40; 56]. These two criterions differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the *minimum* of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters. In either case, two clusters are merged to form a larger cluster based on minimum distance criteria. It has been observed that the complete-link algorithm produces tightly bound or compact clusters [4]. The single-link algorithm, by contrast, suffers from a chaining effect [41]. It has a tendency to produce clusters that are straggly or elongated. Figure 1.3 and Figure 1.4 gives an illustration of hierarchical clustering by single-link criterion. The dendrogram can be broken at different levels to yield different clusterings of data. Figure 1.4 can be interpreted either as agglomerative, if constructed bottom-up, or divisive if constructed top-down.

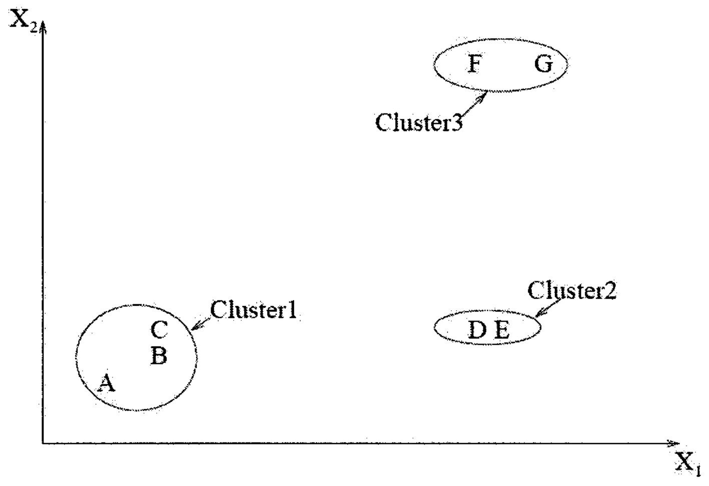

Figure 1.3: Points falling in three clusters.

In terms of algorithms design, one famous agglomerative clustering algorithm is in [56], Ward constructed an agglomerative approach has a complexity of $O(n^2 \log n)$ where n is the number of entities, based on the MSSC criterion. In general, divisive hierarchical clustering is more difficult, however, in low dimension, Hansen provided an algorithm running in $O(n^{d+1} \log n)$ time in [23], where d is the dimension of the space to which the entities belong.

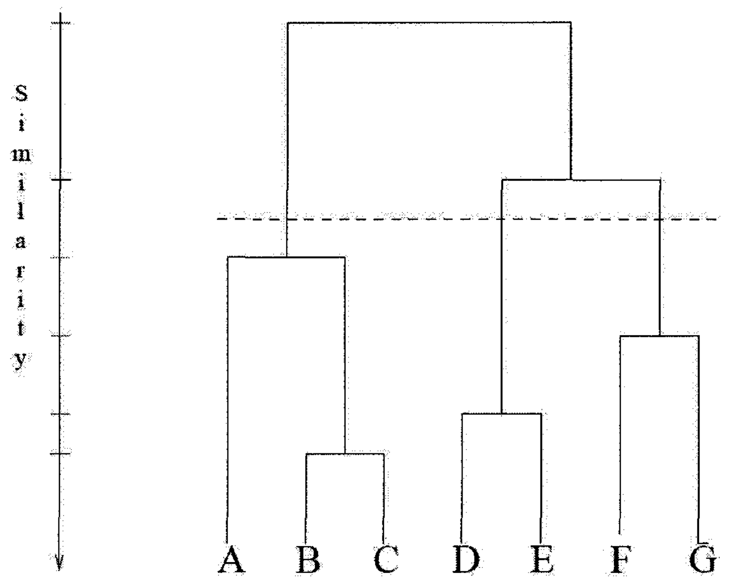

Figure 1.4: The dendrogram obtained using the single-link algorithm.

1.3.2 Partitioning Clustering Methods

A partitional clustering algorithm obtains a single partition from the data set. It is used when the data set is very large, where a hierarchical algorithm is prohibitive due to its high computational cost. The partitional approach usually produces clustering by optimizing a criterion function defined either locally or globally. The most common used criterion is the one we induced earlier, the MSSC. Partitional algorithms optimizing this criterion is also known as Squared Error Algorithm, the classical K-means algorithm is a typical example of the Squared Error Algorithm.

Among various partitioning methods, the classical K-means algorithm is the most popular clustering algorithm so far, and many partitioning algorithms are heuristics based on K-means and its variants [25; 43]. It was first introduced by MacQueen in 1967 [39]. Since then, different versions of this algorithm have been studied by many authors [51], K-means algorithm usually uses a criterion that minimizes the sum-of-squared Euclidean distance from each entity to its assigned cluster center, namely the MSSC. More rigorously, given a set S of n points in a d-dimensional Euclidean space [1], denoted by

$$S = \{\mathbf{s}_i = (s_{i1}, \cdots, s_{id})^T \in \mathbf{R}^d \quad i = 1, \cdots, n\}$$

the task of a partitional MSSC is to find an assignment of the n points into k disjoint clusters $\mathcal{S} = (\mathcal{S}_1, \cdots, \mathcal{S}_k)$ centered at cluster centers $\mathbf{c}_j$ $(j = 1, \cdots, k)$ based on the total sum-of-squared Euclidean distances from each point $\mathbf{s}_i$ to its assigned cluster centroid $\mathbf{c}_i$, i.e.,

$$f(\mathcal{S}) = \sum_{j=1}^{k} \sum_{i=1}^{|\mathcal{S}_j|} \left\| \mathbf{s}_i^{(j)} - \mathbf{c}_j \right\|^2, \tag{1.1}$$

where $|\mathcal{S}_j|$ is the number of points in $\mathcal{S}_j$, and $\mathbf{s}_i^{(j)}$ is the i^{th} point in $\mathcal{S}_j$. Note that if the cluster centers are known, then the function $f(S)$ achieves its minimum when each point is assigned to its closest cluster center. When objects within each cluster are distributed according to a spherical Gaussian, with the same

[1]In this thesis, we always assume that $n \geq k > 1$, because otherwise the underlying clustering problem becomes trivial.

covariance for all clusters, MSSC is a good measure [14].

Geometrically speaking, assigning each point to the nearest center fits into a framework called *Voronoi Program*, and the resulting partition is named *Voronoi Partition*. On the other hand, if the points in cluster $\mathcal{S}_j$ are fixed, then the function

$$f(\mathcal{S}_j) = \sum_{i=1}^{|\mathcal{S}_j|} \left\| \mathbf{s}_i^{(j)} - \mathbf{c}_j \right\|^2$$

is minimal when

$$\mathbf{c}_j = \frac{1}{|\mathcal{S}_j|} \sum_{i=1}^{|\mathcal{S}_j|} \mathbf{s}_i^{(j)}.$$

The classical K-means algorithm [39], based on the above two observations, is described in Algorithm 1:

Algorithm 1 K-means Clustering Algorithm

```
Step 1: Choose $k$ cluster centers randomly generated in a domain containing all the points,
Step 2: Assign each point to the closest cluster center,
Step 3: Recompute the cluster centers using the current cluster memberships,
Step 4: If a convergence criterion is met, stop; Otherwise go to step 2.
```

Figure 1.5 illustrates how K-means algorithm works, it iteratively update the centroids of the clusters and the partition derived by these centroids until convergence. Essencially, K-means algorithm performs coordinate descent in the objective function (1.1).

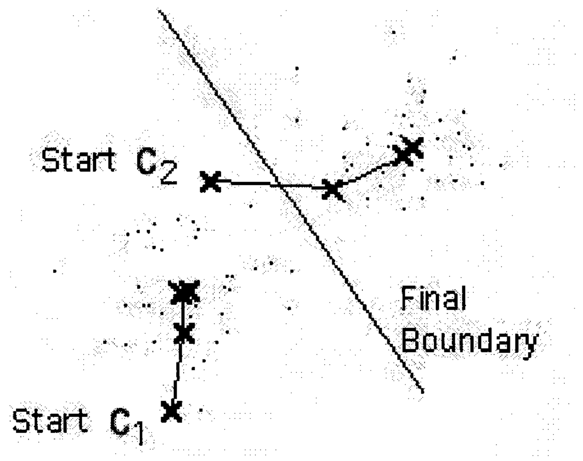

Figure 1.5: K-means algorithm illustration.

The complexity of K-means algorithm is $O(tkn)$, where t is the number of times we run the algorithm, k is the number of clusters and n is the number of patterns. It is the low complexity and efficiency that makes K-means extremely popular in practice. However, although widely adopted in partitioning clustering, K-means also has some drawbacks:

1. K-means is sensitive to initial choice of cluster centers, the clustering can be very different by starting from different centers.

2. K-means tends to converge to a local optimum, in most cases K-means can not find the global minimum of the measurement used in the model.

3. There is no approximation bound available for K-means and its variants.

4. In the case of constrained clustering which we will introduce in next section, K-means can not be applied directly.

The main motivation of this research is to attach the above-mentioned issues.

1.3.3 Other Clustering Methods

Besides hierarchical and partition cluster methods, there are also some other clustering methods proposed by expertise from various disciplines. These methods depict the ideas of different subjects and exhibit the innate multidisciplinary nature of clustering analysis. We list some typical methods and their basic ideas here, for a more comprehensive introduction, we still refer to [5; 28].

K-Nearest Neighbor(KNN) Clustering: In this approach, each unlabeled pattern is assigned to the cluster of its k nearest labeled neighbors as long as the average distance to the k neighbors is below a threshold [33].

Mixture Models: In the mixture model approach, we have an underlining assumption that the patterns are drawn from one of several distributions, and the goal is to identify the parameters of each model. The most common assumption is that individual components of the mixture density are Gaussian, and in this case the parameters of the individual Gaussians are to be estimated by the procedure. The most famous algorithm based on this approach is Expectation Maximization (EM) algorithm, a general purpose maximum likelihood algorithm for missing-data problems [12].

Artificial Neural Networks (ANN): ANN approaches have been used extensively for both classification and clustering [48]. The most common ANNs used for clustering include Kohonens learning vector quantization (LVQ) and self organizing map (SOM) [31] and adaptive resonance theory (ART) models [10]. These networks have simple architectures with single layers and the weights are learnt by iteratively changing them until a termination criterion is satisfied. These learning or weight changing procedures are similar to some used in classical clustering approaches. For example the procedure used in the LVQ is similar to the K-means algorithm.

Fuzzy Clustering: In traditional clustering methods, each pattern belongs to one and only one cluster. Therefore, the resultant clusters are disjoint. Fuzzy clustering extends this notion to associate each pattern with every cluster using a membership function [62]. The output of this approach is not a partition of the patterns.

Global Search: Including well known heuristic approaches to global optimization: Genetic Algorithm (GA), Simulated Annealing (SA) and Tabu Search (TS), although these global search methods performs better than K-means, they suffer from sensitivity to the selection of the control parameters. [43] gives a comparison of the performance of these heuristics and a hybrid system based on them is proposed.

1.4 Constrained Clustering

In many applications, finding clusters that satisfy user-specified constraints is highly desirable. This leads to the so-called constrained clustering, which was first introduced in [7]. [53] gives a taxonomy of constraints for clustering in the application perspective, for self-completeness, we list it as below:

1. **Constrained on individual objects**: This constraint confines the set of objects to be clustered, e.g., cluster only luxury mansions of value over one million dollars. It can be easily handled by preprocessing where we conduct a feature selection or reduce the patterns to those consist

with the constraints. The problem is then turned into a ordinary unconstrained clustering problem.

2. **Obstacle objects as constraints**: For example, we want to cluster some geographical data for a city, the city may have rivers, bridges, highways, lakes, mountains, etc. Such obstacles and their effects must be captured in the clustering process, this can be done by adding restriction on the distance function among objects. Notice that the Must-Link and Cannot-Link constraints belong to this category. In [11], these two constraints are studied and a variant of K-means algorithm is proposed.

3. **Clustering parameters as "constraints"**: Some "constraints" may serve as the parameters in a clustering algorithm, e.g., the number of clusters, k. Such parameters, since specifiable by users, usually are not considered as constraint.

4. **Constraints imposed on each individual cluster**: These constraints corresponds to that each individual cluster is confined to some predefined constraints, most typically, constraints on the cardinality of each resultant clusters.

The fourth case in the above list is also called balanced clustering. A special case of balanced-clustering is that the number of entities in each resultant cluster is fixed. Both of these two cases have extensive applications in practice. For instance, in the design of wireless sensor networks, it is desirable to have a network whose load is balanced. Another example is in clustering analysis arising from market analysis, a direct marketing campaign often starts with segmenting customers into groups of roughly equal size or equal estimated revenue generation, (based on, say, market basket analysis, or purchasing behavior at a web site), so that the same number of sales teams (or marketing dollars) can be allocated to each segment. This is the theme of the second part this thesis.

1.5 Organization of this thesis

The thesis is organized as follows. In chapter 2, we review the clustering literature from an optimization perspective and some related works are addressed. In chapter 3, we first transfer the K-means clustering into a novel optimization model, 0-1 semidefinite programming where the eigenvalues of involved matrix argument must be 0 or 1, this new optimization model also allows us to attack the original problem based on the relaxed linear and semidefinite programming. In chapter 4, we consider the approximate algorithms for solving the 0-1 SDP, a new approximation method which extracts a feasible clustering via PCA (Principal Component Analysis) is proposed. It could also be shown that our algorithm can provide a 2-approximation to the global solution of the original problem. This two chapters compose of the first part of this thesis. In the following part, we first introduce the so-called balanced bi-clustering problem and propose several algorithms to tackle this problem in chapter 5, and an approximation algorithm with an upper bound is proposed. In chapter 6, based on the idea in chapter 5, we focus on the fixed size bi-clustering problem, an algorithm by reformulating the problem to a Quadratic Programming (QP) problem is proposed, it enjoys a lower computational complexity and has a nice approximate bound in the mean time. Preliminary computational results are reported in chapter 7 for the methods proposed in previous sections, and finally, we conclude the thesis by a few remarks.

1.6 Contributions

Finally, we summarize our contributions in this thesis here. Firstly, we come up with a united framework called 0-1 SDP, which is not only able to encompass the MSSC model but also several other clustering criterions. Second, a general scheme for relaxation algorithms and some approximation algorithms based on LP/SDP relaxation are proposed. In particular, an approximation algorithm based on PCA is addressed and we prove that it could give us a 2-approximation solution to the global optimum. In the remaining part of the thesis, we consider specifically the balanced bi-clustering, an improved algorithm based on an existing one and an algorithm with an approximation bound are proposed. Finally, we focus on the bi-clustering problem with fixed size, a novel heuristic called Q-means is proposed to attach this problem. Numerical experiments are performed to validate the performance our algorithms.

Chapter 2

Literature Review

In this chapter, we will give a brief overview about the optimization techniques used in partition clustering methods and some related approximation algorithms for (balanced) clustering proposed in literature.

2.1 Clustering Analysis and Optimization

While a few clustering problems were expressed as mathematical programs before, systematic use of this approach was only advocated about 25 years ago [46; 55]. For partition methods, typical optimization algorithms applied include: dynamic programming, graph theoretical algorithms, branch-and-bound, cutting planes, column generation and heuristics. [24] gives an excellent review for the use of mathematical programming in general clustering analysis. Since our work focus on the partition method with MSSC model, we will elaborate more for this special case.

Recall the MSSC objective function (1.1), it can be described by the following bilevel programming problem (see for instance [9; 35]).

$$\min_{\mathbf{c}_1,\cdots,\mathbf{c}_k} \sum_{i=1}^{n} \min\{\|\mathbf{s}_i - \mathbf{c}_1\|^2, \cdots, \|\mathbf{s}_i - \mathbf{c}_k\|^2\}. \tag{2.1}$$

Figure 2.1 illustrates a plot of the MSSC function in $\mathbf{R}^1$ for a data set with 20 points. If $k > 1$, from an optimization point of view, this function is nonconvex and nonsmooth. It can be shown that this function has many shallow local minimizers, which are close to each other. This is why K-means algorithm is easily trapped into local optimum.

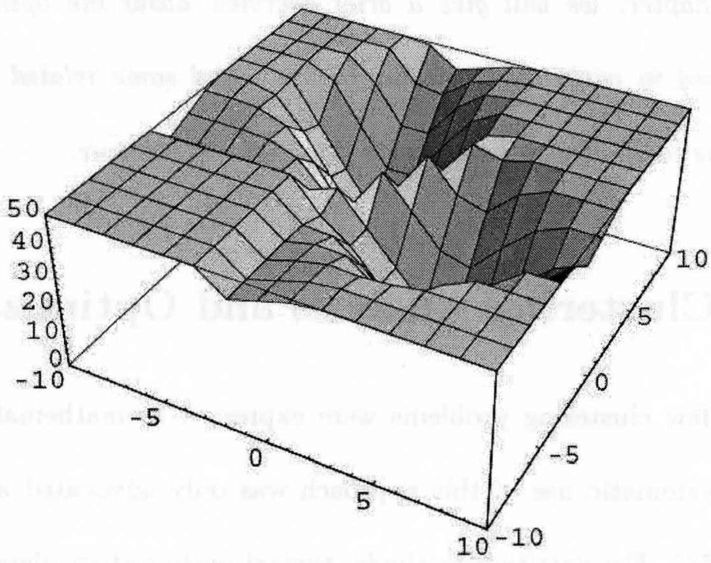

Figure 2.1: Objective function in $\mathbf{R}^2$.

Another way to model MSSC is based on the assignment. Let $X = [x_{ij}] \in \Re^{n\times k}$ be the assignment matrix defined by

$$x_{ij} = \begin{cases} 1 & \text{If } \mathbf{s}_i \text{ is assigned to } S_j; \\ 0 & \text{Otherwise.} \end{cases}$$

As a consequence, the cluster center of the cluster S_j, as the mean of all the points in the cluster, is defined by

$$\mathbf{c}_j = \frac{\sum_{l=1}^{n} x_{lj}\mathbf{s}_l}{\sum_{l=1}^{n} x_{lj}}.$$

Using this fact, we can represent (2.1) as

$$\begin{aligned}
\min_{x_{ij}} \quad & \sum_{j=1}^{k}\sum_{i=1}^{n} x_{ij}\left\|\mathbf{s}_i - \frac{\sum_{l=1}^{n} x_{lj}\mathbf{s}_l}{\sum_{l=1}^{n} x_{lj}}\right\|^2 && (2.2)\\
S.T. \quad & \sum_{j=1}^{k} x_{ij} = 1 \ (i = 1, \cdots, n) && (2.3)\\
& \sum_{i=1}^{n} x_{ij} \geq 1 \ (j = 1, \cdots, k) && (2.4)\\
& x_{ij} \in \{0,1\} \ (i = 1, \cdots, n; \ j = 1, \cdots, k) && (2.5)
\end{aligned}$$

The constraint (2.3) ensures that each point $\mathbf{s}_i$ is assigned to one and only one cluster, and (2.4) ensures that there are exactly k clusters. This is a usually large-scale mixed integer programming with nonlinear objective [24], which is NP-hard. It is complex enough not to be amenable to the direct application of general-purpose global optimization methods. The difficulty of the problem consists of two parts. First, the constraints are discrete. Secondly the objective is nonlinear and nonconvex. Both the difficulties in the objective as well as in the constraints make MSSC extremely hard to solve. Therefore, in order to ensure the practicality of the nonsmooth optimization approach to clustering, proper identification and use of local optimization methods is very important. Some heuristics, like K-means and its variants, can be interpreted as one kind of these local optimization methods.

2.2 Related Work

As the difficulty we mentioned above, approximation methods play an very important role for solving (2.2). There are several different ways to approximate (2.2). For example, by solving the so-called K-Medians problem we can obtain a 2-approximately optimal solution for (2.2) in $O(n^{k+1})$ time [22]. In [34], Mutousek proposed a geometric approximation method that can find an $(1 + \epsilon)$ approximately optimal solution for (2.2) in $O(n \log^k n)$ time, where the constant hidden in the big-O notation depends polynomially on ϵ^{-1}. Although theoretically efficient, no numerical results have been reported based on Mutousek's algorithm. Another efficient way of approximation is to attack the original problem (typically NP-hard) by solving a relaxed polynomially solvable problem. This has been well studied in the field of optimization, in particular, in the areas of combinatorial optimization and semidefinite programming [18]. We noted that recently, Xing and Jordan [59] considered the SDP relaxation for the so-called normalized k-cut spectral clustering.

For balanced clustering, the related work is relatively less. In [8], Bradley et'al proposed a so-called constrained K-means algorithm for balanced clustering, where a Linear Programming (LP) problem is solves iteratively at each step, one of our algorithms is to improve the constrained K-means in the case of bi-clustering. Other constrained clustering problems are discussed in [7; 53].

Chapter 3

0-1 SDP Model for K-means clustering

In this chapter, we first establish the equivalence between K-means type clustering and the so-called 0-1 SDP model in the first section, in the following section, the interrelation between 0-1 SDP and other clustering approaches is elaborated.

3.1 From MSSC to 0-1 SDP

3.1.1 0-1 Semidefinite Programming

In general, SDP refers to the problem of minimizing (or maximizing) a linear function over the intersection of a polyhedron and the cone of symmetric and positive semidefinite matrices. The canonical SDP takes the following form

$$\textbf{(SDP)}\begin{cases} \min & \mathrm{Tr(WZ)} \\ S.T. & \mathrm{Tr(B_iZ) = b_i} \quad \text{for} \quad \mathrm{i = 1, \cdots, m} \\ & Z \succeq 0 \end{cases}$$

Here Tr(.) denotes the trace of the matrix, and $Z \succeq 0$ means that Z is positive semidefinite. If we replace the constraint $Z \succeq 0$ by the requirement that $Z^2 = Z$, then we end up with the following problem

$$\textbf{(0-1 SDP)}\begin{cases} \min & \mathrm{Tr(WZ)} \\ S.T. & \mathrm{Tr(B_iZ) = b_i} \quad \text{for} \quad \mathrm{i = 1, \cdots, m} \\ & Z^2 = Z, Z = Z^T \end{cases}$$

We call it 0-1 SDP owing to the similarity of the constraint $Z^2 = Z$ to the classical 0-1 requirement in integer programming.

3.1.2 Equivalence of MSSC to 0-1 SDP

In this subsection we show that MSSC can be modelled as 0-1 SDP. We mention that the equivalence between MSSC and 0-1 SDP was first established in [44]. However, for self-completeness, we still give a detailed description of the reformulation process.

By rearranging the items in the objective of (2.2), we have

$$\begin{aligned} f(S,\mathcal{S}) &= \sum_{i=1}^{n} \|\mathbf{s}_i\|^2 \left(\sum_{j=1}^{k} x_{ij}\right) - \sum_{j=1}^{k} \frac{\|\sum_{i=1}^{n} x_{ij}\mathbf{s}_i\|^2}{\sum_{i=1}^{n} x_{ij}} \\ &= \mathrm{Tr}(\mathrm{W_S W_S^T}) - \sum_{\mathrm{j=1}}^{\mathrm{k}} \frac{\|\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{x_{ij}s_i}\|^2}{\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{x_{ij}}}, \end{aligned} \tag{3.1}$$

where $W_S \in \Re^{n\times d}$ denotes the matrix whose i-th row is the transfer s_i^T of the vector $\mathbf{s}_i$. Since X is an assignment matrix, we have

$$X^T X = \mathrm{diag}\,(\sum_{i=1}^{n} x_{i1}^2, \cdots, \sum_{i=1}^{n} x_{ik}^2) = \mathrm{diag}\,(\sum_{i=1}^{n} x_{i1}, \cdots, \sum_{i=1}^{n} x_{ik}).$$

Let

$$Z := [z_{ij}] = X(X^T X)^{-1} X^T,$$

we can write (3.1) as $\mathrm{Tr(W_S W_S^T(I-Z))} = \mathrm{Tr(W_S^T W_S)} - \mathrm{Tr(W_S^T W_S Z)}$. Obviously Z is a projection matrix satisfying $Z^2 = Z$ with nonnegative elements. For any integer m, let e^m be the all one vector in $\Re^m$. We can write the constraint (2.3) as

$$Xe^k = e^n.$$

It follows immediately

$$Ze^n = ZXe^k = Xe^k = e^n.$$

Moreover, the trace of Z should equal to k, the number of clusters, i.e.,

$$\mathrm{Tr(Z) = k.}$$

Therefore, we have the following 0-1 SDP model for MSSC

$$\begin{aligned} \min \quad & \mathrm{Tr}\big(\mathrm{W_S W_S^T (I - Z)}\big) \\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z = Z^T, Z^2 = Z. \end{aligned} \tag{3.2}$$

We first give a technical result about positive semidefinite matrix that will be used in our later analysis.

Lemma 3.1.1. *For any symmetric positive semidefinite matrix $Z \in \Re^{n \times n}$, there exists an index $i_0 \in \{1, \cdots, n\}$ such that*

$$Z_{i_0 i_0} = \max_{i,j} Z_{ij}.$$

Proof. For any positive semidefinite matrix Z, it is easy to see that

$$Z_{ii} \geq 0, \quad i = 1, \cdots, n.$$

Suppose the statement of the lemma does not hold, i.e., there exists $i_0 \neq j_0$ such that

$$Z_{i_0 j_0} = \max_{i,j} Z_{ij} > 0.$$

Then the submatrix

$$\begin{pmatrix} Z_{i_0 i_0} & Z_{i_0 j_o} \\ Z_{j_0 i_0} & Z_{j_0 j_0} \end{pmatrix}$$

is not positive semidefinite. This contradicts to the assumptuion in the lemma.

Now we are ready to establish the equivalence between the models (3.2) and (2.2).

Theorem 3.1.2. *Solving the 0-1 SDP problem (3.2) is equivalent to finding a global solution of the integer programming problem (2.2).*

Proof. From the construction of the 0-1 SDP model (3.2), we know that one can easily construct a feasible solution for (3.2) from a feasible solution of (2.2). Therefore, it remains to show that from a global solution of (3.2), we can obtain a feasible solution of (2.2).

Suppose that Z is a global minimum of (3.2). Obviously Z is positive semidefinite. From Lemma 3.1.1 we conclude that there exists an index i_1 such that

$$Z_{i_1 i_1} = \max\{Z_{ij} : 1 \leq i, j \leq n\} > 0.$$

Let us define the index set

$$\mathcal{I}_1 = \{j : Z_{i_1 j} > 0\}.$$

Since $Z^2 = Z$, we have

$$\sum_{j\in\mathcal{I}_1}(Z_{i_1j})^2 = Z_{i_1i_1},$$

which implies

$$\sum_{j\in\mathcal{I}_1}\frac{Z_{i_1j}}{Z_{i_1i_1}}Z_{i_1j} = 1.$$

From the choice of i_1 and the constraint

$$\sum_{j=1}^{n} Z_{i_1j} = \sum_{j\in\mathcal{I}_1} Z_{i_1j} = 1,$$

we can conclude that

$$Z_{i_1j} = Z_{i_1i_1}, \quad \forall j \in \mathcal{I}_1.$$

This further implies that the submatrix $Z_{\mathcal{I}_1\mathcal{I}_1}$ is a matrix whose elements are all equivalent, and we can decompose the matrix Z into a bock matrix with the following structure

$$Z = \begin{pmatrix} Z_{\mathcal{J}_1\mathcal{J}_1} & 0 \\ 0 & Z_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1} \end{pmatrix}, \tag{3.3}$$

where $\bar{\mathcal{I}}_1 = \{i : i \notin \mathcal{I}_1\}$. Since $\sum_{i\in\mathcal{I}_1} Z_{ii} = 1$ and $(Z_{\mathcal{I}_1\mathcal{I}_1})^2 = Z_{\mathcal{I}_1\mathcal{I}_1}$, we can consider the reduced 0-1 SDP as follows

$$\min \quad \mathrm{Tr}\Big((\mathrm{W_S W_S^T})_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1}(\mathrm{I}-\mathrm{Z})_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1}\Big) \tag{3.4}$$

$$Z_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1}e = e, \mathrm{Tr}(\mathrm{Z}_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1}) = \mathrm{k}-1,$$

$$Z_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1} \geq 0, Z^2_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1} = Z_{\bar{\mathcal{I}}_1\bar{\mathcal{I}}_1}.$$

Repeating the above process, we can show that if Z is a global minimum of the 0-1 SDP, then it can be decomposed into a diagonal block matrix as

$$Z = \operatorname{diag}(Z_{\mathcal{I}_1\mathcal{I}_1}, \cdots, Z_{\mathcal{I}_k\mathcal{I}_k}),$$

where each block matrix $Z_{\mathcal{I}_l\mathcal{I}_l}$ is a nonnegative projection matrix whose elements are equal, and the sum of each column or each row equals to 1.

Now let us define the assignment matrix $X \in \Re^{n\times k}$

$$X_{ij} = \begin{cases} 1 & \text{if } i \in \mathcal{I}_j \\ 0 & \text{otherwise} \end{cases}$$

One can easily verify that $Z = X(X^TX)^{-1}X^T$. Our above discussion illustrates that from a feasible solution of (3.2), we can obtain an assignment matrix that satisfies the condition in (2.2). This finishes the proof of the theorem.

Note that for a given data set $\mathcal{S}$, the trace $\mathrm{Tr}\left(\mathrm{W_S W_S^T}\right)$ becomes a fixed quantity. Therefore, we can solve the MSSC model via the following optimization problem

$$\begin{aligned} \max \quad & \mathrm{Tr}\left(\mathrm{W_S W_S^T Z}\right) \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z = Z^T, Z^2 = Z. \end{aligned} \tag{3.5}$$

To distinguish the above problem from the original MSSC model, we call the objective in the above formulation as the refined objective for the MSSC model.

It is worthwhile comparing (3.2) with (2.2). First, the objective in (3.2) is linear while the constraint in (3.2) is still nonlinear, even more complex than the 0-1 constraint in (2.2). The most difficult part in the constraint of (3.2) is the requirement that $Z^2 = Z$. Several different ways for solving (3.2) will be discussed in the next chapter.

3.2 0-1 SDP Reformulation for Other Clustering Approaches

In this subsection, we show that the 0-1 SDP can also be used for other clustering approaches based on other measurements. Let us consider the more general 0-1 SDP model for clustering

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I - Z))} \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{3.6}$$

where W is the so-called affinity matrix whose entries represent the similarities or closeness among the entities in the data set. In the MSSC model, we use the geometric distance between two points to characterize the similarity between them. In this case, we have $W_{ij} = \mathbf{s}_i^T \mathbf{s}_j$. However, we can also use a general function $\phi(\mathbf{s}_i, \mathbf{s}_j)$ to describe the similarity relationship between $\mathbf{s}_i$ and $\mathbf{s}_j$. For example, let us choose

$$W_{ij} = \phi(\mathbf{s}_i, \mathbf{s}_j) = \exp^{-\frac{\|\mathbf{s}_i - \mathbf{s}_j\|^2}{\sigma}}, \quad \sigma > 0. \tag{3.7}$$

In order to apply the classical K-means algorithm to (3.6), we can first use the singular eigenvalue decomposition method to decompose the matrix W into the product of two matrices, i.e., $W = U^T U$. In this case, each column of U can be cast as a point in a suitable space. Then, we can apply the classical K-means method for MSSC model to solving problem (3.6). This is exactly the procedure what the recently proposed spectral clustering follows [3; 42; 57; 59; 61]. However, we now have a new interpretation for spectral clustering, i.e., a variant of MSSC in a different kernel space. It is worthwhile mentioning that certain variants of K-means, called weighted kernel K-means, can be adapted to solve (3.6) directly without using the SVD decomposition of the affinity matrix [13].

We note that recently, the k-ways normalized cut and spectral clustering received much attention in the machine learning community, and many interesting results about these two approaches have been reported [20; 36; 42; 49; 57; 59; 61; 63]. In particular, Zha et'al [63] discussed the links between spectral relaxation and K-means. Similar ideas was also used in [42]. An SDP relaxation for normalized k-cut was discussed [59]. The relaxed SDP in [59] takes a form quite close to (4.2). For self-completeness, we next describe briefly how the k-ways normalized cut can be embedded into the 0-1 SDP model. Let us first recall the exact model for normalized k-cut [59]. Let W be the affinity matrix defined by (3.7) and X be the assignment matrix in the set $\mathcal{F}_k$ defined by

$$\mathcal{F}_k = \{X : Xe^k = e^n, x_{ij} \in \{0,1\}\}.$$

Let $d = We^n$ and $D = \operatorname{diag}(d)$. The exact model for normalized k-cut in [59] can be rewritten as

$$\max_{X \in \mathcal{F}_k} \quad \mathrm{Tr}\big((X^T D X)^{-1} X^T W X\big) \tag{3.8}$$

If we define

$$Z = D^{\frac{1}{2}} X (X^T D X)^{-1} X^T D^{\frac{1}{2}},$$

then we have

$$Z^2 = Z, Z^T = Z, Z \geq 0, Zd^{\frac{1}{2}} = d^{\frac{1}{2}}.$$

Following a similar process as in the proof of Theorem 3.1.2, we can show that the model (3.8) equals to the following 0-1 SDP:

$$\begin{aligned} \min \quad & \mathrm{Tr}\Big(D^{-\frac{1}{2}} W D^{-\frac{1}{2}} (I - Z)\Big) \qquad (3.9) \\ & Zd^{\frac{1}{2}} = d^{\frac{1}{2}}, \mathrm{Tr}(Z) = k, \\ & Z \geq 0, Z^2 = Z, Z = Z^T. \end{aligned}$$

The only difference between (3.6) and (3.9) is the introduction of the scaling matrix D.

Except for the above mentioned cases, the 0-1 SDP model can also be applied to the so-called balanced clustering we introduced in first chapter [8], where the number of entities in every cluster is restricted. One special case of balanced clustering is requiring the number of entities in every cluster must be equal or large than a prescribed quantity, i.e., $|C_i| \geq \tilde{n}$. It is straightforward to see such a problem can be modelled as a 0-1 SDP by adding the constraint $Z_{ii} \leq \frac{1}{\tilde{n}}$ to (3.6), which leads to the following problem

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Z_{ii} \leq \frac{1}{\tilde{n}}, \quad i = 1, \cdots, n, \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{3.10}$$

.

These results shows the power of our 0-1 SDP model, it is the underlining framework for numerous different-first-look clustering algorithms and even balanced clustering.

Chapter 4

Approximation Algorithms for 0-1 SDP

A general scheme for approximation algorithms to solve the 0-1 SDP based on relaxation is addressed in the first part of this chapter, then a new approximation method based on PCA is proposed, finally the solutions from the new approximate algorithm is evaluated.

4.1 Relaxation Algorithms for Solving 0-1 SDP

In this section we discuss how to solve the 0-1 SDP model for clustering. For simplification of our discussion, we restrict us to the model (3.6). Throughout the paper, we further assume that the underlying matrix W is positive definite or semidefinite. This assumption is satisfied in the MSSC model as well as in the so-called spectral clustering where the kernel matrix is defined by (3.7). It is worth mentioning that although we restrict our discussion to (3.6), however, with a little effort, our results can be extended to (3.9) as well.

The section consists of three parts. In the first subsection, we propose a general scheme for solving (3.6). In the second part, an algorithm based on LP relaxation is proposed as an example. Finally, we introduce a new approximation method for (3.6).

4.1.1 General Scheme for Solving 0-1 SDP based on Relaxation

In this subsection, we discuss various algorithms for solving the 0-1 SDP model (3.6). From a viewpoint of the algorithm design, we can categorize all the algorithms for (3.6) into two groups. The first group consists of the so-called feasible iterative algorithms, where all the iterates are feasible regarding the constraints in (3.2) and the objective is increased step by step until some termination criterion is reached. The classical K-means algorithm described in the introduction can be interpreted as a special feasible iterative scheme for attacking (3.6). It is also easy to see that, many variants of the K-means algorithm can also be interpreted as specific iterative schemes for (3.6).

The second group of algorithms for (3.6) consists of approximation algorithms that are based on LP/SDP relaxation. We starts with a general procedure for those algorithm, which is depicted in Algorithm 2.

```
Algorithm 2 Approximation Algorithm Based on Relaxation
  Step 1: Choose a relaxation model for (3.2),
  Step 2: Solve the relaxed problem for an approximate solution,
  Step 3: Use a rounding procedure to extract a feasible solution to (3.2)
  from the approximate solution.
```

The relaxation step has an important role in the whole algorithm. For example, if the approximation solution obtained from Step 2 is feasible for (3.6), then it is exactly an optimal solution of (3.6). On the other hand, when the approximation solution is not feasible regarding (3.6), we have to use a rounding procedure to extract a feasible solution.

4.1.2 LP Relaxation

In this subsection, we propose an LP relaxation for (3.6), it is an example for the general relaxation scheme we introduced in last subsection. First we observe that if s_i and s_j, s_j nd s_k belong to the same clusters, then s_i and s_k belong to the same cluster. In such a circumstance, from the definition of the matrix Z we can conclude that

$$Z_{ij} = Z_{jk} = Z_{ik} = Z_{ii} = Z_{jj} = Z_{kk}.$$

Such a relationship can be partially characterized by the following inequality

$$Z_{ij} + Z_{ik} \leq Z_{ii} + Z_{jk}.$$

Correspondingly, we can define a metric polyhedron MET[1] by

$$MET = Z = [z_{ij}] : z_{ij} \leq z_{ii}, z_{ij} + z_{ik} \leq z_{ii} + z_{jk}.$$

Therefore, we have the following new model

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \qquad (4.1)\\ & Ze = e, \mathrm{Tr(Z) = k},\\ & Z \geq 0,\\ & Z \in MET. \end{aligned}$$

If the optimal solution of (4.1) is not a feasible solution of (3.6), then as we mentioned, we need to refer to the rounding procedure to extract a feasible solution for (3.6).

[1]A similar polyhedron MET had been used by Karisch and Rendl, Leisser and Rendl in their works [30; 32] on graph partitioning. We changed slightly the definition of MET in [32] to adapt to our problem.

Solving (4.1) directly for large-size data set is clearly unpractical due to the huge amount $O(n^3)$ of constraints. However, this LP relaxation model can still provide a lower bound for the global optimum of (3.6) for small scale data sets. Some preliminary results are reported in [44].

4.1.3 Other Related Work

Besides the LP relaxation, various relaxations and rounding procedures have been proposed for solving (3.6) in the literature as well. For example, Xing and Jordan [59] considered the SDP relaxation for normalized k-cuts and proposed a rounding procedure based on the singular value decomposition of the solution Z of the relaxed problem, i.e., $Z = U^T U$. In their approach, every row of U^T is cast as a point in the new space, and then the weighted K-means clustering is performed over the new set of those points in $\Re^k$. Similar works for spectral clustering can also be found in [20; 36; 42; 57; 63] where the singular value decomposition of the underlying matrix W is used and a K-means-type clustering based on the eigenvectors of W is performed. In the above-mentioned works, the solutions obtained from the weighted K-means algorithm for the original problem and that based on the eigenvectors of W has been compared, and simple theoretical bounds have been derived based on the eigenvalues of W.

The idea of using the singular value decomposition of the underlying matrix W is natural in the so-called principal component analysis (PCA) [29]. In [15], the link between PCA and K-means clustering was also explored and simple bounds were derived. In particular, Drineas et'al [17] proposed to use singular value decomposition to form a subspace, and then perform K-means clustering in the subspace $\Re^k$. They proved that the solution obtained by solving the K-means clustering in the reduced space can provide a 2-approximation to the solution of the original K-means clustering.

We note that in [49], Shi and Malik used the eigenvector of a projection matrix of W (not W itself) onto a subspace to construct a feasible partitioning for the original problem. Our work follow a similar idea as [49]. We first use singular value decomposition to obtain the $k-1$ eigenvectors corresponding to the first $k-1$ largest eigenvalues of a projection matrix of W, and then we perform K-means clustering in $\Re^{k-1}$. This allows us to improve the complexity of the algorithm for solving the subproblem in the reduced space. As we shall see later, such a rounding procedure can also provide a 2-approximation to the original problem with theoretical guarantee.

4.2 A New Approximation Method

In this section, we describe our SDP-based approximation method for (3.6). We start our discussion on various relaxation forms for (3.6).

First, recall that in (3.6), the argument Z is stipulated to be a projection matrix, i.e., $Z^2 = Z$, which implies that the matrix Z is a positive semidefinite matrix whose eigenvalues are either 0 or 1. A straightforward relaxation to (3.6) is replacing the requirement $Z^2 = Z$ by the relaxed condition

$$I \succeq Z \succeq 0.$$

Note that in (3.6), we further stipulate that all the entries of Z are nonnegative, and the sum of each row(or each column) of Z equals to 1. This means the eigenvalues of Z is always less than 1. In this circumstance, the constraint $Z \preceq I$ becomes superfluous and can be waived. Therefore, we obtain the following SDP relaxation [2]

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I - Z))} \\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z \succeq 0. \end{aligned} \tag{4.2}$$

The above problem is feasible and bounded below. We can apply many ex-

[2]In [59], the constraint $Ze = e$ in (3.6) is replaced by $Zd = d$, where d is a positive scaling vector associated with the affinity matrix.

isting optimization solvers such as interior-point methods to solve (4.2). It is known that an approximate solution to (4.2) can be found in polynomial time. However, we should point out that although there exist theoretically polynomial algorithm for solving (4.2), most of the present optimization solvers are unable to handle the problem in large scale efficiently.

Another interesting relaxation to (3.6) is to further relax (4.2) by dropping some constraints. For example, if we remove the nonnegative requirement on the elements of Z, then we obtain the following simple SDP problem

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} && (4.3)\\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & I \succeq Z \succeq 0. \end{aligned}$$

In the sequel we discuss how to solve (4.3). Note that if Z is a feasible solution for (4.3), then we have

$$\frac{1}{\sqrt{n}} Ze = \frac{1}{\sqrt{n}} e,$$

which implies $\frac{1}{\sqrt{n}}e$ is an eigenvector of Z corresponding to its largest eigenvalue 1. For any feasible solution of (4.3), let us define

$$Z_1 = Z - \frac{1}{n} ee^T.$$

It is easy to see that

$$Z_1 = (I - \frac{1}{n}ee^T)Z = (I - \frac{1}{n}ee^T)Z(I - \frac{1}{n}ee^T), \qquad (4.4)$$

i.e., Z_1 represents the projection of the matrix Z onto the null subspace of e. Moreover, it is easy to verify that

$$\mathrm{Tr}(\mathrm{Z}_1) = \mathrm{Tr}(\mathrm{Z}) - 1 = \mathrm{k} - 1.$$

Let W_1 denote the projection of the matrix W onto the null space of e, i.e.,

$$W_1 = (I - \frac{1}{n}ee^T)W(I - \frac{1}{n}ee^T). \tag{4.5}$$

Then, we can reduce (4.3) to

$$\begin{aligned} \min \quad & \mathrm{Tr}(\mathrm{W}_1(\mathrm{I} - \mathrm{Z}_1)) \\ & \mathrm{Tr}(\mathrm{Z}_1) = \mathrm{k} - 1, \\ & I \succeq Z_1 \succeq 0. \end{aligned} \tag{4.6}$$

Let $\lambda_1, \cdots, \lambda_{n-1}$ be the eigenvalues of the matrix W_1 listed in the order of decreasing values. The optimal solution of (4.6) can be achieved if and only if

$$\mathrm{Tr}(\mathrm{W}_1\mathrm{Z}_1) = \sum_{\mathrm{i}=1}^{\mathrm{k}-1} \lambda_{\mathrm{i}}.$$

This gives us an easy way to solve (4.6) and correspondingly (4.3). The algorithmic scheme for solving (4.3) can be described in Algorithm 3. From our above discussion, we immediately have:

Theorem 4.2.1. *Let Z^* be the global optimal solution of (3.6), and $\lambda_1, \cdots, \lambda_{k-1}$ be the first largest eigenvalues of the matrix W_1. Then we have*

$$\mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)) \geq \mathrm{Tr}(\mathrm{W}) - \frac{1}{\mathrm{n}}\mathrm{e}^{\mathrm{T}}\mathrm{We} - \sum_{\mathrm{i}=1}^{\mathrm{k}-1} \lambda_{\mathrm{i}}.$$

```
Algorithm 3 Relaxation Algorithm
  Step 1: Calculate the projection $W_1$ via (4.5);
  Step 2: Use singular value decomposition method to compute the first $k-1$
  largest eigenvalues of the matrix $W_1$ and their corresponding eigenvectors
  $v^1, \cdots, v^{k-1}$,
  Step 3: Set
      $Z = \frac{1}{n}ee^T + \sum_{i=1}^{k-1} v^i {v^i}^T.$
```

We point out that if $k = 2$, then Step 2 in the above algorithm uses the eigenvector corresponding to the largest eigenvalue of W_1. Our relaxation method is very similar to the one used by Shi and Malik [49] (See also [57]) for image segmentation where the normalized k-cut clustering problem with $k = 2$ was discussed. Similar bounds for normalized k-cuts and spectral clustering can also be found in [15; 42].

Note that solving the relaxed problem (4.3) can not provide a solution for the original problem (3.6). In the sequel we propose a rounding procedure to extract a feasible solution for (3.6) from a solution of the relaxed problem (4.3) provided by the relaxation Algorithm 1. Our rounding procedure follows a similar vein as the rounding procedure in [17]. Let us denote $V = (\sqrt{\lambda_1}v^1, \cdots, \sqrt{\lambda_{k-1}}v^{k-1}) \in \Re^{n\times(k-1)}$ the solution matrix obtained from relaxation Algorithm 3. We can cast each row of V as a point in $\Re^{k-1}$, and thus we obtain a data set of n points in $\Re^{k-1}$, i.e., $\mathcal{V} = \{v_1, \cdots, v_n\}$. Then we perform the classical K-means clustering for the data set $\mathcal{V}$. From Theorem 3.1.2, this equals to solving the following 0-1 SDP problem

$$\begin{aligned} \min \quad & \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z})\sum_{\mathrm{i=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}} \mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{4.7}$$

For constrained K-means clustering, then we need to solve the following subproblem

$$\begin{aligned} \min \quad & \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z})\sum_{\mathrm{i=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}} \mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) \\ & Z_{ii} \leq \frac{1}{\tilde{n}} \quad i = 1, \cdots, n, \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{4.8}$$

Finally, we partition all the entities in the original space based on the clustering on $\mathcal{V}$, i.e., the entities s_i, s_j belong to the same cluster if and only if v_i, v_j are in the same cluster.

The whole algorithm is described in Algorithm 4.

```
Algorithm 4 Approximation Algorithm
  Step 1: Calculate the projection of the matrix $W$ onto the null space of $e$, i.e.,
      $W_1 = (I - \frac{1}{n}ee^T)W(I - \frac{1}{n}ee^T)$;
  Step 2: Use singular value decomposition method to compute the first $k-1$ largest eigenvalues of the matrix $W_1$ and their corresponding eigenvectors $v^1, \cdots, v^{k-1}$;
  Step 3: Solve problem (4.7) (or (4.8)) for (constrained) K-means clustering;
  Step 4: Assign all the entities in $\mathcal{S}$ based on the assignment obtained from step 3.
```

4.3 Estimation of the Approximate Solution

In this section, we estimate the approximation solution provided by our algorithm. We first consider the case for the classical K-means clustering. It should be pointed out that in [17], Drineas et'al considered a similar algorithm based on the singular value decomposition of W and showed that their algorithm can provide a 2-approximation to the original K-means clustering. However, since the working subspace in our algorithm is quite different from what in [17], a new analysis is necessary to investigate the approximation ratio of the solution obtained from Algorithm 4.

We now discuss the case of bi-clustering. One reason for this is that for bi-clustering, the subproblem involved in Step 3 of Algorithm 4 is in $\Re$. In such a case, the task in Step 3 of Algorithm 4 reduces to partitioning the data set $\mathcal{V} = \{v_i \in \Re, i = 1, \cdots, n\}$ into two clusters based on the MSSC model. Therefore, we can refer to the Refined K-means clustering in one dimension (Algorithm 5) .

Algorithm 5 is similar to the algorithm in [23] for divisive k-clustering in low dimension. The idea is that a minimum sum of squares bipartition of a set maximizes the product of the squared distance between the centroids of C_1 and $\bar{C}$, the center of the whole data set, and the ratio of the cardinalities of C_1 and C_2, namely $N_1\|\bar{v_1} - \bar{v}\|^2/N_2$, and this function is actually unimodal, the detail proof of this theorem can be found in [23]. It is straightforward to see that for bi-clustering problems in $\Re$ based on the MSSC model, the above procedure can find a global solution in $O(n \log n)$ time.

If $k \geq 3$, then we can resort to the algorithm in [17] to solve problem (4.7). It is easy to see that the algorithm takes $O(n^{k^2(k-1)/2})$ time to find the global solution of the subproblem in Step 3 of Algorithm 4, which is roughly a $\frac{1}{n^{k^2/2}}$ fraction of the running time when the same procedure is applied to solve the subproblem in [17]. This is because the working space in our algorithm is one dimension less than the space in [17]. In case of bi-clustering, the improvement is substantial since we can use our simple refined K-means

Algorithm 5 Refined K-means in One Dimension

```
Step 0: Input the data set $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$;
Step 1: Calculate the center $\bar{v} = \sum_{v_i \in \mathcal{V}} v_i/N$, where $N$ is the cardinality of $\mathcal{V}$ and sort the sequence so that $v_{i_1} \geq v_{i_2} \cdots \geq v_{i_n}$, here $\{i_1, \cdots, i_n\}$ is a permutation of the index set $\{1, \cdots, n\}$;
Step 2:
for $l = 1$ to $n$ do
    calculate the center of the first part of the partition, $C_1^l = \{v_{i_1}, \cdots, v_{i_l}\}$
        $$\bar{v}_l = (l-1)\bar{v}_{l-1} + v_{i_l},$$
    and evaluate the function
        $$f(l) = N_1(\bar{v}_l - \bar{v})^2/(N - N1).$$
    if $f(l) < f(l-1)$ then
        output $C_1 = \{v_{i_1}, \cdots, v_{i_{l-1}}\}, C_2 = \{v_{i_l}, \cdots, v_{i_n}\}$ as the final solution
    end if
end for
```

in one dimension.

We next progress to estimate the solution obtained from Algorithm 4. Let Z^* be a global solution to (3.6) and $\bar{Z}$ is the solution provided by Algorithm 2. Let us define

$$U = \frac{1}{n}ee^T + \sum_{i=1}^{k-1} v^i(v^i)^T. \tag{4.9}$$

It follows

$$\mathrm{Tr}\left((\mathrm{I}-\mathrm{U})\sum_{\mathrm{i}=1}^{\mathrm{k}-1} \mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) = 0; \tag{4.10}$$

$$\mathrm{Tr}\left(\mathrm{U}\sum_{\mathrm{i}=\mathrm{k}}^{\mathrm{n}-1} \mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) = 0. \tag{4.11}$$

From Theorem 4.2.1, we have

$$\mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)) \geq \mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{U})). \tag{4.12}$$

It follows

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\bar{\mathrm{Z}})\big) = \mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\mathrm{U}+\mathrm{U}-\bar{\mathrm{Z}})\big) \leq \mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)) + \mathrm{Tr}\big(\mathrm{W}(\mathrm{U}-\bar{\mathrm{Z}})\big).$$

The above relation implies that if

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{U}-\bar{\mathrm{Z}})\big) \leq \mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)), \tag{4.13}$$

then

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\bar{\mathrm{Z}})\big) \leq 2\mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)),$$

i.e., in the worst case, the solution provided by Algorithm 4 is a 2-approximation to the original K-means clustering.

In what follows we prove (4.13), which can be equivalently stated as

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\big) \geq 0. \tag{4.14}$$

By the choices of $Z^*, \bar{Z}$ and U, it is easy to verify

$$(I - Z^* + \bar{Z} - U)e = 0, \tag{4.15}$$

$$(I - \frac{ee^T}{n})(I - Z^* + \bar{Z} - U) = (I - \frac{ee^T}{n})(I - Z^* + \bar{Z} - U)(I - \frac{ee^T}{n}). \tag{4.16}$$

It follows immediately that

$$\begin{aligned}
\mathrm{Tr}(W(I - Z^* + \bar{Z} - U)) &= \frac{1}{n}\mathrm{Tr}(W(I - Z^* + \bar{Z} - U)ee^T) + \mathrm{Tr}(W_1(I - Z^* + \bar{Z} - U)) \\
&= \mathrm{Tr}\left((I - Z^* + \bar{Z} - U)\sum_{i=1}^{n-1}\lambda_i v^i (v^i)^T\right) \\
&= \mathrm{Tr}\left((I - Z^* + \bar{Z} - U)\sum_{i=1}^{k-1}\lambda_i v^i (v^i)^T)\right) \\
&\quad + \mathrm{Tr}\left((I - Z^* + \bar{Z} - U)\sum_{i=k}^{n-1}\lambda_i v^i (v^i)^T\right) \\
&= \mathrm{Tr}\left((\bar{Z} - Z^*)\sum_{i=1}^{k-1}\lambda_i v^i (v^i)^T)\right) + \mathrm{Tr}\left((I - Z^* + \bar{Z})\sum_{i=k}^{n-1}\lambda_i v^i (v^i)^T\right) \\
&\geq \mathrm{Tr}\left((\bar{Z} - Z^*)\sum_{i=1}^{k-1}\lambda_i v^i (v^i)^T)\right),
\end{aligned}$$

where the last equality is given by (4.10) and (4.11), and the last inequality is implied by the fact that the matrix $I - Z^* + \bar{Z}$ is positive semidefinite. Recall that $\bar{Z}$ is the global solution of subproblem (4.7) and Z^* is only a feasible solution of (4.7), we therefore have

$$\mathrm{Tr}\left((\bar{Z} - Z^*)\sum_{i=1}^{k-1}\lambda_i v^i (v^i)^T)\right) \geq 0,$$

which further implies (4.13).

Now we are ready to state the main result in this chapter, which follows immediately from (4.12) and (4.13).

Theorem 4.3.1. *Suppose that Z^* is a global solution to problem (3.6) and $\bar{Z}$ is the solution provided by Algorithm 4. Then, we have*

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\bar{\mathrm{Z}})\big) \leq 2\mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)).$$

In what follows we estimate the approximation rate of the solution provided by Algorithm 4 for constrained K-means clustering. It worth mentioning that in such a case, no polynomial algorithm has been reported in the literature to find a global solution of subproblem (4.8). However, suppose a global solution to (4.8) can be located, then by following a similar argument as in the proof of Theorem 4.3.1, we can prove the following result.

Theorem 4.3.2. *Suppose that Z^* is a global solution to problem (3.10) and $\bar{Z}$ is the solution provided by Algorithm 4. Then, we have*

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\bar{\mathrm{Z}})\big) \leq 2\mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*)).$$

Chapter 5

Balanced Bi-clustering

From this chapter, we start discussing the second part of this thesis, first, we consider the so-called balanced bi-clustering problem, an improved algorithm based on existing ones and an algorithm with an approximation bound are proposed.

5.1 Motivation

The main problem we consider here is bi-clustering, bi-clustering involves with partitioning the data set into two clusters. It is one of the fundamental issues in clustering. This is particularly true in the so-called hierarchical divisive clustering where at each step, a bi-clustering is performed. Due to its importance, bi-clustering has drawn much attention from various researchers, especially from the expertise in computational geometric community [2; 22; 27; 34].

Although we can embed balanced clustering in our model (4.8), we require every cluster shares the same lower bound. There might be situations that we require different clusters satisfied different lower bounds. In this case, we need to explore new avenue to solve the the balanced bi-clustering problem.

Motivated by the above requirements, in the second part of this thesis, we consider the balanced bi-clustering and its special case, where the size of each resultant clusters is fixed. From a mathematical viewpoint, balanced clustering can be casted as a constrained optimization problem. We note that in [8], Bradley *et al* proposed to use linear optimization technique to solve the subproblem in their model. In our work, we use the same theoretical framework as that in [8]. However, by restricting us to the balanced bi-clustering, we propose a local search method that can be used to find the optimal solution of the subproblem in the model. We show that our simple heuristics enjoys a lower complexity than the approach suggested in [8].

Since the problem (1.1) is NP-hard and many algorithms like K-means can only locate a local minimum, the issue of how to find a good approximate solution to (1.1) has caught the attention of many experts in the field. An early remarkable work in this direction is due to Hasegawa et'al who showed that by using the so-called K-Medoids method [22; 27], we can obtain a 2-

approximation solution to problem (1.1). Based on a similar idea as in [22], we propose an algorithm that can provide a 2-approximation solution to balanced bi-clustering.

5.2 Definition of Problem

The problem we consider is to partition a set of n entities into the space to two clusters based on the MSSC model, subject to some constraints on the cardinality of each cluster. More rigorously, given a data set $\mathcal{S}$ of n entities in an Euclidean space, our task is to find a partition (C_1, C_2) of the data set $\mathcal{S}$ such that:

$$\begin{aligned} \min_{C_1,C_2} \quad & \textstyle\sum_{s_i \in C_1} \|s_i - c_1\|^2 + \sum_{s_i \in C_2} \|s_i - c_2\|^2 \\ s.t. \quad & |C_1| \geq \tau_1, \quad |C_2| \geq \tau_2 \quad (\tau_1 + \tau_2 \leq n) \end{aligned} \tag{5.1}$$

here (c_1, c_2) is the geometry center of (C_1, C_2). It can be easily proved that, without the constraints, (5.1) is equivalent to the model that we originally defined for bi-clustering:

$$\min_{c_1,c_2} \sum_{i=1}^{n} \min\{\|s_i - c_1\|^2, \|s_i - c_2\|^2\} \tag{5.2}$$

For the unconstrained bi-clustering problem (5.2), J. Matousek [34] proposed a $(1+\varepsilon)$-approximately optimal algorithm in time $O(\log n + \varepsilon^{-2d(d-1)})$

with a preprocess the set $\mathcal{S}$ in $O(n \log n)$ time, where d is the dimensional number of the space where the entities belong to. However, we can not utilize these results directly due to the constraints we have. On the other hand, the algorithm in [34] itself is very difficult to use in practice. This inspires us to design efficient heuristics to tackle the constrained problem with an appropriate approximate rate.

5.3 Brute-force Algorithm

In Algorithm 6, we describe a brute-force algorithm that can find an optimal of the constrained problem.

Algorithm 6 can solve the constrained problem exactly. Unfortunately, the running time of the algorithm is $O(n^{\tau_1+1})$, which means in the extreme case $\tau_1 = n/2$, complexity might be as high as $O(n^{n/2+1})$. This is definitely impractical.

Algorithm 6 Brute-force Algorithm

```
$t = \infty, C_1^* = \emptyset, C_2^* = \mathcal{S}$
for each subset $C_1$ of $\mathcal{S}$ with size $\tau_1 \le |C_1| \le (n - \tau_2)$ do
    $C_2 = \mathcal{S} - C_1$;
    Compute $c_1$, the center of $C_1$, and $c_2$, the center of $C_2$;
    Calculate the sum-of-squared error
        $f = \sum_{s_i \in C_1} \|s_i - c_1\|^2 + \sum_{s_j \in C_2} \|s_j - c_2\|^2$;
    if $f < t$ then
        $t = f$;
        Replace the existing optimal partition $(C_1^*, C_2^*)$ by the current partition $(C_1, C_2)$;
    end if
end for
Output the bipartition $(C_1^*, C_2^*)$ as a solution;
```

5.4 Improved Constrained K-means Algorithm for Bi-clustering

To solve (5.1), we can also use the constrained K-means algorithm proposed in [8]. For self-completeness, we will describe the so-called constrained K-means algorithm here briefly.

Following the same argument as in chapter 1, (5.1) is equivalent to the problem below, where x_{ij} is the assignment matrix.

$$\begin{aligned}
\min_{x_{ij}} \quad & \textstyle\sum_{j=1}^{2} \sum_{i=1}^{n} x_{ij} \left\| s_i - \frac{\sum_{l=1}^{n} x_{lj} s_l}{\sum_{l=1}^{n} x_{lj}} \right\|^2 \\
S.T. \quad & \textstyle\sum_{j=1}^{2} x_{ij} = 1 \quad (i = 1, ..., n) \\
& \textstyle\sum_{i=1}^{n} x_{ij} \geq \tau_j \quad (j = 1, 2) \\
& x_{ij} \geq 0 \quad (i = 1, \cdots, n; j = 1, 2)
\end{aligned} \tag{5.3}$$

An iterative algorithm, which is similar to the classic K-means approach, is proposed in [8] to solve the problem (5.3), the algorithm is described in Algorithm 7:

According to Proposition 2.3 and Proposition 3.1 in [8], Algorithm 7 will eventually terminate in a finite number of iterations at a cluster assignment that is locally optimal such that $x_{ij} \in \{0, 1\}$ The time complexity of this algorithm depends on the algorithm that is chosen for the subproblem (5.4). In [8], fast network simplex algorithms are used for solving the sub-problem, the running time for the fast network simplex algorithm is at least $O(n \log^2 n)$ according to [1].

In the sequel, we propose a simple heuristic for the balanced bi-clustering problem. Our algorithm is easy to implemented and has better complexity.

Algorithm 7 Constrained K-means Algorithm

Given cluster centers c_1^t and c_2^t at iteration t, compute c_1^{t+1} and c_2^{t+1} at iteration $t+1$ in following 2 steps:

Step 1. **Cluster Assignment.**

Let x_{ij}^t be a solution to the following linear program with c_1^t and c_2^t fixed:

$$\begin{aligned} \min_{x_{ij}} \quad & \textstyle\sum_{j=1}^{2}\sum_{i=1}^{n} x_{ij} \left\|s_i - c_j^t\right\|^2 \\ S.T. \quad & \textstyle\sum_{j=1}^{2} x_{ij} = 1 \quad (i = 1, ..., n) \\ & \textstyle\sum_{i=1}^{n} x_{ij} \geq \tau_j \quad (j = 1, 2) \\ & x_{ij} \geq 0 \quad (i = 1, \dot{s}, n; j = 1, 2) \end{aligned} \tag{5.4}$$

Step 2. **Cluster Update.**

Update C_1^t and C_2^t as follows:

$$c_j^{t+1} = \begin{cases} \frac{\sum_{l=1}^{n} x_{lj} s_l}{\sum_{l=1}^{n} x_{lj}} & \text{If } \sum_{i=1}^{n} x_{i,j}^t > 0, \\ c_j^t & \text{Otherwise.} \end{cases} \tag{5.5}$$

Stop when $c_j^{t+1} = c_j^t$, $j = 1, 2$, otherwise increment t by 1 and go to step 1.

Given c_1^t and c_2^t, a partition of the date set, C_1^t and C_2^t, is derived. If C_1^t and C_2^t satisfy the constraints that we have in (5.4), i.e. if $|C_1^t| \geq \tau_1, |C_2^t| \geq \tau_2$, it is right a solution for the subproblem (5.4); Otherwise, we can use the following rounding procedure to extract a solution that satisfies the constraints in (5.4).

Without loss of gentility, suppose we need to move some entities from C_1 to C_2:

Rounding Procedure 1

```
for every $s_i \in C_1$ do
    Compute the change of objective function by moving $s_i$ to $C_2$, i.e. Calculate a function $f(s_i) = \|s_i - c_2\| - \|s_i - c_1\|$ for $s_i$;
end for
Sort all the entities based on $f(s_i)$, move $(|C_1| - \tau_1)$ entities which have the least $f(s_i)$ to $C_2$ to make the constraints satisfied.
```

This procedure is essentially a greedy algorithm, the time complexity of this procedure is $O(n \log n)$. It provides us a partitioning that satisfies the constraints in (5.4). We can claim such a partitioning is an optimal solution to (5.4). To see this, suppose to the contrary that the solution provided by the rounding procedure is not optimal, then, we can reduce the objective function in (5.4) by moving some entities from C_2 to C_1, or swapping two entities in C_1 and C_2. This is definitely impossible as either moving one entity from C_2 to C_1 or swapping two entities in C_1 and C_2 will lead to an increase of the objective function.

Therefore, by using the rounding procedure 1, we have the improved constrained K-means algorithm as described in Algorithm 8.

Algorithm 8 Improved Constrained K-means Algorithm

Given cluster centers c_1^t and c_2^t at iteration t, compute c_1^{t+1} and c_2^{t+1} at iteration $t+1$ via the following 2 steps:

Step 1. **Cluster Assignment.**

Assign all the entities to clusters C_1^t and C_2^t based on their distances to c_1^t and c_2^t respectively; if such a partition is infeasible for (5.4) , then use the rounding procedure 1 to find the optimal solution of problem 5.4.

Step 2. **Cluster Update.**

Update the cluster centers by

$$c_j^{t+1} = \begin{cases} \frac{1}{|C_j^t|}\sum_{s_i \in C_j^t} s_i & \text{If } |C_j^t| > 0 \\ c_j^t & \text{Otherwise.} \end{cases} \tag{5.6}$$

Stop if $c_j^{t+1} = c_j^t$, $j = 1, 2$; Otherwise increment t by 1 and go to step 1.

The complexity of Algorithm 8 is $O(Tn\log n)$, where T is the times that we run this algorithm. After it stops, Algorithm 8 will end up with a local optimum solution, if the solution is not derived from the rounding procedure.

In what follows, we introduce a a 'dynamic' rounding procedure, which is more precise than the rounding procedure 1. Note that if we move an entity s_i from C_1 to C_2, the centroids of these two clusters after the movement can be easily calculated by [51]

$$c_1 \leftarrow \frac{|C_1|c_1 - s_i}{|C_1| - 1}; \quad c_2 \leftarrow \frac{|C_2|c_2 + s_i}{|C_2| + 1}.$$

Correspondingly, the change of the objective function value cased by this move is

$$v(s_i) = \frac{|C_2|}{|C_2| + 1} \|c_2 - s_i\|^2 - \frac{|C_1|}{|C_1| - 1} \|c_1 - s_i\|^2, \quad s_i \in C_1.$$

This provides a way to examine how the objective function changes if we move one entity in the data set from one cluster to the other.

In this new rounding procedure, we update the centroids of the two clusters immediately after a reassignment occurs. Without loss of gentility, suppose we need to move Δ points from C_1 to C_2 to satisfy the balanced constraints:

```
Rounding Procedure 2
repeat
    For every $s_i \in C_1$, calculate $v(s_i)$ i.e. the change of objective function by
    moving $s_i$ to the $C_2$;
    Move the point with $\min\{v(s_i)\}$ to $C_2$, then update the centroids of two
    clusters according to the formula introduced above;
until $\Delta$ points have been moved from $C_1$ to $C_2$
```

The complexity of this 'dynamic' procedure is $O(n^2)$, which leads to $O(Tn^2)$ of algorithm 8 if rounding procedure 2 is used. We also mention that the rounding procedure 2 could also be applied after algorithm 8 stops. It will consequently serve as a local search procedure, where we run the procedure to reduce the objective function further until $\min\{v(s_i)\} > 0$ or any movement will violate the balanced constraints. This two-phase heuristic is similar to HK-means [25] for unconstrained clustering.

5.5 An 2-Approximation Algorithm

Though efficient, unfortunately, Algorithm 8 can not provide a global optimum solution of the underlying problem even with the local search procedure following. In fact, this is one main disadvantage of all K-means type heuristics. It has been observed that K-means could converges to a local minimum that is arbitrary bad compared to the true global optimal solution. Such an example is shown in Figure 5.1:

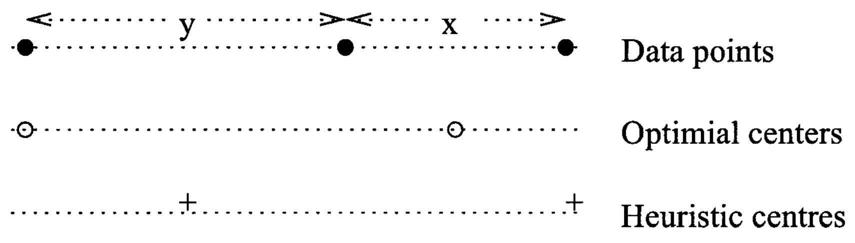

Figure 5.1: K-means can produce arbitrarily bad solution

For $k = 2$ and where $y/2 < x < y$. The optimal sum-of-squared error is $x^2/4$, but it is easy to verify that the solution shown at the bottom is centroidal and has a sum-of-squared error of $y^2/4$. By increasing the ratio y/x the approximation rate for K-means algorithm can be made arbitrarily high.

To find a global solution for the MSSC model, many heuristics for k-clustering, based on methods such as branch-and-bound searching, gradient decent, simulated annealing and genetic algorithms have been proposed and studied by various researchers [6; 16; 54]. However, No proven approximation bounds are known for these methods. In particular, Hasegawa et'al [22] proposed to use the so-called K-Medoids method to solve problem (1.1) and showed that their method can provide a 2-approximation solution to problem (1.1). In the sequel, we will consider a similar algorithm for the balanced case.

First, we introduce a stronger constrained bi-clustering problem as follows. Given a data set $\mathcal{S}$ of n entities, we consider the following partitioning problem:

$$\begin{aligned} \min_{C_1,C_2,c_1\in C_1,c_2\in C_2} \quad & \textstyle\sum_{s_i\in C_1} \|s_i - c_1\|^2 + \sum_{s_i\in C_2} \|s_i - c_2\|^2 \\ S.T. \quad & |C_1| \geq \tau_1, \quad |C_2| \geq \tau_2 \quad (\tau_1+\tau_2 \leq n) \end{aligned} \tag{5.7}$$

In other words, we impose the requirement that the centroids of these two clusters must be chosen from entities in the corresponding clusters. We have

Lemma 5.5.1. *The constrained bi-clustering problem (5.7) for n entities in any fixed dimension d can be solved in $O(n^3 \log n)$ time.*

Proof. We can enumerate all pair of entities in the data set as the (c_1, c_2) in (5.7). For every fixed pair of (c_1, c_2), we can use the step 1 in Algorithm 3 to find a solution for problem (5.7). Since every run of such a procedure takes $O(n \log n)$ time and in total we have $n(n-1)/2$ pairs, the constrained problem (5.7) can be solved in $O(n^3 \log n)$ time.

We next present a technical result that will be used in our later analysis.

Lemma 5.5.2. *Given a data set $\mathcal{S} = \{s_1, \cdots, s_n\}$ with $s_i \in \Re^d, i = 1, \cdots, n$, centered at its geometric centroid $c = \frac{1}{|\mathcal{S}|}\sum_{s_i \in \mathcal{S}} s_i$. If a point $s \in \Re^d$ satisfies*

$$\|s - c\| \le \|s_i - c\|, \quad \forall i = 1, \cdots, n,$$

then we have

$$\sum_{s_i \in \mathcal{S}} \|s_i - s\|^2 \le 2 \sum_{s_i \in \mathcal{S}} \|s_i - c\|^2.$$

Proof. It is straightforward to see that

$$\begin{aligned}
\sum_{s_i \in \mathcal{S}} \|s_i - s\|^2 \quad &= \sum_{s_i \in \mathcal{S}} \|s_i - c + c - s\|^2 \\
&= \sum_{s_i \in \mathcal{S}} \|s_i - c\|^2 + |\mathcal{S}|\|c - s\|^2 + \sum_{s_i \in \mathcal{S}} (s_i - c)^T (c - s) \\
&= \sum_{s_i \in \mathcal{S}} \|s_i - c\|^2 + |\mathcal{S}|\|c - s\|^2 \\
&\le 2 \sum_{s_i \in \mathcal{S}} \|s_i - c\|^2,
\end{aligned}$$

where the last inequality follows from the assumption in the lemma. This finishes the proof of the lemma.

The next lemma leads to the Algorithm 9, which will be proposed later, and the approximation rate of the solution provided by Algorithm 9.

Lemma 5.5.3. *For the constrained problem (5.7), the sum-of-squared error of optimum solution is at most twice the sum-of-squared error of optimum solution of problem (5.1).*

Proof. Let $\{(c_1, C_1), (c_2, C_2)\}$ be an optimum solution of (5.1). For such a fixed partition (C_1, C_2), we consider the following constrained minimization problems

$$\min_{s \in C_i} \|s - c_i\| \quad i = 1, 2. \tag{5.8}$$

Denote the solutions of the above two problems by c'_1 and c'_2 respectively. From Lemma 5.5.2, we obtain

$$\sum_{s \in C_1} \|c'_1 - s\|^2 + \sum_{s \in C_2} \|c'_2 - s\|^2 \leq 2 \left(\sum_{s \in C_1} \|c_1 - s\|^2 + \sum_{s \in C_2} \|c_2 - s\|^2 \right) \tag{5.9}$$

Recall the fact that the partition (C_1, C_2) satisfy the constraints in problem (5.1) and thus (c'_1, C_1, c'_2, C_2) is also a feasible solution of problem (5.7). This implies that the sum-of-squared error of the optimal solution to problem (5.7) is less than or equal to the sum-of-squared error of the optimal solution of problem (5.8), which further concludes the lemma.

Based on the above analysis, we propose the following algorithm.

Algorithm 9 2-Approximation Algorithm

```
for every pair of $(c_1, c_2)$ where $c_1, c_2 \in \mathcal{S}$ do
    Use $(c_1, c_2)$ as the starting centers and apply Algorithm 2 to solve problem (5.1).
end for
Output the bipartition with smallest sum-of-squares error as a solution.
```

Combining Lemma 5.5.1 and Lemma 5.5.3, we have

Theorem 5.5.4. *For a given set of n entities, Algorithm 9 can provides a 2-approximate solution to problem (5.1), and the time complexity of the algorithm is $O(Tn^3 \log n)$, where T is the maximal number of iterations in running Algorithm 2.*

Chapter 6

An Approximate Method to Bi-clustering with Fixed Size

In this chapter, we consider the special case of balance bi-clustering where the sizes of cluster are fixed, we propose a new method (called Q-means), which could not only give a 2-approximation bound theoretically, but also enjoys a lower iteration bound than our algorithm for general balanced bi-clustering.

6.1 Definition of Problem

In this section, we consider a special case of the balanced bi-clustering, called fixed size bi-clustering. Mathematically, this problem can be described as:

$$\begin{array}{ll} \min\limits_{C_1,C_2} & \sum_{s_i \in C_1} \|s_i - c_1\|^2 + \sum_{s_i \in C_2} \|s_i - c_2\|^2 \\ S.T. & \frac{|C_1|}{|C_2|} = R \quad (R \geq 1) \end{array} \tag{6.1}$$

Here (c_1, c_2) is the geometry center of (C_1, C_2).

A straightforward algorithm to solve problem (6.1) is to execute the rounding procedure introduced in last chapter after the classic K-means algorithm stops to obtain a solution that satisfies the additional constraint. In such an approach, the information from the constraint is not used. The algorithm simply performs an unguided search and tries to locate a feasible solution at last passively. We can also try all the pairs in the data set as starting centers, and use an algorithm similar to Algorithms 8 and 9 to solve problem (6.1). Following the results from last chapter, such an algorithm will have a complexity $O(n^3 \log n)$. In the following section, we proposed a more efficient algorithm, called Q-means to solve problem (6.1). Our new method uses the constraint in problem (6.1) to select the starting centers as required in Algorithm 8 and 9. As we shall see later, such a heuristics will improve the efficiency of the algorithm substantially.

6.2 Q-means Algorithm

For a given data set $\mathcal{S}$, let us denote its centroid by $\bar{c}$, i.e., $\bar{c} = \frac{1}{n}\sum_{i=1}^{n} s_i$. Suppose that we partition the set $\mathcal{S}$ into two clusters C_1, C_2, where c_1 and c_2 are the two cluster centroids respectively. Without loss of generality, we assume $|C_1| \geq |C_2| > 0$. It is easy to verify the following relation

$$(1-t)c_1 + tc_2 = \bar{c} \text{ for some } t \in (0, \tfrac{1}{2}].$$

which can be equivalently stated as:

$$c_2 = c_1 + \frac{1}{t}(\bar{c} - c_1). \tag{6.2}$$

Therefore, we have:

$$\begin{aligned} \|s_i - c_2\|^2 &= \|s_i - c_1 - \frac{1}{t}(\bar{c} - c_1)\|^2 \\ &= \|s_i - c_1\|^2 + \frac{1}{t^2}\|\bar{c} - c_1\|^2 - \frac{2}{t}(s_i - c_1)^T(\bar{c} - c_1) \end{aligned} \tag{6.3}$$

For every entity s_i, let $Q = 1/t$ and define

$$\phi(c_1, s_i) = Q^2\|\bar{c} - c_1\|^2 - 2Q(s_i - c_1)^T(\bar{c} - c_1)$$

Then, we have

$$\|s_i - c_2\|^2 = \|s_i - c_1\|^2 + \phi(c_1, s_i).$$

Using the above notation, we can rewrite the MSSC model as the following bilevel optimization problem

$$\min_{c_1, Q} \sum_{i=1}^{n} (\|s_i - c_1\|^2 + \min\{0, \phi(c_1, s_i)\}).$$

For any given cluster center c_1 and fixed Q, we can determine the cluster that each entity belongs to by the property function $\phi(c_1, s_i)$. If $\phi(c_1, s_i) \geq 0$, then $s_i \in C_1$; otherwise, $s_i \in C_2$. Let us define the active index set $I = i : \phi(c_1, s_i) < 0$. Then the MSSC model can be transformed into a quadratic programming(QP) problem:

$$\min_{c_1} \sum_{i=1}^{n} \|s_i - c_1\|^2 + \sum_{i \in I} \phi(c_1, s_i). \tag{6.4}$$

From our above discussion, it becomes clear that solving problem (6.4) equals to assigning entities to two clusters whose centroids are c_1 and $c_1 + Q(\bar{c} - c_1)$ respectively. On the other hand, given two geometric centers (c_1, c_2) of cluster (C_1, C_2), one has

$$\bar{c} = \frac{|C_1|}{n} c_1 + \frac{|C_2|}{n} c_2.$$

For our problem (6.1), since $\frac{|C_1|}{|C_2|} = R$, we have:

$$Q = R + 1.$$

Therefore, when R is fixed, the geometric centers of the two clusters (c_1, c_2) together with $\bar{c}$ have the relation (6.2) with $Q = R + 1$. We mention that the converse statement is not true in general, i.e., Q can not determine precisely the size ratio of two clusters. However, we still expect that Q can give us some guidelines to control the relative cardinality of the two clusters. This inspires us to consider the so-called Q-means algorithm described in Algorithm 10:

Algorithm 10 Q-means Algorithm

```
Given size ratio requirement $R$ ($R \geq 1$);
Step 1: Initialization
    Choose an entity in the space as the starting center of cluster $C_1$, say $c_1$.
    Set $Q = R + 1, i = 1, c_1^i = c_1$.
Step 2: Iteration
    Identify the active set $I = \{i : \phi(c_1^i, s_i) \leq 0\}$;
    Find $c_1^{i+1}$ the solution of the quadratic problem (6.4) based on the current active index set $I$. Goto Step 2 if $c_1^i \neq c_1^{i+1}$;
Step 3: Rounding
    Use the rounding procedure 1 or 2 in Section 2.3 to find a solution that satisfies the size ratio constraint.
Step 4: Alternation
    Set $Q = 1 - 1/(R + 1)$, repeat step 2 and step 3.
Step 5: Choose the better one from the two solutions obtained from these two procedures as the final output.
```

In principle, Q-means is a heuristic similar to K-means. We iteratively reduce the objective function until convergence and then use the rounding procedure to find a solution that satisfies the constraint. Similar to K-means, Q-means is also very sensitive to the choice of initial starting entity and also very easy to be trapped in some local optimum after a few iterations. The complexity of Algorithm 10 is $O(Tnlogn)$ if the rounding procedure 1 is applied, where T is the maximal number of iterations in running Algorithm 10.

Alternatively, if we use rounding procedure 2, it will suffer a complexity of $O(Tn^2)$, but enjoys a better solution in return, since we conduct an even more 'greedy' approach than in the rounding procedure 1. It depends on the problem scale to decide which rounding procedure should be used. We will illustrate this idea in the numerical result chapter.

On the other hand. to get a good approximation solution, we employ the idea described in last chapter. However, in the special case of (6.1), we just need to try every entity in the set as the starting center used in the Q-means algorithm and then compute another center via the relation (6.2). This reduces the complexity of the whole procedure from $O(n^2)$ to $O(n)$ of enumerating all possible starting entities for the bi-clustering problems.

Algorithm 11 Revised Q-means

```
Step 1: Try every entity in the data set as the starting center in Algorithm 10.
Step 2: Output the best bipartition $(C_1^*, C_2^*)$ as a solution.
```

The time complexity of this algorithm is $O(n^2 \log n)$ if using rounding procedure 1, $O(n^3)$ if using rounding procedure 2. In what follows we derive an upper bound for the solution produced by the revised Q-means algorithm.

Theorem 6.2.1. *The sum-of-squared error of the solution derived by Algorithm 11 is at most twice the sum-of-squared error of optimum solution of problem (6.1).*

Proof. Let $\{(c_1, C_1), (c_2, C_2)\}$ be the optimal solution of the problem (6.1). Let us denote the solutions of the problems (5.8) by c_1' and c_2' respectively. Define

$$\begin{aligned} c_1^* &:= \frac{n}{|C_2|}\bar{c} - Rc_1' = c_2 + R(c_1 - c_1'), \\ c_2^* &:= \frac{n}{|C_1|}\bar{c} - \frac{1}{R}c_2' = c_1 + \frac{1}{R}(c_2 - c_2'). \end{aligned}$$

It follows

$$\|c_1^* - c_2\| = R\|c_1 - c_1'\|,$$

$$\|c_2^* - c_1\| = \frac{1}{R}\|c_2 - c_2'\|.$$

From the above two relations, we must have either

$$\|c_1^* - c_2\| \leq \|c_2' - c_2\|, \tag{6.5}$$

or

$$\|c_2^* - c_1\| \leq \|c_1' - c_1\|. \tag{6.6}$$

Without loss of generality, we can assume the inequality (6.5) holds. By

Lemma 5.5.2, we have

$$SSE := \sum_{s_i \in C_1} \|s_i - c_1'\|^2 + \sum_{s_i \in C_2} \|s_i - c_1^*\|^2 \leq 2 \sum_{s_i \in C_1} \|s_i - c_1\|^2 + 2 \sum_{s_i \in C_2} \|s_i - c_2\|^2.$$

Recall the fact that the pair (c_1', c_1^*) was used in Algorithm 11 as the starting centers for two clusters. Therefore, the sum-of-square error provided by the final output from Algorithm 11 must be less than or equal to SSE. In a similar vain, we can derive the same conclusion when (6.6) holds. This finishes the proof of the theorem.

Chapter 7

Computational Results

In this chapter, we present some preliminary computation results for the algorithms we have discussed before, it consists of two parts, the first part is about the bi-clustering via PCA, which is proposed in chapter 4, the second part is about the Q-means algorithm.

7.1 Numerical Experiments for Bi-clustering via PCA

To test the performance for Algorithm 4 in the case of bi-clustering, we have done some preliminary numerical experiments on several data sets from the

UCI Machine Learning Repository[1] and internet newsgroups. All the experiments are done by using Matlab on a personal computer with a Pentium 4 1700 MHz Processor and a 256M memory. The power method is applied for calculating the largest eigenvalue and eigenvector for the matrix [19].

It should be mentioned that although the subproblem (4.7) can be solved by using the procedure in [17], the running time of the procedure is clearly too much for reasonably large data set. Due to this fact, in our experiments, we restrict us only to bi-clustering.

Data Sets from the UCI Machine Learning Repository

- **Soybean Data Set (small):** see also [37]. This data set has 47 instances and each instance has 35 normalized attributes.
- **The Späth's Postal Zones:** This data set is from [51] about the post zones in Bavaria. It has 89 entities with each having 3 attributes.
- **Spam E-mail Database:** created by M. Hopkins *et al*from Hewlett-Packard Labs. It has 4601 samples, 57 attributes. For purpose of clustering, we have removed the last boolean attribute which indicates whether the e-mail was consider spam or not.

In our experiments, we use a two-phase strategy. After we obtain the

[1]http://www.ics.uci.edu/~mlearn/MLRepository.html

partition of the data sets from Approximation Algorithms 4, we use the classical K-means to further improve the partition. In other words, we only use Algorithms 4 as a starting strategy for K-means clustering. In the following tables, we list the solutions from both phase 1 and phase 2.

Since, for all the first two data sets, the global optimum has already been reported in [44] by using a linear programming model in the case of $k = 2$, we list it in the Global Opt. column as reference. The global solution for the third data set has been reported in [45].

Table 7.1: Results on three UCI data sets

data set	Stage 1	Stage 2	Global Opt.
Soybean	404.4593	404.4593	404.4593
The Späth's	$6.0255e + 11$	$6.0255e + 11$	$6.0255e + 11$
Spam E-mail	$9.43479784e + 08$	$9.43479784e + 08$	$9.43479784e + 08$

Numerical Results for Balanced Bi-Clustering

We also test our algorithm for balanced bi-clustering. To find a global solution to balanced bi-clustering, we adapt the LP model in chapter 4 slightly to incorporate balanced constraints. The solution obtained from the LP model gives us a lower bound for the global optimum of the balanced bi-clustering.

We also pointed out that for the third data set, its relatively large size prevents us from the use of the LP model due to the enormous amount $O(n^3)$ of constraints involved in the model. In such a case, since we know the optimum partition of the third data set does not comply with the balanced constraint, which means some rounding must be performed to satisfy the constraint, it will lead the final two clusters have a size ratio of 1:2, recall that the Q-means algorithm we proposed also works for fix size bi-clustering, therefore, the result from Q-means heuristic for the third data set is listed instead, the size ratio constraint for Q-means is 1:2.

In the experiments for the last two data sets, we require that each cluster has at least $n/3$ entities. For the soybean data set, we require each cluster has at least 22 entities. This is because the data set itself is fairly balanced already(the optimal bi-clustering has a $(20, 27)$ distribution). Table 7.2 summaries the results.

Table 7.2: Results for Balanced Bi-clustering

data set	Stage 1	Stage 2	LP/Q-means
Soybean	419.5473	418.5273	418.5273
The Späth's	$1.6310e+012$	$1.6310e+012$	$1.6310e+012$
Spam E-mail	$1.4049e+09$	$1.4046e+09$	$1.4046e+09$

From the above tables we can see that the solution from phase 1 is very close to the solution from phase 2. In all the case, the solution from phase 2 match the global solution of the underling problem.

Internet Newsgroups

Text mining has been popular in document analysis, search engine and knowledge discovery in large volume of text data. We have also performed experiments on newsgroups articles submitted to 20 newsgroups[2]. This data set has also been used in [15; 20; 63], where a similar framework as ours was used to solve the problem. The algorithm we use is still the two-phase heuristic which is introduced in last section.

This data set consists of about 20,000 articles (email messages) evenly distributed among the 20 newsgroups. We list the name of the newsgroups together with the associated group labels in Table 7.3.

Before constructing the word-document matrices, we perform the preprocessing by using the **bow** toolkit, a preprocessor similar to what employed in [15; 20; 63]. In particular, we use the tokenization option such that the UseNet headers are stripped, since the headers include the name of the correct

[2]The news group data together with the bow tookit for preprocessing can be downloaded from http://www-2.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html

Table 7.3: Newsgroups and Their Labels

NG1	alt.atheism	**NG11**	rec.sport.hockey
NG2	comp.graphics	**NG12**	sci.crypt
NG3	comp.os.ms-windows.misc	**NG13**	sci.electronics
NG4	comp.sys.ibm.pc.hardware	**NG14**	sci.med
NG5	comp.sys.mac.hardware	**NG15**	sci.space
NG6	comp.windows.x	**NG16**	soc.religion.christian
NG7	misc.forsale	**NG17**	talk.politics.guns
NG8	rec.autos	**NG18**	talk.politics.mideast
NG9	rec.motorcycles	**NG19**	talk.politics.misc
NG10	rec.sport.baseball	**NG20**	talk.religion.misc

newsgroup, and we also apply stemming [38]. Afterwards, we apply the standard tf.idf weighting scheme and normalized each document vector to have unit Euclidean length. Finally, we conduct feature selection where 500 words are selected according to the mutual information between words and documents in unsupervised manner.

In our experiment, we choose 50 random document vectors each from two newsgroups. Then we apply our approximation algorithm to the problem. The results are summarized in table 5.3. Note that, since the global optimum are not known for these data sets, again, we use the linear programming relaxation model proposed in chapter 4 to get an lower bound on the global optimum. More specifically, we implement the LP relaxation model (4.1) in chapter 4 using package CPLEX 7.1 with AMPL interface on an IBM RS-6000, by solving this LP problem, we can obtain a lower bound for the global optimum solution. Apparently, if the solution obtained from the LP relaxation equals to the solution provided by our two-phase heuristic, then it must be a global optimal solution of the original problem.

From the experiments, we can conclude that our deterministic two-phase heuristic performs very well on these data sets and it finds the global optimum for most of these data sets.

Table 7.4: Results on internet newsgroup data sets

data set	Stage 1	Stage 2	LP
NG1/NG2	92.6690	92.6630	92.6630
NG2/NG3	94.0377	94.0377	94.0377
NG8/NG9	93.7051	93.5380	93.4007
NG10/NG11	92.0785	92.0299	92.0299
NG1/NG15	91.9277	91.9011	91.9011
NG18/NG19	92.2275	92.2035	92.2035

7.2 Numerical Experiments for Q-means

In this section, we present some preliminary computational results to illustrate the performance of our proposed Q-means algorithm, compared with the naive 'rounding K-means algorithm'. The performance of two rounding procedures is mentioned as well. The data sets that we used in the experiments include a synthetic data set and some well-known benchmarks in machine learning literature.

For fairness, we use the same starting strategy for the two algorithms, which means for K-means, we enumerate all the entities as one of the centroid and calculate the other one according to the size ratio constraint, just like what we do in Q-means. It follows that K-means will run deterministically. Also, the same rounding procedure in both K-means and Q-means. The running time and solution quality of two algorithms are compared.

The hardware environment is the same as we mentioned in last section, the CPU time is in seconds.

7.2.1 Synthetic Data Sets

We first use a random generator to produce various two-dimensional synthetic data sets approximately in the mixture Gaussian distribution [26], which are recognized as one of best test beds for MSSC clustering. Four data sets containing 4000 entities in [0,0] × [1,1] plane, are generated for our experiments, S01 with $v = 0.05, r = 0.0$, S02 with $v = 0.10, r = 0.2$, S03 with $v = 0.10, r = 0.4$, S04 with $v = 0.15, r = 0.4$. (v means variance and r stands for noise level.)

The Q-means algorithm and the naive "rounding K-means" algorithm are compared on above four data sets, whose distribution are represented by figures 7.1 and 7.2:

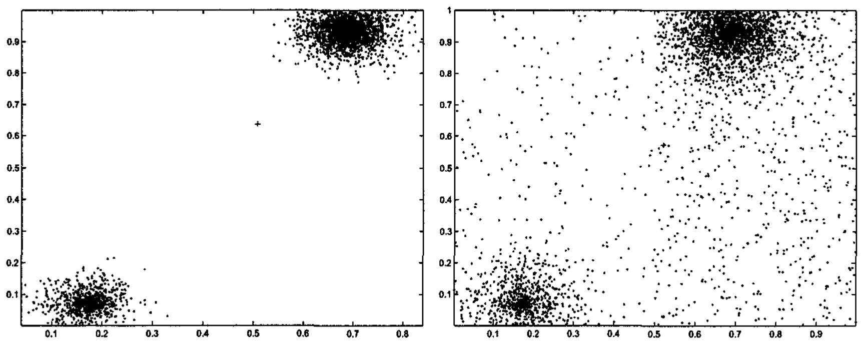

Figure 7.1: Synthetic Set (a) and (b)

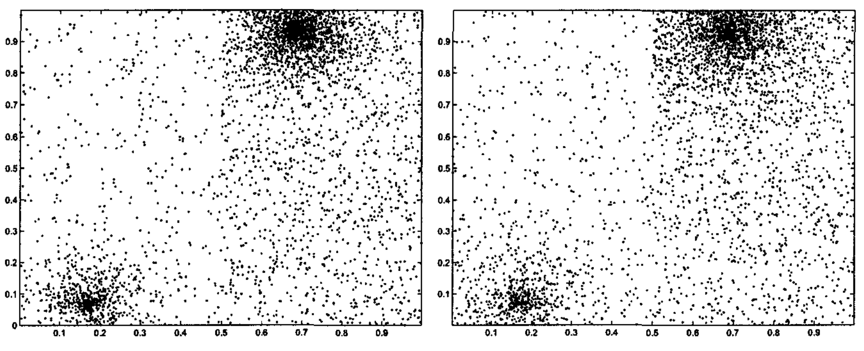

Figure 7.2: Synthetic Set (c) and (d)

The following result is obtained by using rounding procedure 1 in both algorithms, the size ratio is fixed as 1:3 for these four data sets.

Table 7.5: K-means vs. Q-means (Rounding Procedure 1)

data set	v	r	K-means (CPU time)	Q-means (CPU time)
S01	0.05	0.0	9.121028 (14.432218)	9.121028 (9.513855)
S02	0.10	0.2	165.422969 (61.143902)	165.422969 (33.010462)
S03	0.10	0.4	280.028264 (80.566430)	280.028264 (38.851260)
S04	0.15	0.4	281.542429 (109.471963)	281.044463 (42.279831)

The similar result, which is derived by using rounding procedure 2, is given below:

Table 7.6: K-means vs. Q-means (Rounding Procedure 2)

data set	v	r	K-means(CPU time)	Q-means(CPU time)
S01	0.05	0.0	9.121028 (13.834134)	9.121028 (8.171109)
S02	0.10	0.2	165.354702 (117.635497)	165.354702 (93.065645)
S03	0.10	0.4	278.773878 (287.669128)	278.769649 (242.697759)
S04	0.15	0.4	279.253263 (340.284469)	279.207675 (272.663927)

From the tests above, on the one hand, it turns out that Q-means outperforms 'rounding K-means' in terms of both running time and solution quality, regardless which rounding procedure is used. On the other hand, the rounding procedure 2, which has a higher complexity, could indeed find better solutions than rounding procedure 1, but takes much longer running time. Therefore, for small sale data sets, the rounding procedure 2 should be used to derive better solution without increasing the running time too much, whereas for large data set, rounding procedure 1 should be applied.

7.2.2 Data Sets in Literature

We have also tested the Q-means algorithm on the the following test problems in literature. The first two problems are small scale, therefore, the rounding procedure 2 is used. For the third data set, which is relatively large, we apply the rounding procedure 1 instead.

- **The Späth's Postal Zones Data Set**

  This data set is from [51] about the post zones in Bavaria. It has 89 entities with each having 3 characteristics. Given the same CPU time (indicated in the table) for the two algorithms, the result is summarized in the Table 7.7.

- **Soybean Data Set (large)**

  The second data set we use is the *Soybean data (large)* from the UCI

Table 7.7: The Späth's Postal Zones Data

size ratio	K-means (CPU time)	Q-means (CPU time)
1:1	1774654778161 (0.094566)	1774654778161 (0.019134)
1:2	1621271650670 (0.076535)	1621271650670 (0.018398)
1:3	1536771626734 (0.048473)	1536771626734 (0.017221)
1:4	1472023822662 (0.061756)	1472023822662 (0.017208)

Machine Learning Repository[3], see also [37]. This data set has 307 instances and each instance has 35 normalized attributes. The result is summarized in the Table 7.8.

Table 7.8: The Soybean Data (large)

size ratio	K-means (CPU time)	Q-means (CPU time)
1:1	4558.433495 (4.225763)	4558.433495 (1.594678)
1:2	4517.974430 (3.419375)	4517.974430 (1.409188)
1:3	4478.362896 (0.897566)	4478.362896 (0.326273)
1:4	4543.901639 (0.331163)	4543.901639 (0.085307)

From the above results, it can be seen that the solution of two algorithms are the same, the reason is that the scales of these two data sets are relatively small, it is easy for both algorithm to locate the global optimum

[3]http://www.ics.uci.edu/~mlearn/MLRepository.html

with rounding procedure 2. However, the Q-means algorithm still enjoys smaller running time. In the next experiment, we will test on a large data set. Notice that the rounding procedure 1 is used since, as the problem scale (number of entities and dimension) grows, the rounding procedure 2 has turned impractical.

- **Spam E-mail Database**

  The third data set we use is the *Spam E-mail Database*, which is created by M. Hopkins *et al.* It has 4601 samples, 57 attributes (we remove the last boolean attribute which indicates whether the e-mail was consider spam or not). We obtain the following result for this data set:

Table 7.9: Spam E-mail Database

size ratio	K-means (CPU time)	Q-means (CPU time)
1:1	1609292966.72 (2060.0481)	1586877787.41 (301.9538)
1:2	1405187739.99 (1807.5085)	1404622481.26 (334.4043)
1:3	1276474668.00 (1800.1751)	1276437654.01 (334.0490)
1:4	1276437654.01 (1565.5531)	1187542869.25 (311.2299)

It should point out that for the first and second experiments, since the rounding procedure 2 is applied, most CPU time is spent on the rounding process, therefore, the difference between K-means and Q-means is not that significant if rounding procedure 1 is applied. However, if only rounding procedure 1 is used, from experiment 3 we observe a substantial improvement.

From the above experiments we can conclude that, at least for these data sets, Q-means has outperformed regular K-means plus the rounding process in terms of both running time and solution quality. This is not surprising, as we pointed out earlier, besides reducing the complexity of enumerating all the entities, Q-means uses the constraint to guide the search process rather than to satisfy the constraint passively as in the rounding K-means.

Chapter 8

Conclusion and Future Works

For the first part in this thesis, we reformulated the classical MSSC as a 0-1 SDP. Our new model not only provides a unified framework for several existing clustering approaches, but also opens new avenues for clustering. An approximation method based on the SDP relaxation and PCA has been proposed to attack the underlying 0-1 SDP. It is shown that in the worst case, our method can provide a 2-approximate solution to the original classical or constrained K-means clustering. Preliminary numerical tests indicate that our two-phase algorithm can always find a global solution for bi-clustering.

For the second part, we proposed several algorithms to deal with the so-called balanced and fixed size bi-clustering problems. For the balanced bi-clustering, the improved constrained K-means algorithm in this paper provides a good alternative for the constrained K-means proposed in [8]. For the fix size case, the Q-means algorithm could extract a solution which theoretically has worst case approximation ratio at most two. Its theoretical properties are verified via our preliminary limited experiments. This is fair to conclude that Q-means is also good alternative for rounding K-means algorithm in practice, when fix size constraint is encountered.

There are several different ways to extend our results. First, for general $k \geq 3$, although subproblem (4.7) can be solved by using the procedure in [17], its complexity is still exponential in k. This makes the algorithm impractical for relatively large data set. Secondly, the current model can deal with only a simple case of constrained K-means clustering. The issue of how to deal with general constrained K-means clustering still remains open. It worths mentioning that, in this case, Q-means is a feasible choice if we know in advance that the optimum partition of the data set does not comply with the balanced constraints, since if this is true, the balanced clustering problem can be reduced to the fix-size constrained problem. Third, for the second part of our work, at present our algorithms can only deal with bi-clustering now. It will be interesting to investigate how our ideas can be extended to deal with balanced and fix size k-clustering with $k > 2$. It is also an interesting topic to study how the idea of Q-means could be extended to the balanced bi-clustering or even k-clustering cases, which could possibly lead to new techniques for the balanced clustering.

Bibliography

[1] Aggarwal, C., Kaplan, H., Orlin, J. and Tarjan, R.,(1996) A Faster Primal Network Simplex Algorithm. Operations Research Center, Massachusetts Institute of Technology. OR 315-96.

[2] Asano, T.(2000) Effective Use of Geometry properties for clustering. *JCDCG'98, LNCS 1763*, pp 30-46.

[3] Bach, F.R. and Jordan, M.I. Learning spectral clustering, *Advances in Neural Information Processing Systems (NIPS)*, 16, 2004.

[4] Baeza-yates, R. A.(1992) Introduction to data structures and algorithms related to information retrieval. In Information Retrieval: Data Structures and Algorithms, W. B. Frakes and R. Baeza-Yates, Eds. Prentice- Hall, Inc., Upper Saddle River, NJ, 13C27.

[5] Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N. V. and Yearwood, J.L.

(2003). Unsupervised and supervised data classification via nonsmooth and global optimization. *Top*, 11, no.1, pp 1-93.

[6] Bandyopadhyay, S., Maulik, U. and Pakhira, M. K.(2001) Clustering using simulated annealing with probabilistic redistribution. *International J. Patt. Recog. and Artif. Intell*, 15, pp 269-285.

[7] Batagelj, V. and Ferligoj, A.(1998) Constrained Clustering Problems. *Proceeding of IFCS'98.*

[8] Bradley, P., Bennet, K. and Demiriz, A.,(2000) Constrained K-Means Clustering.*MSR-TR-2000-65*, Microsoft Research.

[9] Bradley,P., Fayyad, U.M., and Mangasarian, O.L.(1999) Mathematical Programming for data mining: formulations and challenges. *Informs J. Comput.*, 11, 217-238.

[10] Carpenter, G. and Grossberg, S.(1990) Art3: Hierarchical search using chemical transmitters in self organising pattern recognition architectures. *Neural Networks*, 3:129C152

[11] Davidson I., Ravi, S.S.(2005) Clustering under Constraints: Feasibility Issues and the K-Means Algorithm, *Proceeding of 5th SIAM Data Mining Conference.*

[12] Dempster, A.P., Laird, N.M. and Bubin, D.B.(1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Stats. Society.* B. 39 1, 1-38

[13] Dhillon, I., Guan, Y. and Kulis, B.(2004) A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts. *UTCS Technical Report TR-04-25.*

[14] Ding, C., He, X.(2004) Principal Component Analysis and Effective K-Means Clustering. *Proceeding of SIAM International Conference on Data Mining.*

[15] Ding, C. and He, X. (2004) K-means clustering via principal component analysis. *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada.

[16] DlGamal, A. E., Hemanchandra, L. A., Shperling, I. and Wei, V. K.,(1987) Using simulated annealing to design good codes. *IEEE Trans. Information Theory*, 33, pp 116-123.

[17] Drineas, P., Frieze, A., Kannan, R., Vempala, R. and Vinay, V. (2004). Clustering large graphs via singular value decomposition. *Machine Learning*, 56, 9-33.

[18] Goemans, M.X. (1997). Semidefinite programming in combinatorial optimization. *Mathematical Programming.* 79, 143–161.

[19] Gulub, G. and Loan, C. V. (1996) *Matix Computation.* John Hopkins University Press.

[20] Gu, M., Zha, H., Ding, C., He, X. and Simon, H. (2001) Spectral relaxation models and structure analysis for k-way graph Clustering and bi-clustering. *Penn State Univ Tech Report.*

[21] Han, Jiawei., Kamber, Micheline.(2000) Data Mining: Concepts and Techniques. *Morgan Kaufmann*

[22] Hasegawa, S., Imai, H., Inaba, M.(1993) Efficient Algorithms for Variance-Based k-clustering. *First Pacific Conference on Computer Graphics and Applications*

[23] Hansen, P., Jaumard, B. and Mladenovic, N.(1998). Minumum sum of squares clustering in a low dimensional space. *J. Classification*, 15, 37-55.

[24] Hansen, P. and Jaumard, B.(1997) Clustering analysis and mathematical programming. *Math. Programming*, 79(B), pp 191-215.

[25] Hansen, P. and Mladenovic, N.(2001) J-means: a new local search heuris-

tic for minimum sum-of-squares clustering. *Pattern Recognition*, 34 (2), pp 405-413.

[26] He, J., Lan, M., Tan, C., Sung, S. and Low, H.(2004) Initialization of clusters refinement algorithms: a review and comparative study. *International Joint Conference on Neural Networks*, IJCNN, pp 25-29.

[27] Inaba, M., Katoh, N. and Imai, H.(1994) Applications of Weighted Voronoi Diagrams and Randomization to Variance-Based k-clustering. *Proceeding of 10th ACM Symposium on Computational Geometry*, pp 332-339.

[28] Jain, A.K., Murty, N.M. and Flynn, P.J.(1999) Data Clustering: A Review. *ACM Computing Surveys*, Vol.31 No.3, pp 264-323.

[29] Jolliffe, I. (2002) *Principal component analysis*. Springer, 2nd edition.

[30] Karisch, S.E. and Rendl, F. (1998) Semidefinite programming and graph equipartition. *Fields Institute Communications*. 18, 77-95.

[31] Kohonen, T.(1989) Self Organization and Associative Memory. Springer Infomation Sciences Series. Springer-Verlag, 3 edition.

[32] Leisser, A. and Rendl, F. (2003) Graph partitioning using linear and semidefinite programming. *Mathematical Programming (B)*. 95,91-101.

[33] Lu, S.Y. and Fu, K.S.(1978) A sentence to sentence clustering procedure for pattern analysis. *IEEE transactions on Systems Mans and Cybernetics*, 8:381-389.

[34] Matoušek, J.(2000) On Approximate Geometric k-clustering. *Discrete and Computational Geometry*, 24, pp 61-84.

[35] Mangasarian, O.L. (1997) Mathematical programming in data mining. *Data Min. Knowl. Discov.*, 1, 183-201.

[36] Meila, M. and Shi, J. (2001) A random walks view of spectral segmentation. *Int'l Workshop on AI & Sta.*

[37] Michalski, R.S. and Chilausky, R.L. (1980). Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis. *International Journal of Policy Analysis and Information Systems*, 4(2), pp 125-161.

[38] McCallum, A. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.umass.edu/~mccallum/bow/

[39] McQueen, J.(1967) Some methods for classification and analysis of multivariate observations. *Computer and Chemistry*, 4, 257-272.

[40] Murtagh, F.(1984) A survey of recent advances in hierarchical clustering algorithms which use cluster centers. *Comput. J.* 26, 354C359.

[41] Nagy, G. 1968. State of the art in pattern recognition. *Proc. IEEE* 56, 836C862.

[42] Ng, A.Y., Jordan, M.I. and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Proc. Neural Info. Processing Systems, NIPS*, 14.

[43] Pacheco, J. and Valencia, O.(2003) Design of hybrids for the minimum sum-of-squares clustering problem. *Computational Statistics and Data Analysis*, 43, pp 235-248.

[44] Peng, J. and Xia, Y.(2004) A new theoretical framework for K-means-type clustering. To appear in *Foundation and recent advances in data mining*, Eds Chu and Lin, Springer Verlag.

[45] Peng, J. and Xia, Y.(2005) A Cutting Plane Method for the Minimum-Sum-of-Squared Error Clustering. To appear in: *Proceeding of SIAM International Conference on Data Mining.*

[46] Rao, M. R.(1971) Clustering analysis and mathematical programming. *J. American Statistical Association*, 66, 622-626.

[47] Sartipi, K. and Kontogiannis, K. (2003) A user-assisted approach to component clustering. *Journal of Software Maintenance and Evolution: Research and Practice*, 15, 265-295.

[48] Sethi, I. and Jain, A.K.(1991) Artificial Neural Networks and Pattern Recognition: Old and new connestions. Elsevier Science, New York, NY

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

[50] Sneath, P. H. A. and Sokal, R. R.(1973) Numerical Taxonomy. Freeman, London, UK.

[51] Späth, H. (1980) *Cluster analysis Algorithms for Data Reduction and Classification of Objects*, John Wiley and Sons, Ellis Horwood Ltd.

[52] Strehl, A. and Ghosh, J. (2002) Relationship-Based Clustering and Visualization for High-Dimensional Data Mining *INFORMS Journal on Computing*, 00, 1-23.

[53] Tung, A., Ng, R., Lakshmanan, L. and Han, J.,(2001) Constraint-Based

Clustering in Large Databases. *Proceedings of the 8th International Conference on Database Theory*, pp 405 - 419.

[54] Vaisey, J. and Gersho, A.(1988) Simulated annealing and codebook design. *Proceeding of IEEE international conference on Acustics, Speech, and Signal Processing(ICASSP)*, pp 1176-1179.

[55] Viod, H. D.(1969) Integer programming and the theory of grouping, *J. American Statistical Association*, 64, 506-519.

[56] Ward, J. H. Jr.(1963) Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* 58, 236C244.

[57] Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. *Proceedings IEEE International Conference on Computer Vision*, 975-982.

[58] Wiggerts, TA. (1997). Using clustering algorithms in legacy systems modularization. emphProceeding Fourth Working Conference on Reverse Engineering, WCRE'97, IEEE Computer Society: Los Alamitos CA, 33-43.

[59] Xing, E.P. and Jordan, M.I. (2003) On semidefinite relaxation for normalized k-cut and connections to spectral clustering. *Tech Report CSD-03-1265*, UC Berkeley.

[60] Ye, N. Editor(2003). The Handbook of Data Mining, Lawrence Erlbaum Associate, Inc, pp. 247-277.

[61] Yu, S. and Shi, J. (2003) Multiclass Spectral Clustering *International Conference on Computer Vision (ICCV'01).*

[62] Zadeh, L.A.(1965) Fuzzy sets. *Information Control 8*, 338-353

[63] Zha, H., Ding, C., Gu, M., He, X. and Simon, H. (2002) Spectral Relaxation for K-means Clustering. In Dietterich, T., Becker, S. and Ghahramani, Z. Eds., *Advances in Neural Information Processing Systems 14*, pp. 1057-1064. MIT Press.