Approximating K-means-type clustering via semidefinite programming

Jiming Peng Yu Wei [*]

October 5, 2005

Abstract

One of the fundamental clustering problems is to assign n points into k clusters based on the minimal sum-of-squares(MSSC), which is known to be NP-hard. In this paper, by using matrix arguments, we first model MSSC as a so-called 0-1 semidefinite programming (SDP). We show that our 0-1 SDP model provides an unified framework for several clustering approaches such as normalized k-cut and spectral clustering. Moreover, the 0-1 SDP model allows us to solve the underlying problem approximately via the relaxed linear and semidefinite programming.

Secondly, we consider the issue of how to extract a feasible solution of the original 0-1 SDP model from the approximate solution of the relaxed SDP problem. By using principal component analysis, we develop a rounding procedure to construct a feasible partitioning from a solution of the relaxed problem. In our rounding procedure, we need to solve a K-means clustering problem in $\Re^{k-1}$, which can be solved in $O(n^{k^2-2k+2})$ time. In case of bi-clustering, the running time of our rounding procedure can be reduced to $O(n \log n)$. We show that our algorithm can provide a 2-approximate solution to the original problem. Promising numerical results for bi-clustering based on our new method are reported.

[*]The first author is the corresponding author. Advanced optimization Lab, Department of Computing and Software McMaster University, Hamilton, Ontario L8S 4K1, Canada. Email: pengj@mcmaster.ca, weiy3@mcmaster.ca. The research is supported by the grant # RPG 249635-02 of the National Sciences and Engineering Research Council of Canada (NSERC) and a PREA award. This research is also supported by the MITACS project “New Interior Point Methods and Software for Convex Conic-Linear Optimization and Their Application to Solve VLSI Circuit Layout Problems”.

Key words. K-means clustering, Principal component analysis, Semidefinite programming, 0-1 SDP, Approximation.

1 Introduction

In general, clustering involves partition a given data set into subsets based on the closeness or similarity among the data. Clustering is one of major issues in data mining and machine learning with many applications arising from different disciplines including text retrieval, pattern recognition and web mining[19, 23].

There are many kinds of clustering problems and algorithms, resulting from various choices of measurements used in the model to measure the similarity/dissimilarity among entities in a data set. Most clustering algorithms belong to two groups: hierarchical clustering and partitioning. The hierarchical approach produces a nested series of partitions consisting of clusters either disjoint or included one into the other. Those clustering algorithms are either agglomerative or divisive. An agglomerative clustering algorithm starts with every singleton entity as a cluster, and then proceeds by successively merging clusters until a stopping criterion is reached. A divisive approach starts with an initial cluster with all the entities in it, and then performs splitting until a stopping criterion is reached. 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. In contrast, partitional methods assume given the number of clusters to be found and then look for the optimal partition based on the object function. Partitional methods produce only one partition. 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. There are several other ways to categorize various clustering algorithms. For a comprehensive introduction to the topic, we refer to the book [19, 23], and for more recent results, see survey papers [9] and [20].

Among various criterion in clustering, the minimum sum of squared Euclidean distance from each entity to its assigned cluster center is the most intuitive and broadly used. Both hierarchical and partitional procedures for MSSC have been investigated. For example, Ward's [42] agglomerative approach for MSSC has a complexity of $O(n^2 \log n)$ where n is the number of entities. The divisive hierarchical approach is more difficult. In [15], the authors provided an algorithm running in $O(n^{d+1} \log n)$ time, where d is the dimension of the space to which the entities belong. Promising numerical results are reported for data sets in low dimensions.

However, in many applications, assuming a hierarchical structure in partitioning based on MSSC is unpractical. In such a circumstance, the partitional approach directly minimizing the sum of squares distance is more applaudable. The traditional way to deal with this problem is to use some heuristics such as the well-known K-means [32]. To describe the algorithm, let us go into a bit more details.

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 $\mathcal{S} = (S_1, \cdots, S_k)$ 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.,

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

where $|S_j|$ is the number of points in S_j, and $\mathbf{s}_i^{(j)}$ is the i^{th} point in S_j. Note that if the cluster centers are known, then the function $f(S, \mathcal{S})$ achieves its minimum when each point is assigned to its closest cluster center. Therefore, MSSC can be described by the following bilevel programming problem (see for instance [4, 29]).

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

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 S_j are fixed, then the function

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

is minimal when

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

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

The classical K-means algorithm [32], based on the above two observations, is described as follows:

K-means clustering algorithm

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

$$\min_{x_{ij}} \quad \sum_{j=1}^k \sum_{i=1}^n x_{ij} \left\| \mathbf{s}_i - \frac{\sum_{l=1}^n x_{lj}\mathbf{s}_l}{\sum_{l=1}^n x_{lj}} \right\|^2 \tag{2}$$

$$S.T. \quad \sum_{j=1}^k x_{ij} = 1 \ (i = 1, \cdots, n) \tag{3}$$

$$\sum_{i=1}^n x_{ij} \geq 1 \ (j = 1, \cdots, k) \tag{4}$$

$$x_{ij} \in \{0, 1\} \ (i = 1, \cdots, n; \ j = 1, \cdots, k) \tag{5}$$

The constraint (3) ensures that each point $\mathbf{s}_i$ is assigned to one and only one cluster, and (4) ensures that there are exactly k clusters. This is a mixed integer programming with nonlinear objective [14], which is NP-hard. 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.

Many different approaches have been proposed for attacking (2) both in the communities of machine learning and optimization [1, 14, 8]. Most methods for (2) are heuristics that can locate only a good local solution, not the exact global solution for (2). Only a few works are dedicated to the exact algorithm for (2) as listed in the references of [8]. In particular, by using the notion of *Voronoi Partitions*, Inaba et'al [18] showed that the exact solution of (2) can be located in $O(n^{kd+1})$ time. Due to its high complexity, the algorithm can be applied only to small data sets.

Approximation methods provide a useful approach for (2). There are several different ways to approximate (2). For example, Hasegawa et'al [16] proposed to select the k points from the present data set and then run the classical K-means for the selected centers. If we try all possible combinations of the k starting centers and output the best solution as the final one, then we can obtain a 2-approximation for (2) in $O(n^{k+1})$ time [16]. In [30], Mutousek proposed a geometric approximation method that can find an $(1+\epsilon)$ approximately optimal solution for (2) in $O(dn \log^k n)$ time, where the constant hidden in the big-O notation depends polynomially on ϵ^{-1}. Though theoretically efficient, so far no numerical results have been reported based on Matousek's algorithm. More recently, Kumar, Sabbarwal and Sen [25] proposed a linear time ($O(2^{\epsilon^{-O(1)}}nd)$ algorithm for K-means clustering based on random sampling techniques and showed that their algorithm can find an $(1+\epsilon)$ approximation with a certain probability. Only the theoretical analysis based on the probabilistic model is presented, however. For more results on approximation algorithms for K-means clustering based on randomization techniques, we refer the reader to [25] and the references therein.

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 [10]. We noted that recently, Xing and Jordan [44] considered the SDP relaxation for the so-called normalized k-cut spectral clustering.

In the present paper, we first focus on developing approximation methods for (2) based on semidefinite programming (SDP) relaxation. A crucial step in relaxing (2) is to rewrite the objective in (2) as a simple convex function of matrix argument that can be tackled easily, while the constraint set still enjoy certain geometric properties. This was possibly first suggested in [12] where the authors owed the idea to an anonymous referee. However, the authors of [12] did not explore the idea in depth to design any usable algorithm. A similar effort was made in [45] where the authors rewrote the objective in (2) as a convex quadratic function in which the argument is a $n \times k$ orthonormal matrix.

Our model follows the same stream as in [12, 45]. However, different from the approach [45] where the authors used only a quadratic objective and simple spectral relaxation, we elaborate more on how to characterize (2) exactly by means of matrix arguments. In particular, we show that MSSC can be modelled as the so-called 0-1 semidefinite programming (SDP), which can be further relaxed to polynomially solvable linear programming (LP) and SDP. Our model provides novel avenues not only for solving MSSC, but also for solving clustering problems based on some other criterions. For example, the K-means clustering in the kernel space and the clustering problem based on normalized cuts can also be embedded into our model. Moreover, by slightly change the constraints in the 0-1 SDP model, we can attack clustering problems with constraints such as in the so-called balanced clustering.

We also discuss how to find a good approximate solution for our 0-1 SDP model. For this, we first relax the model by removing some constraints. We show that the relaxed problem can be solved by using singular value decomposition of the underlying coefficient matrix. This is similar to the principal component analysis (PCA) for data analysis [22]. Then we introduce a rounding procedure to extract a feasible solution for the original 0-1 model. In our rounding procedure, we need to solve a clustering problem whose entities are in $\Re^{k-1}$.

Our rounding procedure follows similar idea as in the so-called spectral clustering [7, 45, 35], where PCA is used to project the data into a lower-dimensional space generated by the eigenvectors corresponding to the first largest k eigenvalues of the underlying coefficient matrix, and then K-means clustering is performed in the lower dimension. The interesting link between K-means clustering and principal component analysis has been noted by several experts in the machine learning community. For example, Ding and He [5] showed that a solution for MSSC can be constructed by using PCA and a simple rounding heuristics. In [7], Drineas et'al proposed to solve a K-means clustering problem in $\Re^k$ whose solution can be found in $O(n^{k^2+1})$ time and showed that their method can provide a 2-approximate solution to the original problem [2].

[2]It should be pointed out that although the algorithm in [7] enjoy some nice properties, the high complexity in solving the subproblem might prevent it from practical efficiency when dealing with large-scale data sets. For example, for the bi-clustering problem with a data set with $n = 10^4$, the running time of the algorithm in [7] will reach formidable $O(10^{20})$.

However, different from all the above-mentioned approaches, we project the coefficient matrix into a subspace in $\Re^{k-1}$. Mathematically, this equals to transfer all the points s_i to $s_i - \bar{s}$ where $\bar{s}$ is the geometric center of the whole data set. Such a process can be viewed as a simple normalization procedure in preprocessing data. Then we apply PCA to project the new normalized coefficient matrix into a lower space $\Re^{k-1}$ where k is the number of clusters. In the last step of our algorithm, we perform K-means clustering in space $\Re^{k-1}$, which can be solved in $O(n^{k^2-2k+2})$ time. We show that our algorithm can provide a 2-approximate solution to the original K-means clustering. In case that $k = 2$, a global solution to K-means clustering in $\Re^1$ can be found in $O(n \log n)$ time [3]. In particular, if $k = 2$ and $d << n$, then the overall complexity of our algorithm is $O(n \log n)$. Although such a complexity bound is only slightly better than the complexity of the algorithm in [30], our algorithm is practically much simpler and more efficient for many problems where $d << n$. This is because the hidden constant in the big-O notation in our estimation is a very small constant. This allows us to efficiently solve large-scale bi-clustering problems in many applications, in particular in the so-called hierarchical divisive clustering.

It should be mentioned that for the cases with $k \geq 3$, the complexity of our algorithm is not as good as the algorithm in [16] and [30]. However, since we are dealing with the unified 0-1 model, which covers numerous scenarios other than the classical K-means clustering, the proposed algorithm in the present work is still very interesting and helpful in solving these new problems arising from data mining and machine learning for which no approximation algorithms have been reported in the literature. On the other hand, we note that PCA is a very popular and powerful approach in data analysis, in particular for problems arising from text mining where the data is typically in very high dimension ($n << d$). The work [7] by Drineas et'al gave an insightful analysis and partially explains why the PCA-based approach works so efficiently in practice. In our paper, instead of the classical PCA, we suggest to perform a simple normalization on the given data set and then apply PCA to the normalized data set. We show that such a simple normalization process can improve the performance of PCA-based algorithms in cluster analysis. Since the PCA-based approach has become the working horse in the subject of cluster analysis for high-dimensional data [22], our algorithm provides a useful approach in many data mining applications. Moreover, if we use the algorithm proposed in [16] for the reduced problem, then we can find a 4-approximation to the original prob-

[3]It is worthwhile mentioning that K-means bi-clustering is also NP-hard [7].

lem in $O(n^{k+1})$ time. The approximation ratio obtained in such a way is worse than what obtained by applying the algorithm in [16] directly to the original data set. However, for the so-called kernel K-means clustering and spectral clustering, since $d \geq n$ when the underlying coefficient matrix W is nonsingular, the complexity of the algorithm in [16] is at least $O(n^{k+2})$. In this case, our algorithm has a lower iteration bound. Such an improvement holds for high-dimensional data as well.

The paper is organized as follows. In Section 2, we show that MSSC can be modelled as 0-1 SDP, which allows convex relaxation such as SDP and LP. In Section 3, we consider approximation algorithms for solving our 0-1 SDP model. We propose to use PCA to reduce the dimension of the problem, and then perform K-means clustering in the lower dimension. Approximate ratio between the obtained solution and the global solution of the original K-means clustering is estimated in Section 4. In Section 5, we report some preliminary numerical tests, and finally we close the paper by few concluding remarks.

2 0-1 SDP for K-means clustering

In this section, we establish the equivalence between K-means type and 0-1 SDP. The section has three parts. In the first part, we briefly describe SDP and 0-1 SDP. In the second part, we establish the equivalence between MSSC and 0-1 SDP model. In the last part, we explore the interrelation between 0-1 SDP and other clustering approaches.

2.1 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

$$\textbf{(SDP)}\left\{\begin{array}{ll} \min & \mathrm{Tr(WZ)} \\ S.T. & \mathrm{Tr(B_iZ) = b_i \quad fori} = 1, \cdots, \mathrm{m} \\ & Z \succeq 0 \end{array}\right.$$

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

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

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.

2.2 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 [37]. However, for self-completeness, we still give a detailed description of the reformulation process.

By rearranging the items in the objective of (2), we have

$$\begin{aligned} f(S, \mathcal{S}) \quad &= \sum_{i=1}^{n} \|\mathbf{s}_i\|^2 \left(\sum_{j=1}^{k} x_{ij} \right) - \sum_{j=1}^{k} \frac{\|\sum_{i=1}^{n} x_{ij}\mathbf{s}_i\|^2}{\sum_{i=1}^{n} x_{ij}} \qquad (6) \\ &= \mathrm{Tr}\big(\mathrm{W_S W_S^T}\big) - \sum_{\mathrm{j=1}}^{\mathrm{k}} \frac{\|\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{x_{ij}}\mathbf{s}_\mathrm{i}\|^2}{\sum_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{x_{ij}}}, \end{aligned}$$

where $W_S \in \Re^{n \times d}$ denotes the matrix whose i-th row is the transfer $\mathbf{s}_i^T$ 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 (6) as $\mathrm{Tr}\big(\mathrm{W_S W_S^T (I - Z)}\big) = \mathrm{Tr}\big(\mathrm{W_S^T W_S}\big) - \mathrm{Tr}\big(\mathrm{W_S^T W_S Z}\big)$. 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 (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

$$\begin{array}{ll} \min & \mathrm{Tr}\big(\mathrm{W_S W_S^T (I - Z)}\big) \\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z = Z^T, Z^2 = Z. \end{array} \tag{7}$$

Here $Z \geq 0$ means the componentwise inequality, and the constraints $Z^T = Z$ and $Z^2 = Z$ imply that Z is an orthogonal projection matrix. We first give a technical result about positive semidefinite matrix that will be used in our later analysis.

Lemma 2.1. *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 (7) and (2).

Theorem 2.2. *Solving the 0-1 SDP problem (7) is equivalent to finding a global solution of the integer programming problem (2).*

Proof. From the construction of the 0-1 SDP model (7), we know that one can easily construct a feasible solution for (7) from a feasible solution of (2). Therefore, it remains to show that from a global solution of (7), we can obtain a feasible solution of (2).

Suppose that Z is a global minimum of (7). Obviously Z is positive semidefinite. From Lemma 2.1 we conclude that there exists an index i_1 such that

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

Let us define the index set

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

Since $Z^2 = Z$, we have

$$\sum_{j \in \mathcal{I}_1} (Z_{i_1 j})^2 = Z_{i_1 i_1},$$

which implies

$$\sum_{j \in \mathcal{I}_1} \frac{Z_{i_1 j}}{Z_{i_1 i_1}} Z_{i_1 j} = 1.$$

From the choice of i_1 and the constraint

$$\sum_{j=1}^{n} Z_{i_1 j} = \sum_{j \in \mathcal{I}_1} Z_{i_1 j} = 1,$$

we can conclude that

$$Z_{i_1 j} = Z_{i_1 i_1}, \quad \forall j \in \mathcal{I}_1.$$

This further implies that the submatrix $Z_{\mathcal{I}_1 \mathcal{I}_1} = \frac{1}{|\mathcal{I}_1|} E_{|\mathcal{I}_1|}$ where E_m is a $m \times m$ matrix of all ones. Therefore, we can decompose the matrix Z into a bock matrix with the following structure

$$Z = \begin{pmatrix} Z_{\mathcal{J}_1 \mathcal{J}_1} & 0 \\ 0 & Z_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} \end{pmatrix}, \tag{8}$$

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

$$\begin{aligned} \min \quad & \mathrm{Tr}\Big((\mathrm{W_S W_S^T})_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} (\mathrm{I} - \mathrm{Z})_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} \Big) \\ & Z_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} e = e, \mathrm{Tr}\big(\mathrm{Z}_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1}\big) = \mathrm{k} - 1, \\ & Z_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} \ge 0, Z^2_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1} = Z_{\bar{\mathcal{I}}_1 \bar{\mathcal{I}}_1}. \end{aligned} \tag{9}$$

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

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

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 (7), we can obtain an assignment matrix that satisfies the conditions in (3)-(5). 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

$$\begin{aligned} \max \quad & \mathrm{Tr}\left(\mathrm{W_S W_S^T Z}\right) \\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z = Z^T, Z^2 = Z. \end{aligned} \tag{10}$$

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 (7) with (2). First, the objective in (7) is linear while the constraint in (7) is still nonlinear, even more complex than the 0-1 constraint in (2). The most difficult part in the constraint of (7) is the requirement that $Z^2 = Z$. Several different ways for solving (7) will be discussed in the next section.

2.3 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

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Ze = e, \mathrm{Tr(Z) = k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{11}$$

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

$$W_{ij} = \phi(\mathbf{s}_i, \mathbf{s}_j) = \exp^{-\frac{\|\mathbf{s}_i - \mathbf{s}_j\|^2}{\sigma}}, \quad \sigma > 0. \tag{12}$$

This leads to the so-called K-means clustering in the kernel space. In order to apply the classical K-means algorithm to (11), 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 (11). This is exactly the procedure what the recently proposed spectral clustering follows [2, 35, 43, 44]. 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 can be adapted to solve (11) directly without using the SVD decomposition of the affinity matrix.

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 [13, 33, 35, 40, 43, 44, 45]. In particular, Zha et'al [45] discussed the links between spectral relaxation and K-means. Similar ideas was also used in [35]. An SDP relaxation for normalized k-cut was discussed [44]. The relaxed SDP in [44] takes a form quite close to (16). 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 [44]. Let W be the affinity matrix defined by (12) 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 $D = \mathrm{diag}\,(d)$. The exact model for normalized k-cut in [44] can be rewritten as

$$\max_{X \in \mathcal{F}_k} \quad \mathrm{Tr}\big((\mathrm{X^T D X})^{-1} \mathrm{X^T W X}\big) \tag{13}$$

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 2.2, we can show that the model (13) equals to the following 0-1 SDP:

$$\begin{aligned} \min \quad & \mathrm{Tr}\Big(\mathrm{D}^{-\frac{1}{2}}\mathrm{W}\mathrm{D}^{-\frac{1}{2}}(\mathrm{I}-\mathrm{Z})\Big) \\ & Zd^{\frac{1}{2}} = d^{\frac{1}{2}}, \mathrm{Tr}(\mathrm{Z}) = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T. \end{aligned} \tag{14}$$

The only difference between (11) and (14) 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 [3] 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 (11), which leads to the following problem

$$\begin{aligned} \min \quad & \mathrm{Tr}(\mathrm{W}(\mathrm{I}-\mathrm{Z})) \\ & Z_{ii} \leq \frac{1}{\tilde{n}}, \quad i = 1, \cdots, n, \\ & Ze = e, \mathrm{Tr}(\mathrm{Z}) = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{15}$$

.

3 Approximate 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 (11). 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 (or Kernel K-means) where the kernel matrix is defined by (12) or some other kernel functions. It is worth mentioning that although we restrict our discussion to (11), however, with a little effort, our results can be extended to (14) as well.

The section consists of two parts. In the first subsection, we give a general introduction to algorithms for solving (11). In the second part, we introduce a new approximation method for (11).

3.1 Algorithms for 0-1 SDP

In this subsection, we discuss various algorithms for solving the 0-1 SDP model (11). From a viewpoint of the algorithm design, we can categorize all the algorithms for (11) into two groups. The first group consists of the so-called feasible iterative algorithms, where all the iterates are feasible regarding the constraints in (7) and the objective is decreased 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 (11). It is also easy to see that, many variants of the K-means algorithm such as the variants proposed in [17, 21], can also be interpreted as specific iterative schemes for (11).

The second group of algorithms for (11) consists of approximation algorithms that are based on LP/SDP relaxation. We starts with a general procedure for those algorithm.

Approximation Algorithm Based on Relaxation

Step 1: Choose a relaxation model for (7),
Step 2: Solve the relaxed problem for an approximate solution,
Step 3: Use a rounding procedure to extract a feasible solution to (7) 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 (11), then it is exactly an optimal solution of (11). On the other hand, when the approximation solution is not feasible regarding (11), we have to use a rounding procedure to extract a feasible solution.

Various relaxations and rounding procedures have been proposed for solving (11) in the literature. For example, in [37], Peng and Xia considered a relaxation of (11) based on linear programming and a rounding procedure was also proposed in that work. Xing and Jordan [44] 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 [13, 33, 35, 43, 45] 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) [22]. In [5], the link between PCA and K-means clustering was also explored and simple bounds were derived. In particular, Drineas et'al [7] 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 [40], 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. In this paper, we follow a similar idea as [40]. 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.

3.2 A New Approximation Method

In this subsection, we describe our SDP-based approximation method for (11). We start our discussion on various relaxation forms for (11).

First, recall that in (11), 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 (11) is replacing the requirement $Z^2 = Z$ by the relaxed condition

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

Note that in (11), 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 all the eigenvalues of Z are less than or equal to 1. In this circumstance, the constraint $Z \preceq I$ becomes superfluous and can be waived. Therefore, we obtain the following SDP relaxation [4]

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & Z \geq 0, Z \succeq 0. \end{aligned} \tag{16}$$

The above problem is feasible and bounded below. We can apply many existing optimization solvers such as interior-point methods to solve (16). It is known that an approximate solution to (16) can be found in polynomial time. However, we would like to point out here that although there exist theoretically polynomial algorithm for solving (16), most of the present optimization solvers are unable to handle the problem in large size efficiently.

Another interesting relaxation to (11) is to further relax (16) 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

$$\begin{aligned} \min \quad & \mathrm{Tr(W(I-Z))} \\ & Ze = e, \mathrm{Tr(Z)} = \mathrm{k}, \\ & I \succeq Z \succeq 0. \end{aligned} \tag{17}$$

In the sequel we discuss how to solve (17). Note that if Z is a feasible solution for (17), 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 (17), let us define

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

It is easy to see that

$$Z_1 = (I - \frac{1}{n} ee^T) Z = (I - \frac{1}{n} ee^T) Z (I - \frac{1}{n} ee^T), \tag{18}$$

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

$$\mathrm{Tr(Z_1)} = \mathrm{Tr(Z)} - 1 = \mathrm{k} - 1.$$

[4]In [44], the constraint $Ze = e$ in (11) is replaced by $Zd = d$, where d is a positive scaling vector associated with the affinity matrix.

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

$$W_1 = (I - \frac{1}{n}ee^T)W(I - \frac{1}{n}ee^T). \tag{19}$$

Then, we can reduce (17) to

$$\begin{aligned} \min \quad & \mathrm{Tr(W_1(I - Z_1))} \\ & \mathrm{Tr(Z_1) = k - 1}, \\ & I \succeq Z_1 \succeq 0. \end{aligned} \tag{20}$$

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 (20) can be achieved if and only if [36]

$$\mathrm{Tr(W_1Z_1)} = \sum_{\mathrm{i=1}}^{\mathrm{k-1}} \lambda_{\mathrm{i}}.$$

This gives us an easy way to solve (20) and correspondingly (17). The algorithmic scheme for solving (17) can be described as follows:

Relaxation Algorithm 1

Step 1: Calculate the projection W_1 via (19);

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

$$Z = \frac{1}{n}ee^T + \sum_{i=1}^{k-1} v^i {v^i}^T.$$

From our above discussion, we immediately have

Theorem 3.1. *Let Z^* be the global optimal solution of (11), and $\lambda_1, \cdots, \lambda_{k-1}$ be the first largest eigenvalues of the matrix W_1. Then we have*

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

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 [40] (See also [43]) 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 [35, 5].

Note that solving the relaxed problem (17) can not provide a solution for the original problem (11). In the sequel we propose a rounding procedure to extract a feasible solution for (11) from a solution of the relaxed problem (17) provided by the relaxation Algorithm 1. Our rounding procedure follows a similar vein as the rounding procedure in [7]. 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 1. 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 2.2, this equals to solving the following 0-1 SDP problem

$$\begin{aligned} \min \quad & \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z})\sum_{\mathrm{i}=1}^{\mathrm{k}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) \\ & Ze = e, \mathrm{Tr}(\mathrm{Z}) = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{21}$$

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

$$\begin{aligned} \min \quad & \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z})\sum_{\mathrm{i}=1}^{\mathrm{k}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right) \\ & Z_{ii} \geq \frac{1}{\tilde{n}} \quad i = 1, \cdots, n, \\ & Ze = e, \mathrm{Tr}(\mathrm{Z}) = \mathrm{k}, \\ & Z \geq 0, Z^2 = Z, Z = Z^T, \end{aligned} \tag{22}$$

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 can be described as follows.

Approximation Algorithm 2

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 (21) (or (22)) for (constrained) K-means clustering;

Step 4: Assign all the entities in $\mathcal{S}$ based on the assignment obtained from Step 3.

Before closing this section, we discuss briefly the computational cost involved in the first two steps in the above algorithm. In general, to compute the projection matrix W_1, we need $O(n^2)$ operations in total. Typically, performing the SVD for W_1 requires $O(n^3)$ time. If we use the power method or some other iterative methods [11] to compute the eigenvectors corresponding to the first $k-1$ largest eigenvalues of W_1, then the overall computational cost can be reduced to $O(kn^2)$. It should be mentioned that for the classical K-means clustering, we even do not need to compute the matrices W and W_1 explicitly. Recall that in case of the classical K-means clustering, we have $W = W_s W_{\mathcal{S}}^T$. It is straightforward to see that for calculating W_1, we can first perform a simple normalization for the data set $s_i = s_i - \bar{s}$ where $\bar{s}$ is the geometric center of the whole data set. Correspondingly, the task in Step 2 of the algorithm can be realized by performing the SVD on the new coefficient matrix $W_{\bar{S}}$ for the normalized data set, which can be done in $O(\min\{n,d\}^3 + \min\{n,d\}nd)$ time [5]. If $d << n$ (this is true for data sets in many applications), the computational cost involved in Step 2 is linear in n.

4 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 [7], Drineas et'al considered a similar algorithm similar 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 [7], a new analysis is necessary to investigate the approximation ratio of the solution obtained from Algorithm 2.

[5]To see this, let us first consider the case $d < n$. We can perform the SVD for the matrix $W_{\bar{S}}^T W_{\bar{S}} = V^T \mathrm{diag}\{\lambda_1, \cdots, \lambda_d\} V$, which takes $O(d^3)$ time. Then we can get the eigenvalues and their corresponding eigenvectors of $W_{\bar{S}}$ from the product $W_{\bar{S}} V$. The whole process takes only $O(d^3 + d^2 n)$ operations. The case for $d \geq n$ follows similarly.

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 2 is in $\Re$. In such a case, the task in Step 3 of Algorithm 2 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 following refined K-means clustering in one dimension.

Refined K-means in One Dimension

Step 0: Input the data set $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$;

Step 1: Sort the sequence so that

$$v_{i_1} \geq v_{i_2} \cdots \geq v_{i_n},$$

where $\{i_1, \cdots, i_n\}$ is a permutation of the index set $\{1, \cdots, n\}$;

Step 2: For $l = 1$ to n, set

$$C_1^l = \{v_{i_1}, \cdots, v_{i_l}\}, C_2^l = \{v_{i_{l+1}}, \cdots, v_{i_n}\},$$

and calculate the objective function

$$f(C_1^l, C_2^l) = \sum_{v_i \in C_1^l} \left(v_i - \frac{1}{l} \sum_{v_i \in C_1^l} v_i \right)^2 + \sum_{v_i \in C_2^l} \left(v_i - \frac{1}{n-l} \sum_{v_i \in C_2^l} v_i \right)^2.$$

based on the partition (C_1^l, C_2^l);

Step 3: Find the optimal partition (C_1^*, C_2^*) such that

$$(C_1^*, C_2^*) = \arg \min_{l \in \{1, \cdots, n\}} f(C_1^l, C_2^l),$$

and output it as the final solution.

The above algorithm is similar to the algorithm in [15] for divisive k-clustering in low dimension. 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 [7] or the algorithm in [18] [6] to solve problem (21). According to Theorem 5 of [18], the algorithm takes $O(n^{(k-1)^2})$ time to find the global solution of the subproblem in Step 3

[6] We point out that both works [7] and [18] employed the same technique for the MSSC model. However, in the present work, we cite only the results from [18] because the estimation of the complexity in [7] is not precise [6].

of Algorithm 2, which improves the running time when the same procedure is applied to solve the subproblem in [7]. This is because the working space in our algorithm is one dimension less than the space in [7]. In case of bi-clustering, the improvement is substantial since we can use our simple refined K-means in one dimension.

We next progress to estimate the solution obtained from Algorithm 2. Let Z^* be a global solution to (11) and $\bar{Z}$ is the solution provided by Algorithm 2. Let us define

$$U = \frac{1}{n}ee^T + \sum_{i=1}^{k-1} v^i (v^i)^T. \tag{23}$$

It follows

$$\mathrm{Tr}\left((\mathrm{I} - \mathrm{U}) \sum_{\mathrm{i}=1}^{\mathrm{k}-1} \mathrm{v}^{\mathrm{i}} (\mathrm{v}^{\mathrm{i}})^{\mathrm{T}} \right) = 0; \tag{24}$$

$$\mathrm{Tr}\left(\mathrm{U} \sum_{\mathrm{i}=\mathrm{k}}^{\mathrm{n}-1} \mathrm{v}^{\mathrm{i}} (\mathrm{v}^{\mathrm{i}})^{\mathrm{T}} \right) = 0.. \tag{25}$$

From Theorem 3.1, we have

$$\mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)) \geq \mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{U})). \tag{26}$$

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

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{U} - \bar{\mathrm{Z}})\big) \leq \mathrm{Tr}(\mathrm{W}(\mathrm{I} - \mathrm{Z}^*)), \tag{27}$$

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 2 is a 2-approximation to the original K-means clustering.

In what follows we prove (27), which can be equivalently stated as

$$\mathrm{Tr}\big(\mathrm{W}(\mathrm{I} - \mathrm{Z}^* + \bar{\mathrm{Z}} - \mathrm{U})\big) \geq 0. \tag{28}$$

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

$$(I - Z^* + \bar{Z} - U)e = 0, \tag{29}$$

$$(I - \frac{ee^T}{n})(I - Z^* + \bar{Z} - U) = (I - \frac{ee^T}{n})(I - Z^* + \bar{Z} - U)(I - \frac{ee^T}{n}). \tag{30}$$

It follows immediately that

$$\begin{aligned}
\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\big) &= \frac{1}{n}\mathrm{Tr}\big(\mathrm{W}(\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\mathrm{ee}^{\mathrm{T}}\big)+\mathrm{Tr}\big(\mathrm{W}_1(\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\big)\\
&= \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\sum_{\mathrm{i}=1}^{\mathrm{n}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right)\\
&= \mathrm{Tr}\left((\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\sum_{\mathrm{i}=1}^{\mathrm{k}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}})\right)\\
&\quad +\mathrm{Tr}\left((\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}}-\mathrm{U})\sum_{\mathrm{i}=\mathrm{k}}^{\mathrm{n}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right)\\
&= \mathrm{Tr}\left((\bar{\mathrm{Z}}-\mathrm{Z}^*)\sum_{\mathrm{i}=1}^{\mathrm{k}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}})\right)+\mathrm{Tr}\left((\mathrm{I}-\mathrm{Z}^*+\bar{\mathrm{Z}})\sum_{\mathrm{i}=\mathrm{k}}^{\mathrm{n}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}}\right)\\
&\geq \mathrm{Tr}\left((\bar{\mathrm{Z}}-\mathrm{Z}^*)\sum_{\mathrm{i}=1}^{\mathrm{k}-1}\lambda_{\mathrm{i}}\mathrm{v}^{\mathrm{i}}(\mathrm{v}^{\mathrm{i}})^{\mathrm{T}})\right),
\end{aligned}$$

where the last equality is given by (24) and (25), 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 (21) and Z^* is only a feasible solution of (21), we therefore have

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

which further implies (27).

Now we are ready to state the main result in this section, which follows immediately from (26) and (27).

Theorem 4.1. *Suppose that Z^* is a global solution to problem (11) and $\bar{Z}$ is the solution provided by Algorithm 2. 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 2 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 (22). However, suppose a global solution to (22) can be located, then by following a similar chain of reasoning as in the proof of Theorem 4.1, we can prove the following result.

Theorem 4.2. *Suppose that Z^* is a global solution to problem (15) and $\bar{Z}$ is the solution provided by Algorithm 2. Then, we have*

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

We next discuss the overall complexity of Algorithm 2. For general kernel K-means clustering or spectral clustering, the overall complexity becomes $O(kn^2 + n^{(k-1)^2})$. For the classical K-means clustering, the overall complexity is $O(\min\{n,d\}^3 + \min\{n,d\}mn + n^{(k-1)^2})$. However, in case $k = 2$ and $d << n$, then the complexity reduces to $O(n \log n)$. On the other hand, since our algorithm uses only SVD and the constant in the big-O notation is very small, the algorithm for bi-clustering is very efficient and can be implemented easily. Since bi-clustering is the basis of the so-called divisive hierarchical clustering, our results will have a significant impact on clustering methods based on divisive approaches.

We also point out it is also possible to use another approximate algorithm to solve the reduced problem in Algorithm 2. For example, if we apply the algorithm in [16] to the reduced problem, then we can find a 2-approximation to the reduced problem in $O(n^{k+1})$ time. From Theorem 4.1, we can immediately conclude that the obtained solution is a 4-approximation to the original problem. Note that the quality of the solution we have now is worse than what obtained by applying the algorithm in [16] to the original data set. However, when $d \geq n$, the complexity of the algorithm in [16] is at least $O(n^{k+2})$. This implies the complexity of our algorithm is better than the direct algorithm in [16] for scenarios like the kernel K-means clustering, spectral clustering or problems in high dimension.

5 Numerical Experiments

To test the new algorithm, we have done some preliminary numerical experiments on several data sets from the UCI Machine Learning Repository[7] 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 [11].

It should be mentioned that although the subproblem (21) can be solved by using the procedure in [7], 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 ($k = 2$).

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

We mention that in the following tables, we concentrate mainly on the quality of the obtained solutions. This is due to the fact that our tests are on bi-clustering problems. Based on our theoretical analysis, our algorithm enjoys the best complexity for the underlying problems compared with the algorithms in [16] and [37]. Take the Email spam database for example, the algorithm in this paper takes less than one second to find the solution, while such a problem can not be handled by using the LP relaxation in [37] because of the huge amount (n^3) of constraints introduced in the LP model [8].

Data Sets from the UCI Machine Learning Repository

- **Soybean Data Set (small):** see also [34]. This data set has 47 points in $\Re^{35}$.
- **The Späth's Postal Zones:** This data set is from [41] about the post zones in Bavaria. It has 89 points in $\Re^3$.
- **Spam E-mail Database:** created by M. Hopkins *et al*from Hewlett-Packard Labs. It has 4601 in $\Re^{57}$. 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 partition of the data sets from Approximation Algorithms 2, we use the classical K-means to further improve the partition. In other words, we only use Algorithms 2 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 [37] 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 [38].

Table 5.1: Results on three UCI data sets

[8]In one of our recent works [27], we also compared the algorithm in [16] with the so-called Q-means developed in [27] and our preliminary tests indicated that the Q-means algorithm over-performed the algorithm in [16]. For example, for the Email spam database under the same computational environment as in the present work, the Q-means took more than 300 seconds and the algorithm [16] took more than half hour to find a solution [27].

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 [37] 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. It should be noted that for the largest test problem (the Email spam database), 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, we list the result from [27], which is derived by a so-called Q-Means heuristic for the same data and same balanced constraint.

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 5.2 summaries the results.

Table 5.2: 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[9]. This data set has also been used in [5, 13, 45], where a similar framework as ours was

[9]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

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.

Table 5.2: 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

Before constructing the word-document matrices, we perform the preprocessing by using the **bow** toolkit, a preprocessor similar to what employed in [5, 13, 45]. In particular, we use the tokenization option such that the UseNet headers are stripped, since the headers include the name of the correct newsgroup, and we also apply stemming [31]. 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. This implies for each test, we have a data set with 100 points in $\Re^{500}$. 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, we use the linear programming relaxation model proposed in [37] to get a lower bound on the global optimum. More specifically, we implement the LP relaxation model (14) in [37] 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.

Table 5.4: 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

From the above 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.

6 Conclusions

In this paper, 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 algorithm can always find a global solution for bi-clustering.

Several important issues regarding the new framework remain open. First, for general $k \geq 3$, although subproblem (21) can be solved by using the procedure in [7], 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. More study is needed to address these questions.

Acknowledgement The authors would like to thank Dr. Y. Xia, Prof. F. Rendl, Prof. K.C. Toh for their useful advice in the preparation of this paper. They also thank two anonymous referees for their valuable suggestions.

References

[1] Agarwal, P.K. and Procopiuc. (2002). Exact and approximation algorithms for clustering. *Algorithmica*, 33, 201-226.

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

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

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

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

[6] Drineas, P. Private Communication, 2005.

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

[8] Du Merle, O., Hansen, P., Jaumard, B. and Mladenović, N. (2000). An interior-point algorithm for minimum sum of squares clustering. *SIAM J. Sci. Comput.,* 21, 1485-1505.

[9] Ghosh J.(2003). Scalable Clustering. In N. Ye, Editor, The Handbook of Data Mining, Lawrence Erlbaum Associate, Inc, pp. 247-277.

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

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

[12] Gordon, A.D. and Henderson, J.T. (1977). An algorithm for Euclidean sum of squares classification. *Biometrics.* 33, 355-362.

[13] 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.

[14] Hansen, P. & Jaumard B. (1997). Cluster analysis and mathematical programming. *Math. Programming*, 79(B), 191-215.

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

[16] Hasegawa, S., Imai, H., Inaba, M., Katoh, N. and Nakano, J. (1993). Efficient algorithms for variance-based k-clustering. In *Proc. First Pacific Conf. Comput. Graphics Appl., Seoul, Korea.* 1, 75-89. World Scientific. Singapore.

[17] Howard, H.(1966). Classifying a population into homogeneous groups. In Lawrence, J.R. Eds., *Opertational Research in Social Science*, Tavistock Publ., London.

[18] Inaba, M., Katoh, N. and Imai, H. Applications of weighted voroni diagrams and randomization to variance-based k-clustering:(extended abstract). In *Proceedings of the tenth annual symposium on Computational Geometry*, pages 332-339. ACM Press, 1994.

[19] Jain, A.K., & Dubes, R.C. (1988). *Algorithms for clustering data.* Englewood Cliffs, NJ: Prentice Hall.

[20] Jain, A.K., Murty, M.N. and Flynn, P.J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31, 264-323.

[21] Jancey, R.C.(1966). Multidimensional group analysis. *Australian J. Botany*, 14, 127-130.

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

[23] Kaufman, L. and Peter Rousseeuw, P. (1990). Finding Groups in Data, an Introduction to Cluster Analysis, John Wiley.

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

[25] A. Kumar, Y. Sabharwal, and S. Sen. A Simple Linear Time $(1+\epsilon)$-Approximation Algorithm for k-Means Clustering in Any Dimensions. 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04), pp. 454-462, 2004.

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

[27] Ma, G., Peng, J. and Wei, Y. (2005) On approximate balanced biclustering. *Lecture Notes on Computer Science 3595, Proceeding of the 11th international conference on Computing and Combinatorics*, pp. 661-670.

[28] Mangasarian, O.L. *Nonlinear Programming.* SIAM Publishers, Philadelphia 1994.

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

[30] Matousek, J. (2000). On approximate geometric k-clustering. *Discrete Comput. Geom.*, 24, 61-84.

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

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

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

[34] Michalski, R.S. and Chilausky, R.L. (1980a). 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), 125-161.

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

[36] Overton, M.L. and Womersley, R.S. (1993). Optimality Conditions and Duality Theory for Minimizing Sums of the Largest Eigenvalues of Symmetric Matrices, *Mathematical Programming* 62, pp. 321-357.

[37] Peng, J.M.. and Xia, Y. A new theoretical framework for K-means clustering, *Foundation and recent advances in data mining*, Eds Chu and Lin, Springer Verlag, pp. 79–98, 2005.

[38] Peng, J.M.. and Xia, Y. A Cutting Plane Method for the Minimum-Sum-of-Squared Error Clustering. *Proceeding of SIAM Conference on*

Data Mining, Eds H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, 2005

[39] Ruspini, E.H. (1970). Numerical methods for fuzzy clustering. *Inform. Sci.*, 2, 319-350.

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

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

[42] Ward, J.H. (1963). Hierarchical grouping to optimize an objective function. *J. Amer. Statist. Assoc.,* 58, 236-244.

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

[44] 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.

[45] 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.