

# A Cutting Algorithm for the Minimum Sum-of-Squared Error Clustering

Jiming Peng\*

Yu Xia†

## Abstract

The minimum sum-of-squared error clustering problem is shown to be a concave continuous optimization problem whose every local minimum solution must be integer. We characterize its local minima. A procedure of moving from a fractional solution to a better integer solution is given. Then we adapt Tuy's convexity cut method to find a global optimum of the minimum sum-of-squared error clustering problem. We prove that this method converges in finite steps to a global minimum. Promising numerical examples are reported.

## 1 Introduction.

*Clustering* (or *cluster analysis*) is one of the basic tools in data analysis. In this paper, we consider clustering based on minimum within-group sum-of-squared error criterion.

Many early studies on minimum sum-of-squared error clustering (or MSSC in brief) were focused on the well-known K-means algorithm [5, 13, 15] and its variants (see [12] for a survey). Usually, these methods can only reach a local solution, not a global minimum of the distortion function. From a theoretical viewpoint, the minimum sum-of-squared error

clustering problem can be formulated as a nonlinear integer programming model. In [7], Hansen and Jaumard give a review on optimization approaches to some general clustering problems. There are some attempts to solve the MSSC problem exactly through mathematical programming; however, only numerical examples on data sets with less than 200 samples are reported.

Next, we will briefly describe some mathematical models for the minimum sum-of-squared error clustering problem and introduce our algorithm.

### 1.1 Problem description.

To partition  $n$  entities into  $k$  groups, people usually cast an entity into a vector in a Euclidean space:  $\mathbf{a}_i = ((a_i)_1, \dots, (a_i)_d)^T \in \mathbb{R}^d$  ( $i = 1, \dots, n$ ), where  $d$  is the number of attributes of the entity. Although coordinates of different points may be the same, we assume that there are at least  $k + 1$  different points; otherwise, one only needs to group points with same coordinates together. Below are some mathematical programming models for the minimum sum-of-squared error clustering problem.

**Bi-level program** The objective of MSSC can be described as finding  $k$  representatives of the clusters  $\mathbf{c}_i$  ( $i = 1, \dots, k$ ), and an assignment of the  $n$  entities to the  $k$  representatives such that the total sum-of-squared errors within each cluster, i.e. the sum of squared Euclidean distance from each point to its cluster representative, is minimum. This problem can be represented as the following bi-level programming model (see for instance [14]).

(1)

$$\min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \sum_{i=1}^n \min\{\|\mathbf{a}_i - \mathbf{c}_1\|_2^2, \dots, \|\mathbf{a}_i - \mathbf{c}_k\|_2^2\}.$$

---

\*Advanced optimization Lab, Department of Computing and Software McMaster University, Hamilton, Ontario L8S 4K1, Canada (pengj@mcmaster.ca). Research partially supported by the grant # RPG 249635-02 of the National Sciences and Engineering Research Council of Canada (NSERC) and a PREA award. This research was 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".

†The Institute of Statistical Mathematics, 4-6-7 Minami-Azabu, Minato-ku, Tokyo 106-8569, Japan (yuxia@ism.ac.jp). Research supported in part through JSPS (Japan Society for the Promotion of Science).

This model is not easy to solve.

**Mixed integer program** The bi-level program is equivalent to partitioning the  $n$  points into  $k$  groups, and then for each group finding a representative such that the total within-group sum-of-squared Euclidean distances is minimized. Define the assignment matrix  $X = [x_{ij}] \in \mathbb{R}^{n \times k}$  as

$$x_{ij} \stackrel{\text{def}}{=} \begin{cases} 1 & \mathbf{a}_i \text{ assigned to } j\text{th group;} \\ 0 & \text{otherwise.} \end{cases}$$

Then (1) can be transformed to

$$(2a) \quad \min_{x_{ij}, \mathbf{c}_j} \sum_{j=1}^k \sum_{i=1}^n x_{ij} \|\mathbf{a}_i - \mathbf{c}_j\|_2^2$$

$$(2b) \quad \text{s.t.} \sum_{j=1}^k x_{ij} = 1 \quad (i = 1, \dots, n)$$

$$(2c) \quad \sum_{i=1}^n x_{ij} \geq 1 \quad (j = 1, \dots, k)$$

$$(2d) \quad x_{ij} \in \{0, 1\} \quad (i = 1, \dots, n; j = 1, \dots, k).$$

The constraint (2b) ensures that each point  $\mathbf{a}_i$  is assigned to one and only one group. It can be replaced by

$$(3) \quad \sum_{j=1}^k x_{ij} \geq 1 \quad (i = 1, \dots, n),$$

since the objective is minimization. And (2c) ensures that there are exactly  $k$  clusters. We will prove that this constraint is redundant later. In addition, for a cluster  $j$ , given  $x_{ij}$  ( $i = 1, \dots, n$ ) —  $x_{ij}$ 's are not necessarily integer — the distortion function  $\sum_{i=1}^n x_{ij} \|\mathbf{a}_i - \mathbf{c}_j\|_2^2$  is convex in  $\mathbf{c}_j$ , and attains its global minimum at the arithmetical mean of the entities in the cluster, i.e.,

$$\mathbf{c}_j^* = \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}}.$$

Therefore, (1) is equivalent to

$$(4a) \quad \min_{x_{ij}} \sum_{j=1}^k \sum_{i=1}^n x_{ij} \left\| \mathbf{a}_i - \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}} \right\|_2^2$$

$$(4b) \quad \text{s.t.} \sum_{j=1}^k x_{ij} = 1 \quad (i = 1, \dots, n)$$

$$(4c) \quad x_{ij} \in \{0, 1\} \quad (i = 1, \dots, n; j = 1, \dots, k).$$

This is the nonlinear integer programming model of the minimum sum-of-squared error clustering problem. Note that its objective is not convex in  $x_{ij}$  and the constraints (2d) are discrete. This makes the problem very hard to solve. There is no evidence that the standard IP techniques can be applied to MSSC on large data sets.

**Continuous relaxation** Another way to deal with (4) is to relax the integer constraints (2d) to  $x_{ij} \in [0, 1]$ .

$$(5) \quad \min_{x_{ij}} \sum_{j=1}^k \sum_{i=1}^n x_{ij} \left\| \mathbf{a}_i - \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}} \right\|_2^2$$

$$\text{s.t.} \sum_{j=1}^k x_{ij} = 1 \quad (i = 1, \dots, n)$$

$$\sum_{i=1}^n x_{ij} \geq 1 \quad (j = 1, \dots, k)$$

$$x_{ij} \geq 0 \quad (i = 1, \dots, n; j = 1, \dots, k)$$

To our knowledge, this is first considered by Gordon and Henderson [6], who have further proved that at a global minimizer of (5), all the variables  $x_{ij}$ 's must have values 0 or 1. This indicates that the global minima of (5) are exactly those of (4). Although that statement is correct, the proof in [6] is not rigorous and not easy to follow. In this paper, we will prove that every local minimum of (5) is integer; furthermore, our proof gives a procedure of moving from a fractional solution of (5) to an integer solution with better objective value.

## 1.2 Our algorithm.

In [19], Selim and Ismail have proved that a class of distortion functions used in K-means-type clustering are essentially concave functions of the assignment variables. Specializing their conclusion to MSSC, we can claim that the objective function (2a) is concave in the feasible domain of the continuous relaxation of (2). In this paper, we will give a direct but alternative proof of the concavity of the function given in (4a) under certain conditions. Although a concave function achieves its local minima at some extreme points of its feasible region, unless the function is strictly convex, it is not always true that all of its local minimizers are extreme points. For (5), we will show that at any of its local minimum, the assignment variables  $x_{ij}$ 's are either 0 or 1, even though (5) is not a strictly concave program. Thus, we can safely work on (5) instead of (4).

Since a concave objective function may have many local minima, the global minima to (5) is still very hard to locate. Neither [6] nor [19] addresses how to find such a global solution. Until recently there have been only a few works on finding the exact solutions of the MSSC problem. In [21], Tuy, Bagirov and Rubinov cast some clustering problems to d.c.<sup>1</sup> programs. And branch-and-bound methods are suggested to solve the resulting d.c. program. Only numerical results on small-size data sets are reported. In [2], Merle et'al consider the Lagrangian relaxation of (4)

$$(6) \quad \max_{\lambda_i \geq 0} \left\{ \sum_{j=1}^k \min_{x_{ij} \in \{0,1\}} \left[ \sum_{i=1}^n x_{ij} \left\| \mathbf{a}_i - \frac{\sum_{l=1}^n x_{lj} \mathbf{a}_l}{\sum_{l=1}^n x_{lj}} \right\|^2 - \left( \sum_{i=1}^n \lambda_i x_{ij} \right) \right] + \sum_{i=1}^n \lambda_i \right\}.$$

It is proved in [2] that there is no duality gap between (4) and its Lagrangian relaxation. Further, the authors of [2] propose to use Dinkelbach's nonlinear fractional programming method ([1]) combined with some heuristics to solve the subproblem of (6) for temporarily fixed  $\lambda_i$ . It should be noted that Dinkelbach's method is convergent only when the numera-

tor is convex and the denominator is positive, which is not satisfied by the function defined in (6). This partially explains why the pure Dinkelbach's method takes a longer time than some heuristic method to solve the subproblem of (6), as is observed in [2]. Moreover, the Lagrangian relaxation of the integer programming model deals directly with the objective, the total within-group sum-of-squared error, not the partitioning itself. Therefore, extra effort is needed to recover the optimal partition. Again, only numerical experiments on data sets having up to 150 samples are reported. To summarize, it is fair to claim that although the MSSC problem is an important problem that has attracted many researchers' attentions, no globally convergent and efficient algorithms have been reported in the literature, in particular for moderately large data sets.

The main target of this paper is to propose a globally convergent and efficient algorithm for the minimum sum-of-squared error clustering problem. For this purpose, we use the fact that the MSSC problem can be formulated as a concave minimization problem over a polyhedron. For self-completeness, we give a detailed and direct proof of the concavity of the function (4a). We also characterize the local optimality of (5) and (4). It should be mentioned that in [19], Selim and Ismail have discussed the local optimality conditions for a class of optimization problems based on K-means-type clustering. However, the theoretical framework in [19] focuses on the mathematical essence of the local optimality for several different clustering approaches, while our analysis is emphasized on the difference between the stop criteria of the K-means algorithm and the local optimality conditions of (4). We also compare the local optimality conditions of the continuous model (5) with those of its discrete counterpart (4). As we shall see later, our discussion can help us skip a local minimum and further improve the objective value.

Many methods for concave minimization have been proposed in the optimization community. Those include cone splitting ([20, 11]), successive underestimation ([3]) and branch and bound ([9]) etc. Among them, Tuy's convexity cut method ([10]) is particularly interesting because the complexity of each step is very low. Note that all the above-mentioned meth-

<sup>1</sup>Here, d.c. stands for the difference of convex functions.

ods are designed for full dimensional feasible region; thus are not directly applicable to (5). In this paper, we will adapt Tuy's cut method to solve the concave minimization problem derived from MSSC and prove its finite convergence. Promising numerical results will be provided.

The rest of the paper is organized as follows. In §2, we will prove that the minimum sum-of-squared error clustering problem can be formulated as a concave minimization problem and give a constructive proof showing that each local optimum of (5) is integer. We will also give local optimality conditions for (5) and compare it with that of (4). In §3, we will adapt Tuy's concavity cuts to find a global minimum of (5). Preliminary numerical examples will be given in §4.

Few words about notations throughout this paper. We use bold lower case letters for column vectors; lower case letters for scalars; capital letters for matrices. Superscript  $T$  is used to represent the transpose of a matrix or vector.

## 2 Characteristics of the integer formulation for MSSC and its continuous relaxation.

In this section, we will give some optimal conditions for (4) and its continuous relaxation (5) for the purpose of algorithm design. To describe the common characteristics of (4) and (5), we assume  $x_{ij} \in [0, 1]$  ( $i = 1, \dots, n$ ;  $j = 1, \dots, k$ ) is a continuous variable in this section.

Let  $\mathbf{x}_j \stackrel{\text{def}}{=} (x_{1j}, \dots, x_{nj})^T$  ( $j = 1, \dots, k$ ) denote the assignment vector for the  $j$ th cluster.

Represent the within-group sum-of-squared error for the  $j$ th group by the function:

$$s_j(\mathbf{x}_j) \stackrel{\text{def}}{=} \sum_{i=1}^n x_{ij} \sum_{p=1}^d \left[ (a_i)_p - \frac{\sum_{i=1}^n x_{ij} (a_i)_p}{\sum_{i=1}^n x_{ij}} \right]^2.$$

The total within-group sum-of-squared error is denoted as:

$$s(X) \stackrel{\text{def}}{=} \sum_{j=1}^k s_j(\mathbf{x}_j).$$

Denote the difference from entity  $\mathbf{a}_l$  to the centroid of the  $j$ th group as  $\mathbf{v}_{lj} \in \mathbb{R}^d$ . Here,  $\mathbf{a}_l$  is not necessarily assigned to the  $j$ th group.

$$(7) \quad \mathbf{v}_{lj} \stackrel{\text{def}}{=} \mathbf{a}_l - \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}} = \mathbf{a}_l - \mathbf{c}_j.$$

As we have mentioned in the introduction, the concavity of (2a) in the feasible domain of the continuous relaxation of (2) can follow from the conclusions in [19]. Below, we give a direct proof of the concavity of (4a) under certain conditions, since our algorithm is based on (4a) instead of (2a). Another reason for our proof is that some technical results in our proof will be used in our later discussion.

**Proposition 1** *The objective function (4a) is concave whenever  $\sum_{i=1}^n x_{ij} > 0$  ( $j = 1, \dots, k$ ).*

**Proof:** We give the gradient and Hessian of  $s(X)$  to verify that the objective function of (5) is concave in its feasible region.

$$\begin{aligned} \frac{\partial s(X)}{\partial x_{lj}} &= \left\| \mathbf{a}_l - \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}} \right\|_2^2 \\ &+ 2 \sum_{i=1}^n x_{ij} \left( \frac{\sum_{p=1}^d x_{pj} \mathbf{a}_p}{\sum_{p=1}^d x_{pj}} - \mathbf{a}_i \right)^T \\ &\quad \left( \frac{1}{\sum_{p=1}^d x_{pj}} \mathbf{a}_l - \frac{\sum_{p=1}^d x_{pj} \mathbf{a}_p}{\left( \sum_{p=1}^d x_{pj} \right)^2} \right) \end{aligned}$$

Since  $\sum_{i=1}^n x_{ij} \left( \frac{\sum_{p=1}^d x_{pj} \mathbf{a}_p}{\sum_{p=1}^d x_{pj}} - \mathbf{a}_i \right) = \mathbf{0}$  and  $\left( \frac{1}{\sum_{p=1}^d x_{pj}} \mathbf{a}_l - \frac{\sum_{p=1}^d x_{pj} \mathbf{a}_p}{\left( \sum_{p=1}^d x_{pj} \right)^2} \right)$  is independent of  $i$ , the second term vanishes. Therefore,

$$(8) \quad \frac{\partial s(X)}{\partial x_{lj}} = \left\| \mathbf{a}_l - \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i}{\sum_{i=1}^n x_{ij}} \right\|_2^2.$$

Let  $V_j$  represent the matrix whose  $l$ th row is the vector  $\mathbf{v}_{lj}$ . Let  $\mathbf{e}$  denote the vector of all 1's.

Then for any  $l, g \in \{1, \dots, n\}$  and  $j \neq m \in$

$\{1, \dots, k\}$ :

$$\begin{aligned} \nabla^2 s_j &= -\frac{2}{\mathbf{x}_j^T \mathbf{e}} AA^T + \frac{2}{(\mathbf{x}_j^T \mathbf{e})^2} (AA^T \mathbf{x}_j \mathbf{e}^T + \mathbf{e} \mathbf{x}_j^T A^T A) \\ &\quad - \frac{2 \mathbf{x}_j^T AA^T \mathbf{x}_j}{(\mathbf{x}_j^T \mathbf{e})^3} \mathbf{e} \mathbf{e}^T = -\frac{2}{\sum_{i=1}^n x_{ij}} V_j V_j^T, \\ \frac{\partial^2 s}{\partial x_{lj} \partial x_{gm}} &= 0 \quad (j \neq m). \end{aligned}$$

Denote  $\mathbf{x} \stackrel{\text{def}}{=} (\mathbf{x}_1^T, \dots, \mathbf{x}_n^T)$ . It follows that

$$\nabla^2 s(\mathbf{x}) = \text{Diag}(\nabla^2 s_1(\mathbf{x}_1), \dots, \nabla^2 s_k(\mathbf{x}_k)).$$

Hence  $-\nabla^2 s(\mathbf{x})$  is positive semidefinite whenever  $\sum_{i=1}^n x_{ij} > 0$  ( $j = 1, \dots, k$ ). This implies that the objective function of (5) is concave in its feasible region.  $\blacksquare$

Observe that  $s(x)$  is not strictly concave under the above assumptions. Next, we will show that each local minimizer of (5) is integer, so that we can work on (4) instead of (4). We will also discuss the optimal conditions for both (4) and (5).

First, we give a proposition that will be used to prove some properties of local minima and construct our algorithm.

**Proposition 2** *Perturb  $x_{lj}$  by  $\Delta x_{lj}$ . Then the new within-group sum-of-squared distances about the new centroid  $\mathbf{c}'_j$  for the  $j$ th group,  $s_j(\mathbf{x}_j + \Delta \mathbf{x}_j)$ , is*

$$(9) \quad \begin{cases} s_j(\mathbf{x}_j) + \Delta x_{lj} \|\mathbf{v}_{lj}\|_2^2 - \frac{(\Delta x_{lj})^2 \|\mathbf{v}_{lj}\|_2^2}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} & \sum_{i=1}^n x_{ij} + \Delta x_{lj} \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

The difference from  $\mathbf{a}_l$  to  $\mathbf{c}'_j$  is

$$(10) \quad \mathbf{v}'_{lj} = \begin{cases} \mathbf{0} & \sum_{i=1}^n x_{ij} + \Delta x_{lj} = 0, \\ \frac{\sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \mathbf{v}_{lj} & \sum_{i=1}^n x_{ij} + \Delta x_{lj} \neq 0. \end{cases}$$

**Proof:** 1) We first consider the case  $\sum_{i=1}^n x_{ij} + \Delta x_{lj} = 0$ . This condition means that the  $j$ th group is empty after perturbation by  $\Delta x_{lj}$ . Therefore

$$s_j(\mathbf{x}_j + \Delta \mathbf{x}_j) = 0, \quad \mathbf{v}_{lj} = \mathbf{0}.$$

2) Now assume  $\sum_{i=1}^n x_{ij} + \Delta x_{lj} \neq 0$ .

If  $\sum_{i=1}^n x_{ij} = 0$ , the  $j$ th group is empty before perturbation. In this case,  $s_j(\mathbf{x}_j + \Delta \mathbf{x}_j) = 0$  and  $\mathbf{v}'_{lj} = \mathbf{0}$ , since the  $j$ th group now contains only  $\Delta x_{lj} \mathbf{a}_l$ .

In the remaining of the proof, we assume  $\sum_{i=1}^n x_{ij} \neq 0$ . After perturbation, the  $j$ th centroid is

$$\mathbf{c}'_j = \frac{\sum_{i=1}^n x_{ij} \mathbf{a}_i + \Delta x_{lj} \mathbf{a}_l}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}}.$$

By (7),

$$(11) \quad \sum_{i=1}^n x_{ij} \mathbf{a}_i = (\mathbf{a}_l - \mathbf{v}_{lj}) \left( \sum_{i=1}^n x_{ij} \right).$$

Plugging (11) into the expression of  $\mathbf{c}'_j$ , we get

$$(12) \quad \begin{aligned} \mathbf{c}'_j &= \frac{(\mathbf{a}_l - \mathbf{v}_{lj}) \left( \sum_{i=1}^n x_{ij} \right) + \Delta x_{lj} \mathbf{a}_l}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \\ &= \mathbf{a}_l - \frac{\sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \mathbf{v}_{lj}. \end{aligned}$$

Using (7) and (12), we have

$$(13) \quad \begin{aligned} \mathbf{c}_j - \mathbf{c}'_j &= (\mathbf{a}_l - \mathbf{v}_{lj}) - \left( \mathbf{a}_l - \frac{\sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \mathbf{v}_{lj} \right) \\ &= -\frac{\Delta x_{lj}}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \mathbf{v}_{lj}. \end{aligned}$$

Then

$$\begin{aligned} s_j(\mathbf{x}_j + \Delta \mathbf{x}_j) &= \sum_{i=1}^n x_{ij} \|\mathbf{a}_i - \mathbf{c}_j + \mathbf{c}_j - \mathbf{c}'_j\|_2^2 \\ &\quad + \Delta x_{lj} \|\mathbf{a}_l - \mathbf{c}'_j\|_2^2 = 2 \sum_{i=1}^n x_{ij} (\mathbf{a}_i - \mathbf{c}_j)^T (\mathbf{c}_j - \mathbf{c}'_j) \\ &\quad + \sum_{i=1}^n x_{ij} \|\mathbf{c}_j - \mathbf{c}'_j\|_2^2 + s_j(\mathbf{x}_j) + \Delta x_{lj} \|\mathbf{a}_l - \mathbf{c}'_j\|_2^2. \end{aligned}$$

The first term vanishes, because by the definition of  $\mathbf{c}_j$ ,

$$\sum_{i=1}^n x_{ij} (\mathbf{a}_i - \mathbf{c}_j) = \mathbf{0}.$$

After plugging (12) and (13) into the above expression of  $s_j(\mathbf{x}_j + \Delta \mathbf{x}_j)$ , we get

$$\begin{aligned} s_j(\mathbf{x}_j + \Delta \mathbf{x}_j) &= s_j(\mathbf{x}_j) + \frac{\sum_{i=1}^n x_{ij} (\Delta x_{lj})^2}{(\sum_{i=1}^n x_{ij} + \Delta x_{lj})^2} \|\mathbf{v}_{lj}\|_2^2 \\ &\quad + \frac{(\sum_{i=1}^n x_{ij})^2 \Delta x_{lj}}{(\sum_{i=1}^n x_{ij} + \Delta x_{lj})^2} \|\mathbf{v}_{lj}\|_2^2 \\ &= s_j(\mathbf{x}_j) + \frac{\sum_{i=1}^n x_{ij} \Delta x_{lj}}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \|\mathbf{v}_{lj}\|_2^2 \\ &= s_j(\mathbf{x}_j) + \Delta x_{lj} \|\mathbf{v}_{lj}\|_2^2 - \frac{(\Delta x_{lj})^2}{\sum_{i=1}^n x_{ij} + \Delta x_{lj}} \|\mathbf{v}_{lj}\|_2^2. \end{aligned}$$

Hence, (9) is proved. The expression (10) is from (12). ■

**Corollary 1** *At a local optimum of (4) or that of its continuous relaxation (5) (without constraint (2c)),*

1. *no cluster is empty;*
2. *all the centroids  $\mathbf{c}_j$ 's are distinct.*

**Proof:** To prove (1), we only need to show that the total sum-of-squared error of a feasible solution to (4) or (5) with an empty cluster is strictly larger than that of some other feasible solution in its neighborhood (related to (4) or (5)). Suppose the  $g$ th cluster ( $1 \leq g \leq k$ ) is empty. From (2b) and the assumption that there are at least  $k$  different points, we deduce that there exist at least a cluster  $j$  and two different entities  $\mathbf{a}_l, \mathbf{a}_m$ , so that  $x_{lj} > 0, x_{mj} > 0$ . In addition,  $\mathbf{a}_l \neq \mathbf{c}_j$ . Perturbing  $x_{lj}$  and  $x_{lg}$  to  $x_{lj} - \Delta x$  and  $x_{lg} + \Delta x$ , with  $0 < \Delta x \leq x_{lj}$  ( $\Delta x = 1$  for (4)), by (9), we get

$$s(X) - s(X + \Delta X) = \frac{\sum_{i=1}^n x_{ij} \Delta x}{\sum_{i=1}^n x_{ij} + \Delta x} \|\mathbf{v}_{lj}\|_2^2 > 0.$$

Therefore,  $X$  is not a local minimum. This shows that the constraint (2c) is redundant.

Next, we prove that no two clusters have the same centroid at a local minimum. Assume an assignment  $X$  has two centroids  $\mathbf{c}_j = \mathbf{c}_g$ . Merging the  $j$ th and the  $g$ th clusters will not change the total sum-of-squared error, but produce an empty cluster. By (1),  $X$  is not a local minimum.

Hence, we have proved (1) and (2). ■

The above results also imply that the optimum sum-of-squared error decreases with  $k$ .

**Lemma 1** *The matrix  $X$  is a local minimum of (5), iff for each  $l \in \{1, \dots, n\}$  and  $j \in \{1, \dots, k\}$ ,*

$$(14) \quad x_{lj} = \begin{cases} 1 & \|\mathbf{v}_{lj}\|_2 < \|\mathbf{v}_{lm}\|_2 \ (\forall m = 1, \dots, k, m \neq j), \\ 0 & \text{otherwise.} \end{cases}$$

**Remark 1** *The relation (14) equals to the following conditions:*

- (i) *all the assignment variables have values 0 or 1, i.e., no entity is assigned fractionally to two clusters;*
- (ii) *for every entity  $\mathbf{a}_l$  ( $l = 1, \dots, n$ ), there is only one cluster whose centroid is closest to it;*
- (iii) *every entity  $\mathbf{a}_l$  ( $l = 1, \dots, n$ ) is assigned to the cluster whose centroid is closest to it.*

**Proof:** We first prove the sufficiency, i.e., if (14) is satisfied by some  $X^* \in \mathbb{R}^{n \times k}$ ,  $X^*$  is a local minimum of (5). It suffices to prove that for any feasible search direction  $Y \neq 0$ , the directional derivative

$$D_Y s(X^*) \stackrel{\text{def}}{=} \lim_{t \downarrow 0} \frac{s(X^* + tY) - s(X^*)}{t} > 0.$$

By (8),

$$(15) \quad D_Y s(X^*) = \sum_{i=1}^n \sum_{j=1}^k \|\mathbf{v}_{ij}\|_2^2 y_{ij}.$$

Because  $X^*$  satisfies (2b) and  $Y$  is a feasible search direction, we have

$$(16) \quad \sum_{j=1}^k y_{lj} = 0 \quad (l = 1, \dots, n).$$

In addition, from (14) and the constraints  $0 \leq x_{lj} \leq 1$ , we get for all  $l \in \{1, \dots, n\}$  and  $j \in \{1, \dots, k\}$ ,

$$(17) \quad y_{lj} \begin{cases} \leq 0 & \|\mathbf{v}_{lj}\|_2 < \|\mathbf{v}_{lm}\|_2 \ (\forall m = 1, \dots, k, m \neq j), \\ \geq 0 & \text{otherwise.} \end{cases}$$

Combining (16) and (17), we conclude that

$$\sum_{j=1}^k \|\mathbf{v}_{lj}\|_2^2 y_{lj} \geq 0 \quad (l = 1, \dots, n).$$

Since  $Y \neq 0$ , the above inequality holds strictly for at least one  $l \in \{1, \dots, n\}$ . Therefore,

$$D_Y(X^*) > 0.$$

Next, we use contradiction to prove the necessity.

Suppose at a local minimizer of (5), denoted as  $X$ , the assignment of entity  $\mathbf{a}_l$  didn't satisfy the conditions in (14). Assume the distance from  $\mathbf{a}_l$  to the  $j$ th cluster is one of the shortest, i.e.,

$$j \in \arg \min_y \|\mathbf{v}_{ly}\|_2.$$

Suppose (i) or (iii) was violated, i.e.,  $x_{lj} \neq 1$ . Then there existed  $m \neq j$  such that  $x_{lm} > 0$  and  $\|\mathbf{v}_{lm}\|_2 \geq \|\mathbf{v}_{lj}\|_2$ . Note that  $\|\mathbf{v}_{lm}\|_2 = \|\mathbf{v}_{lj}\|_2$  would be possible if the  $j$ th cluster was not the only closest cluster to  $\mathbf{a}_l$ , i.e.,  $m \in \arg \min_y \|\mathbf{v}_{ly}\|_2$ .

If (ii) wasn't satisfied as well, i.e.,  $\|\mathbf{v}_{lm}\|_2 = \|\mathbf{v}_{lj}\|_2$ . Then neither  $\|\mathbf{v}_{lj}\|_2$  nor  $\|\mathbf{v}_{lm}\|_2$  could be zero. Were either of them zero,  $\mathbf{a}_l$  would be the centroid of both the  $j$ th and  $m$ th clusters, which violated Corollary 1. This also shows that  $\sum_{i=1}^n x_{im} > x_{lm}$  whether  $\|\mathbf{v}_{lm}\|_2 = \|\mathbf{v}_{lj}\|_2$  or  $\|\mathbf{v}_{lm}\|_2 \neq \|\mathbf{v}_{lj}\|_2$ , since otherwise  $\mathbf{a}_l$  would be the unique element in the  $m$ th cluster; hence  $\mathbf{v}_{lm} = 0$ .

According to (9), perturbing  $x_{lj}$  and  $x_{lm}$  to

$$x'_{lj} = x_{lj} + \Delta x, \quad x'_{lm} = x_{lm} - \Delta x, \quad (0 < \Delta x \leq x_{lm})$$

would not violate any constraints of (5), but would decrease its objective value by

$$\begin{aligned} & \Delta x \left( \|\mathbf{v}_{lm}\|_2^2 - \|\mathbf{v}_{lj}\|_2^2 \right) \\ & + (\Delta x)^2 \left( \frac{\|\mathbf{v}_{lj}\|_2^2}{\sum_{i=1}^n x_{ij} + \Delta x} + \frac{\|\mathbf{v}_{lm}\|_2^2}{\sum_{i=1}^n x_{im} - \Delta x} \right) > 0. \end{aligned}$$

In other words, consider the feasible search direction  $Y$  whose only nonzero entries were  $y_{lj} = 1$ ,  $y_{lm} = -1$ . By (1), the first order and second order

directional derivative of  $s$  in the direction  $Y$  evaluated at  $X$  would be

$$\begin{aligned} D_Y s(X) &= \|\mathbf{v}_{lj}\|_2^2 - \|\mathbf{v}_{lm}\|_2^2 \leq 0, \\ (18) \quad D_{YY} s(X) &\stackrel{\text{def}}{=} \lim_{t \downarrow 0} \frac{D_Y s(X + tY) - D_Y s(X)}{t} \\ &= -\frac{2\|\mathbf{v}_{lj}\|_2^2}{\sum_{i=1}^n x_{ij}} - \frac{2\|\mathbf{v}_{lm}\|_2^2}{\sum_{i=1}^n x_{im}} < 0. \end{aligned}$$

Therefore,  $Y$  would be a strictly descent direction of  $s$  at  $X$ .

That means  $X$  could not be a local minimum for  $s$ .

In addition, by (10),  $\|\mathbf{v}_{lj}\|_2$  would decrease which would make  $j$  the unique solution to

$$\arg \min_y \|\mathbf{v}_{ly}\|_2.$$

Proceeding with the above procedure, we can increase  $x_{lj}$  to 1 and obtain a better objective value.

Suppose that only (iii) was violated, i.e.,  $x_{lj} = 1$ , and  $\exists m \neq j$ ,  $\|\mathbf{v}_{lm}\|_2 = \|\mathbf{v}_{lj}\|_2$ . Then similarly as above, perturbing  $x_{lj}$  and  $x_{lm}$  to  $x_{lj} - \Delta x$  and  $x_{lm} + \Delta x$  would decrease the total within-group sum-of-squared error and make  $m$  the unique solution to  $\arg \min_y \|\mathbf{v}_{ly}\|_2$ .

In other words, consider the feasible direction  $Y$  whose only nonzero entries were  $y_{lj} = -1$ ,  $y_{lm} = 1$ . Similarly as (18),  $Y$  would be a strictly descent direction of  $s$  at  $X$ .

Thus, we have shown that a local optimum of (5) must satisfy (14).  $\blacksquare$

**Remark 2** Note that (14) is not a sufficient condition for a local minimum of the integer programming model (4). To see this, consider the following example:

$$d = 1, \quad k = 2, \quad \mathbf{a}_1 = -2, \quad \mathbf{a}_2 = 0, \quad \mathbf{a}_3 = 3.$$

The optimal clustering is  $(\{\mathbf{a}_1, \mathbf{a}_2\}, \{\mathbf{a}_3\})$ . However, the clustering  $(\{\mathbf{a}_1\}, \{\mathbf{a}_2, \mathbf{a}_3\})$  also satisfies (14).

Next we give a necessary and sufficient condition for a local minimum of (5).

Setting  $\Delta \mathbf{v}_{lj} = -1$  and  $\Delta \mathbf{v}_{lg} = 1$  in (2), we get the following.

- Switching  $\mathbf{a}_l$  from the  $j$ th cluster to the  $g$ th cluster will change the total sum-of-squared error by

$$(19) \quad \begin{cases} \frac{\|\mathbf{v}_{lg}\|_2^2 \sum_{i=1}^n x_{ig}}{\sum_{i=1}^n x_{ig} + 1} - \frac{\|\mathbf{v}_{lj}\|_2^2 \sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij} - 1} & \sum_{i=1}^n x_{ij} \neq 1 \\ \frac{\|\mathbf{v}_{lg}\|_2^2 \sum_{i=1}^n x_{ig}}{\sum_{i=1}^n x_{ig} + 1} & \text{otherwise.} \end{cases}$$

In our algorithm, we use (19) to find the steepest descent neighbor vertex and calculate the objective value change after the reassignment.

From (19), we have the following conclusion.

- At a local optimum of (4), the entity  $\mathbf{a}_l$  is assigned to the  $j$ th cluster iff 1)  $\mathbf{a}_l$  is the unique entity of the  $j$ th cluster; or 2) for any  $m \in \{1, \dots, k\}$ ,  $m \neq j$ :

$$(20) \quad \frac{\sum_{i=1}^n x_{im}}{\sum_{i=1}^n x_{im} + 1} \|\mathbf{v}_{lm}\|_2^2 \geq \frac{\sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij} - 1} \|\mathbf{v}_{lj}\|_2^2.$$

K-means type algorithms and some other heuristic algorithms for the minimum sum-of-squared error clustering are based on assigning each entity to the cluster whose centroid is closest to it; so these methods may find a local minimum of (5), but cannot guarantee to find a local minimum of (4), let along a global minimum.

Note that (20) is stronger than (14); so although our algorithm is designed for (5), we use (20) to search for a local minimum.

### 3 Concavity cuts for MSSC.

Our finitely convergent algorithm for the minimum sum-of-squared error clustering problem is based on concave optimization technique. A large number of approaches for concave minimization problems can be traced back to Tuy's cutting algorithm [20] for minimizing a concave function over a full dimensional polytope. We will briefly describe Tuy's cuts in the first part of this section for completeness. In general, without adding more expensive cuts, this procedure cannot find a global optimum in finite steps. Furthermore, Tuy's cuts can't be applied directly to (5), because its feasible region doesn't have full dimension.

In the second part of this section, we will show how to adapt Tuy's cutting algorithm to (5) and prove that this method can find a global minimum of (5) in finite steps.

#### 3.1 Basic ideas of Tuy's cuts.

For self-completeness, we sketch Tuy's cuts (also known as convexity cuts) below (see [10] for details). We assume  $\mathbf{x} \in \mathbb{R}^n$  in this subsection.

Tuy's cuts are originally designed to find a global minimum of a concave function  $f(\mathbf{x})$  over a polyhedron  $D \in \mathbb{R}^n$ . It requires 1)  $D$  has full dimension, i.e.  $\text{int } D \neq \emptyset$ ; 2) for any real number  $\alpha$ , the level set  $\{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) \geq \alpha\}$  is bounded.

Let  $\mathbf{x}^0$  be a local minimum and a vertex of  $D$ . Denote  $\gamma = f(\mathbf{x}^0)$ . Since  $D$  is full-dimensional,  $\mathbf{x}^0$  has at least  $n$  adjacent vertices. Let  $\mathbf{y}^1, \dots, \mathbf{y}^p$  denote the vertices adjacent to  $\mathbf{x}^0$  ( $p \geq n$ ). For  $i = 1, \dots, n$ , let

$$(21) \quad \theta_i \stackrel{\text{def}}{=} \sup\{t : t \geq 0, f(\mathbf{x}^0 + t(\mathbf{y}^i - \mathbf{x}^0)) \geq \gamma\};$$

denote

$$\mathbf{z}^i \stackrel{\text{def}}{=} \mathbf{x}^0 + \theta_i(\mathbf{y}^i - \mathbf{x}^0).$$

Then the cone originated at  $\mathbf{x}^0$  generated by the halflines in the directions  $\mathbf{z}^i$  covers the feasible region. Because  $f$  is concave, any point in the simplex  $\text{Spx} \stackrel{\text{def}}{=} \text{conv}\{\mathbf{x}^0, \mathbf{z}^1, \dots, \mathbf{z}^n\}$  has objective value no less than  $\gamma$ . Therefore, one can cut off Spx from further search for a global minimum. Since  $\mathbf{x}^0$  is a vertex of  $D$  which has full dimension, one can always find  $n$  binding constraints at  $\mathbf{x}^0$ , and  $\mathbf{x}^0$  has  $n$  linearly independent edges. Without loss of generality, assume  $\mathbf{z}^1 - \mathbf{x}^0, \dots, \mathbf{z}^n - \mathbf{x}^0$  are linearly independent. Define

$$(22) \quad \pi = \mathbf{e}^T \text{Diag}\left(\frac{1}{\theta_1}, \dots, \frac{1}{\theta_n}\right) U^{-1}, \quad U = [\mathbf{y}^1 - \mathbf{x}^0, \dots, \mathbf{y}^n - \mathbf{x}^0].$$

Then the inequality

$$(23) \quad \pi(\mathbf{x} - \mathbf{x}^0) > 1$$

provides a  $\gamma$ -valid cut for  $(f, D)$ , i.e., any  $\mathbf{x}$  having objective value  $f(\mathbf{x}) < \gamma$  must satisfy (23). In other words, if (23) has no solution in  $D$ ,  $\mathbf{x}^0$  must be a

global minimum. Note that 1)  $\theta_i \geq 1$ ; so  $S_{\text{px}}$  contains  $\mathbf{x}^0$  and all its neighbor vertices; 2) the larger the  $\theta_i$ , the deeper the cuts. Following is the original pure convexity cutting algorithm based on the above idea.

**Cutting Algorithm (Algorithm V.1., Chapter V, [10])**

**Initialization**

Search for a vertex  $\mathbf{x}^0$  which is a local minimizer of  $f(\mathbf{x})$ . Set  $\gamma = f(\mathbf{x}^0)$ ,  $D_0 = D$ .

**Iteration  $i = 0, 1, \dots$**

1. At  $\mathbf{x}^i$  construct a  $\gamma$ -valid cut  $\pi^i$  for  $(f, D_i)$ .
2. Solve the linear program

$$(24) \quad \max \pi^i(\mathbf{x} - \mathbf{x}^i) \quad \text{s.t. } \mathbf{x} \in D_i.$$

Let  $\omega^i$  be a basic optimum of this LP. If  $\pi^i(\omega^i - \mathbf{x}^i) \leq 1$ , then stop:  $\mathbf{x}^0$  is a global minimum. Otherwise, go to step 3.

3. Let  $D_{i+1} = D_i \cap \{\mathbf{x} : \pi^i(\mathbf{x} - \mathbf{x}^i) \geq 1\}$ . Starting from  $\omega^i$  find a vertex  $\mathbf{x}^{i+1}$  of  $D_{i+1}$  which is a local minimum of  $f(\mathbf{x})$  over  $D_{i+1}$ . If  $f(\mathbf{x}^{i+1}) \geq \gamma$ , then go to iteration  $i + 1$ . Otherwise, set  $\gamma \leftarrow f(\mathbf{x}^{i+1})$ ,  $\mathbf{x}^0 \leftarrow \mathbf{x}^{i+1}$ ,  $D_0 \leftarrow D_{i+1}$ , and go to iteration 0.

**Theorem 1** (Theorem V.2, [10]) *If the sequence  $\{\pi^i\}$  is bounded, then the above cutting algorithm is finite.*

### 3.2 The adapted Tuy's cutting algorithm for MSSC.

To adapt Tuy's cuts to (5). We include constraints (2c) in (5) to ensure that it is a concave program (Proposition 1), although we have proved that these constraints are redundant for finding a local solution (Corollary 1). In this section, we will first show how we find a local minimum. Our algorithm searches for a local minimum of (4), since it is stronger than that of (5), as is discussed before. Then, we will describe how we construct the concavity cuts. Finally, we will prove the finite convergence of our algorithm and compare it with the K-means algorithm.

#### 3.2.1 Finding a local minimum.

To find a local minimum of (4), we use pivot: moving from one vertex of the feasible domain to an adjacent one that can mostly decrease the total within-group sum-of-squared error based on (19), until to a vertex complying with (20).

At  $r$ th iteration,

do **Loop** until (20) is satisfied for all  $l = 1, \dots, n$ .

**Loop** For  $l = 1, \dots, n$ :

Assume  $\mathbf{a}_l$  is assigned to  $j$ th cluster. When  $\sum_{i=1}^n x_{ij} > 1$ , let

$$f_l = \min_{q \neq j} \min_{q \in D_r} \frac{\sum_{i=1}^n x_{iq}}{\sum_{i=1}^n x_{iq+1}} \|\mathbf{v}_{lq}\|_2^2. \quad \text{If}$$

$f_l < \frac{\sum_{i=1}^n x_{ij}}{\sum_{i=1}^n x_{ij-1}} \|\mathbf{v}_{lj}\|_2^2$ , move  $\mathbf{a}_l$  to cluster  $l$  and update  $s(X)$  by (19).

Note that we only search for a local minimum of (4). The number of vertices of the domain of (4) is finite and the objective value of (4a) is strictly decreased after each pivot. Hence, the number of pivots for finding a local minimum is finite.

#### 3.2.2 Construction of the cutting planes.

Once we find a local minimum, we need to add some cut to the constraints set. At  $r$ th iteration, let  $X^0 \in \mathbb{R}^{n \times k}$  be a local optimal solution to (4) in  $D_r$  and  $\gamma$  be the smallest total within-group sum-of-squared errors obtained from the previous iterations. Next, we will give details on how we form (22), including the construction of  $U$  and  $\theta_i$  for (22), although the feasible region of (4) doesn't have full dimension — each vertex is adjacent to only  $n \times (k - 1)$  other vertices.

##### 1) Adjacent vertices.

We give  $n \times k$  adjacent vertices to  $X^0$  below.

Let  $E_{i,j}$  denote the matrix whose  $(i, j)$  entry is 1, the other entries are 0; and let  $E_{(i,\cdot)}$  denote the matrix whose  $i$ th row are 1's, the remaining entries are 0. The orders of  $E_{i,j}$  and  $E_{(i,\cdot)}$  will be clear from the context. For  $l = 1, \dots, n$ , assume  $\mathbf{a}_l$  is assigned to cluster  $l_j$ . Let  $Y^{l,i}$  ( $i = 1, \dots, k; i \neq l_j$ ) denote the matrix different from  $X^0$  only by the assignment of  $\mathbf{a}_l$  to cluster  $i$ . Choose  $1 \leq l_p \leq k, l_p \neq l_j$ . And let

$Y^{l,l_j}$  denote the matrix different from  $X^0$  only in its  $(l, l_p)$  entry being 1 as well, i.e.,

$$Y^{l,i} = \begin{cases} X^0 - E_{l,l_j} + E_{l,i} & i \neq l_j \\ X^0 + E_{l,l_p} & i = l_j \end{cases}.$$

Then  $Y^{l,i}$  ( $l = 1, \dots, n; i = 1, \dots, k, i \neq l_j$ ) are  $n \times (k-1)$  adjacent vertices of  $X^0$  in the feasible domain of (4). We form the vector  $\mathbf{x}^0$  by stacking all the columns of  $X^0$  together. Similarly, we form the vectors  $\mathbf{y}^{l,i}$ . It is not hard to see that  $U = [\mathbf{y}^{1,1} - \mathbf{x}^0, \dots, \mathbf{y}^{1,k} - \mathbf{x}^0, \dots, \mathbf{y}^{n,k} - \mathbf{x}^0]$  has full rank. Let  $I$  represent the identity matrix. It is straightforward to verify that the corresponding  $U^{-1}$  of (23) is a block diagonal matrix with  $l$ th block being  $I + E_{(l_j, \cdot)} - E_{(l_m, \cdot)} - E_{l_j, l_j}$ .

Because  $Y^{l,l_j}$  is not feasible to (5), part of the simplex  $\text{conv}\{\mathbf{x}^0, \mathbf{y}^{1,1}, \dots, \mathbf{y}^{1,k}, \dots, \mathbf{y}^{n,k}\}$  lies outside the feasible region of (5); nevertheless, the concavity cut can exclude some part of the feasible region of (4).

## 2) The cutting plane.

Next, we will determine the  $\theta_i$ 's of (22), and subsequently the cutting plane  $\pi$ .

For  $l = 1, \dots, n$ :  $\theta^{l,l_j} = +\infty$ ; when  $m \neq l_j$ , by Proposition 2,  $\theta^{l,m}$  is a solution to the problem below. (25)

$$\begin{aligned} & \max t \\ & \text{s.t. } 0 \leq t \leq \sum_{i=1}^n x_{i,l_j}^0, \\ & s(X^0) - \frac{\sum_{i=1}^n x_{i,l_j}^0 t}{\sum_{i=1}^n x_{i,l_j}^0 - t} \|\mathbf{v}_{l,l_j}\|_2^2 + \frac{\sum_{i=1}^n x_{i,m}^0 t}{\sum_{i=1}^n x_{i,m}^0 + t} \|\mathbf{v}_{l,m}\|_2^2 \geq \gamma. \end{aligned}$$

Denote the number of points in the  $j$ th cluster as  $N_j \stackrel{\text{def}}{=} \sum_{i=1}^n x_{ij}^0$ . Next, we will solve (25).

Solving the second inequality in (25), we get  $t \leq t^*$ , where

$$t^* = - \frac{(s(X^0) - \gamma)(N_m - N_{l_j}) + N_m N_{l_j} \|\mathbf{v}_{l,l_j}\|_2^2 - \|\mathbf{v}_{l,m}\|_2^2 - \omega}{2 s(X^0) - \gamma + N_{l_j} \|\mathbf{v}_{l,l_j}\|_2^2 + N_m \|\mathbf{v}_{l,m}\|_2^2},$$

$$\text{where } \omega = \left[ (s(X^0) - \gamma)(N_m + N_{l_j}) + N_m N_{l_j} (\|\mathbf{v}_{l,l_j}\|_2^2 - \|\mathbf{v}_{l,m}\|_2^2) \right]^2 + 4(s(X^0) - \gamma) N_{l_j} N_m (N_{l_j} + N_m) \|\mathbf{v}_{l,m}\|_2^2.$$

Since  $s(X^0) \leq \gamma$  and  $X^0$  is a local minimum of (4), by (19),  $t^* \geq 1$ . In view of the first constraint in (25), we set

$$\theta^{l,m} = \min \{N_{l_j}, t^*\}, \quad (m \neq l_j).$$

Observe that when  $s(X^0) = \gamma$ , since  $X^0$  is a local minimum of (4), by Lemma 1,  $\|\mathbf{v}_{l,m}\|_2 > \|\mathbf{v}_{l,l_j}\|_2$ ; hence

$$t^* = \frac{N_{l_j} N_m (\|\mathbf{v}_{l,m}\|_2^2 - \|\mathbf{v}_{l,l_j}\|_2^2)}{N_{l_j} \|\mathbf{v}_{l,l_j}\|_2^2 + N_m \|\mathbf{v}_{l,m}\|_2^2} \leq N_{l_j}.$$

Therefore, in this case,  $\theta^{l,m} = t^*$ .

It follows that the coefficients of (24) are

$$\pi^{l,i} = \begin{cases} \frac{1}{\theta^{l,i}} - \frac{1}{\theta^{l,l_p}} & i \neq l_j \\ -\frac{1}{\theta^{l,l_p}} & i = l_j \end{cases}, \quad \pi_{\mathbf{x}^0} = - \sum_{l=1}^n \frac{1}{\theta^{l,l_p}}.$$

## 3) Solving the LP subproblems.

Without the constraints (2c), the LP for the first cut (24) is a knapsack problem, whose solution is trivial and is integer. If the solution doesn't satisfy (2c), by splitting some clusters as is discussed in Corollary 1, one can find a vertex that satisfies (2c)

The solutions to the succeeding LPs (24) are not necessarily integers. From a fractional solution to (24), one can move to a vertex using the procedure in Lemma 1. If the vertex is not in the remaining feasible region  $D_i$ , we use breadth-first search to find a vertex that satisfies all the cutting plane constraints. When pivoting to a local minimum, we also skip those vertices that are not in  $D_i$ . This is different from the pivoting in K-means algorithm. This ensures that no vertices in cut-out region will be revisited.

## 3.3 Finite convergence of the algorithm.

The simplex Spx in our algorithm is centered at a local minimum of (4); so each concavity cut eliminates at least one vertex of (4). In addition, the number of vertices of (4) is finite. Therefore, the number of cuts is finite. From this along with the previous argument that only finite pivots are needed to reach a local minimum of (4), we conclude that our method can find a global minimum of the MSSC problem in finite steps.

**Remark 3** *It is possible to get a good solution for the MSSC problem by multiple runs of K-means algorithm with random restart. However, it is difficult*

to guess the new starting points. Without trying all the vertices, a global minimum cannot be guaranteed. An advantage of Tuy’s convexity cut to enumeration is that each convexity cut may eliminate a certain number of vertices.

## 4 Numerical Examples

We have implemented the above cutting algorithm in C with the LP for cutting plane solved by CPLEX 8.0.

Minimizing a concave function over a polytope is NP-hard (see for example [17]). In the worst case, Tuy’s cutting method needs to enumerate all the vertices in a polytope; hence the worst-case complexity of Tuy’s method for a general concave program is exponential. At this moment, we don’t know whether the complexity of our algorithm can be reduced based on some special properties of MSSC. In addition, although the computational cost of each iteration is very low — solving an LP and pivoting, as more and more cuts are added, the size of the LP subproblem might be very large, which may require large amount of memory. From our preliminary numerical experience, we notice that the objective value improves significantly in the first several cuts, but very slowly in the later cuts. To maintain a tradeoff between the difficulty of finding a global minimum of the MSSC problem and the practical efficiency of the algorithm, we specify that our computer program terminates iff one of the following conditions is satisfied:

1. a global optimum is found;
2. the objective cannot be improved after 5 succeeding cuts;
3. the total number of cuts exceeds 20.

To give an initial solution, we assign the first  $k$  entities to the  $k$  clusters. Each of the remaining entities are assigned to the cluster whose centroid is closest to it. The initial sum-of-squared error is calculated from this assignment. Next, we run the K-means algorithm to find a local optimum of (5), from which we further pivot to a local minimum of (4) to start the cutting algorithm.

The figures below give some numerical examples. We use  $x$ -axis to represent the number of clusters— $k$ ,

and  $y$ -axis to denote the total within-group sum-of-squared errors. Each group of vertical bars represents various sum-of-squared errors for a certain  $k$ . The first bar in a group denotes the initial sum-of-squared error; the second one is that when each entity is assigned to a cluster whose centroid is closest to it; the third one reports the sum-of-squared error when (20) is satisfied for each entity; the fourth one is that resulting from the cutting algorithm. The fifth bar of the first two examples represents the best known objective value taken from [2].

### 1. *The Iris plants database.*

Our first example is from [4]. It has 150 samples each having 4 attributes.

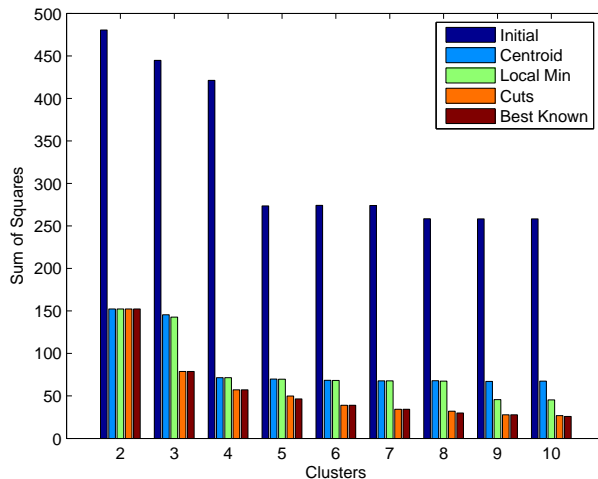


Figure 1: Iris Plants Database

We have tested for  $k = 2, \dots, 10$ . Of the 9 cases, the cutting algorithm hits the best solution in 6 cases, and outperforms the k-means algorithm as well as the initial local minima in 8 cases.

### 2. *The Ruspini data set.*

This data set, consisting of 75 observations on 2 variables, is from [18].

For this data set, the cutting algorithm finds the global solution in 4 out of total 9 cases. It improves the K-means results in 8 cases; and get

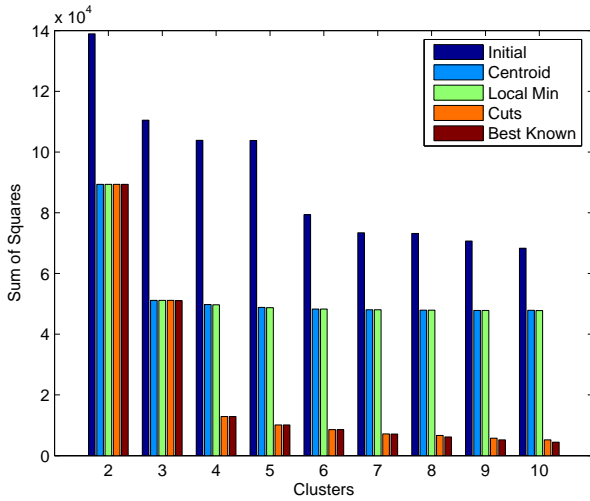


Figure 2: Ruspini data

a better solution than the initial local minimum in 7 cases.

By Corollary 1, the optimum within-group sum-of-squared error decreases with the number of cluster  $k$ . In the following examples, which are taken from the UCI Machine Learning Repository [16], we use the solution from the  $k$ -clustering as the starting point for the  $k + 1$  clustering ( $k = 2, \dots$ ).

1. *The Boston housing data set.*

This data set is from [8] concerning housing values in suburbs of Boston. It has 506 samples each having 13 attributes.

Among all the 9 cases  $k = 2, \dots, 10$ , the cutting algorithm over-performs the k-means algorithm in 7 cases and gets a better solution than the original local minimum of the integer programming formula (4) in 6 cases.

2. *The spam E-mail database*

This dataset is created by M. Hopkins et al. at Hewlett-Packard Labs in June-July 1999 for spam filter. It has 4601 samples, 57 attributes (we remove the last class attribute which denotes whether the e-mail was considered spam or not).

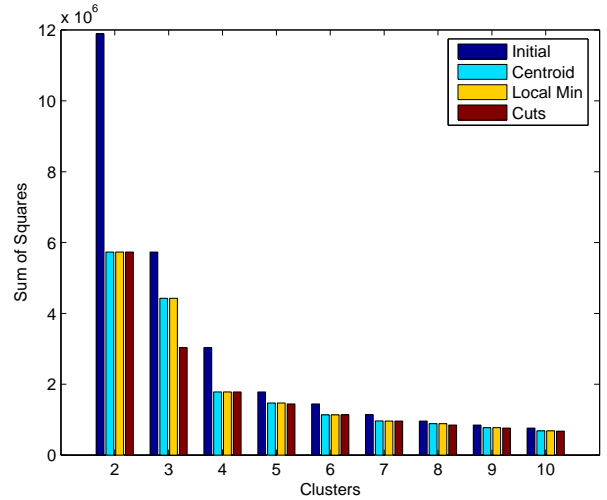


Figure 3: Boston Housing

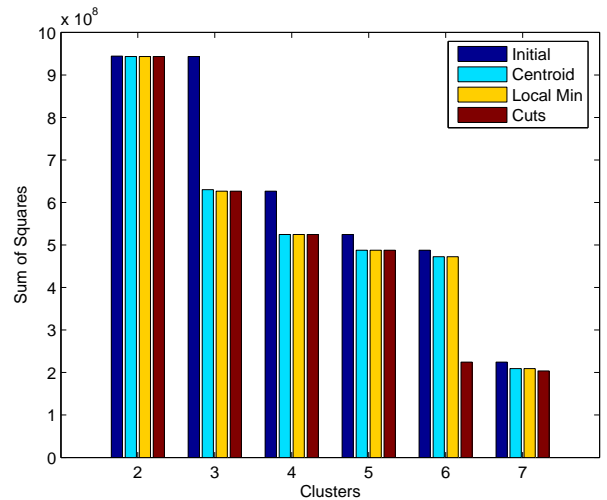


Figure 4: Spam E-mail

Of the 6 cases  $k = 2, \dots, 7$ , the cutting algorithm finds a better solution than that by the K-means algorithm in 4 cases, the objective value is reduced from the initial local minimum of (4) in 2 instances.

For all the examples, the computer program terminated due to no improvement in objective value after 5 succeeding cuts. All of them terminated within 25 seconds. However, the results are satisfactory. Observe that in many cases, the solution obtained by the K-means algorithm is worse than that by the local minimum criterion (20).

## 5 Conclusion

In this paper, we have proved that the minimum sum-of-squared error clustering problem can be formulated as a concave minimization problem whose every local optimum solution is integer. We have characterized the local optimality of the continuous optimization model (5) and compared it with that of the discrete one (4). These properties can help us find a better solution. We have adapted the globally convergent Tuy's convexity cut to the concave optimization problem derived from MSSC. Preliminary numerical examples demonstrate that our method outperforms the popular K-means algorithm in the quality of the solution without a big increase in the running time.

## References

- [1] Werner Dinkelbach. On nonlinear fractional programming. *Management Sci.*, 13:492–498, 1967.
- [2] O. Du Merle, P. Hansen, B. Jaumard, and N. Mladenović. An interior point algorithm for minimum sum-of-squares clustering. *SIAM J. Sci. Comput.*, 21(4):1485–1505 (electronic), 1999/00.
- [3] James E. Falk and Karla R. Hoffman. Successive underestimation method for concave minimization problem. *Math. Oper. Res.*, 1(3):251–259, Aug. 1976.
- [4] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annual of Eugenics*, 7:179–188, 1936.
- [5] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768, 1965.
- [6] A. D. Gordon and J. T. Henderson. An algorithm for euclidean sum of squares classification. *Biometrics*, 33(2):355–362, Jun. 1977.
- [7] Pierre Hansen and Brigitte Jaumard. Cluster analysis and mathematical programming. *Math. Programming*, 79(1-3, Ser. B):191–215, 1997.
- [8] D. Harrison and D. L. Rubinfeld. Hedonic prices and the demand for clean air. *J. Environ. Economics & Management*, 5:81–102, 1978.
- [9] Reiner Horst. An algorithm for nonconvex programming problems. *Math. Programming*, 10(3):312–321, 1976.
- [10] Reiner Horst and Hoang Tuy. *Global optimization*. Springer-Verlag, Berlin, 1993.
- [11] Stephen E. Jacobsen. Convergence of a Tuy-type algorithm for concave minimization subject to linear inequality constraints. *Appl. Math. Optim.*, 7(1):1–9, 1981.
- [12] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
- [13] R. C. Jancey. Multidimensional group analysis. *Australian J. Botany*, 14:127–130, 1966.
- [14] O. L. Mangasarian. Mathematical programming in data mining. *Data Min. Knowl. Discov.*, 1(2):183–201, 1997.
- [15] J. McQueen. Some methods for classification and analysis of multivariate observations. *Computer and Chemistry*, 4:257–272, 1967.
- [16] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases. Technical report,

University of California, Department of Information and Computer Science, Irvine, CA, 1994.  
<http://www.ics.uci.edu/~mlearn/MLRepository.html>.

- [17] P. M. Pardalos and J. B. Rosen. Methods for global concave minimization: a bibliographic survey. *SIAM Rev.*, 28(3):367–379, 1986.
- [18] E. H. Ruspini. Numerical methods for fuzzy clustering. *Inform. Sci.*, 2(3):319–350, 1970.
- [19] Shokri Z Selim and M. A. Ismail. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):81–87, Jan. 1984.
- [20] H. Tuy. Concave programming under linear constraints. *Soviet Mathematics*, 5:1437–1440, 1964.
- [21] Hoang Tuy, A. M. Bagirov, and A. M. Rubinov. Clustering via d.c. optimization. In *Advances in convex analysis and global optimization (Pythagorion, 2000)*, volume 54 of *Nonconvex Optim. Appl.*, pages 221–234. Kluwer Acad. Publ., Dordrecht, 2001.