A Scalable Deterministic Global Optimization Algorithm for Clustering Problems

Kaixun Hua [1] **Mingfei Shi** [1] **Yankai Cao** [1]

Abstract

The minimum sum-of-squares clustering (MSSC) task, which can be treated as a Mixed Integer Second Order Cone Programming (MISOCP) problem, is rarely investigated in the literature through deterministic optimization to find its global optimal value. In this paper, we modelled the MSSC task as a two-stage optimization problem and proposed a tailed reduced-space branch and bound (BB) algorithm. We designed several approaches to construct lower and upper bounds at each node in the BB scheme, including a scenario grouping based Lagrangian decomposition approach. One key advantage of this reduced-space algorithm is that it only needs to perform branching on the centers of clusters to guarantee convergence, and the size of centers is independent of the number of data samples. Moreover, the lower bounds can be computed by solving small-scale sample subproblems, and upper bounds can be obtained trivially. These two properties enable our algorithm easy to be paralleled and can be scalable to the dataset with up to 200,000 samples for finding a global ϵ-optimal solution of the MSSC task. We performed numerical experiments on both synthetic and real-world datasets and compared our proposed algorithms with the off-the-shelf global optimal solvers and classical local optimal algorithms. The results reveal a strong performance and scalability of our algorithm.

1. Introduction

Clustering is the prototypical unsupervised learning activity that identifies cohesive and well-differentiated groups of records in data (Jain, 2010). The target to get a clustering result can always be treated as an optimization problem. The different definitions of cost function induce different types of clustering algorithms. In this paper, we focus on a fundamental target of the clustering problems that minimize the within-cluster sum-of-squared-error or in short the minimum sum-of-squares criteria. It tends to minimize the distance between points to their corresponding cluster centers to achieve the best cohesion and separation of the resulted clusters (Späth, 1980). There are many heuristic methods proposed for solving the minimum sum-of-squares clustering (MSSC) task. For instance, the k-means clustering algorithm (Lloyd, 1982) provides a coordinate descent based method to produce a result for the MSSC task. However, due to the non-convexity of the MSSC objectives, the classic k-means algorithm is sensitive to the initialization and easy to fall under the local minimum (Xu & Lange, 2019). To overcome this limitation, several modifications to the classical k-means clustering algorithm have been proposed in order to obtain the global optimal solutions for the MSSC task (Likas et al., 2003; Tzortzis & Likas, 2014; Xu & Lange, 2019; Agarap & Azcarraga, 2020). However, none of these works provide the deterministic guarantee of locating the global minimum. Investigating directly on the global solution of the MSSC problem is still in deficiency.

One direction to solve the MSSC problem to global optimality deterministically starts from the work of (Peng & Wei, 2007) who modeled the MSSC problem as a 0-1 semidefinite programming (SDP) problem. (Aloise & Hansen, 2009) applied this discovery and proposed a branch-and-cut SDP algorithm for MSSC problem. Their algorithm can solve problems with dataset up to 200 samples.

Another direction applies the column generation to MSSC problem. Indeed, a column generation based clustering algorithm proposed by (Du Merle et al., 1999) is, for the first time, capable to solve the MSSC problem with median size of sample (100-200 samples). This method was further improved by (Aloise et al., 2012) which used a geometric-based approach to solve the auxiliary problem. Their work improved the solvable problem size to 2300, which is currently the state of the art. However, column generation method could face the exponential growth of the size of master problem with the growth of the number of iterations. Therefore, the method cannot scale further to problems with even larger datasets. Moreover, Aloise's method is more

[1]Department of Chemical and Biological Engineering, University of British Columbia, Vancouver, British Columbia, Canada. Correspondence to: Yankai Cao <yankai.cao@ubc.ca>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

suitable for the problems when the ratio between sample size and number of clusters is particularly small (e.g. ≤ 10). Therefore, this approach is also less efficient for problems of small number of clusters but large size of dataset.

It is well-known that clustering problems can be reformulated as mixed integer programming problems (Freed & Glover, 1983; Sağlam et al., 2006; Komodakis et al., 2008). Specifically, the MSSC task can be treated as a Mixed Integer Second Order Cone Programmin (MISOCP) problem. Branch and bound (BB) scheme is the most widely used algorithm for solving these optimization problems to global optimality and is well implemented in several popular solving packages, like BARON (Tawarmalani & Sahinidis, 2005), ANTIGONE (Misener & Floudas, 2014), and SCIP (Gamrath et al., 2020). The BB paradigm depends on the efficient evaluation of lower and upper bounds of the optimal solution. It is able to reduce the gap between lower and upper bounds based on two key principles, that is to partition the search spaces into smaller regions that can be solved recursively (e.g. branching), and to prune the search regions that it can prove will not contain an optimal solution (e.g. bounding) (Morrison et al., 2016). Nevertheless, the direct application of the BB scheme does not scale well with the data size because branching may need to be performed on all variables to guarantee convergence and the number of variables increases linearly as the number of samples. Thus, it is almost infeasible to find the global optimum of the MSSC tasks using the classical BB scheme that implemented in those off-the-shelf solvers, even if the data size is moderate (e.g. 100 samples).

The first branch and bound clustering algorithm for MSSC problem was proposed by (Koontz et al., 1975) and further developed by (Diehr, 1985). They use the solution of the MSSC problem from a subset of the main dataset to generate a tighter lower bound. They discover that if the dataset is separated into two subsets, then the sum of the optimal values of the two MSSC subproblem forms a lower bound of the original problem. (Brusco, 2006) proposed the repetitive branch and bound algorithm (RBBA) which effectively reorder the samples and solving a sequence of subproblems with increasing size. RBBA can solve problem with datasets up to 240 samples. (Sherali & Desai, 2005) proposed a BB scheme for MSSC problem which applied the reformulation-linearization-technique (RLT) to form a tighter lower bound. Their algorithm is claimed to able to solve datasets up to 1000 samples. However, their work was questioned that can only solve problem with size lower than 20 samples (Aloise & Hansen, 2011).

To handle issues from the basic BB procedure, in the paper, we start from a new direction that reformulates the MSSC task as a two-stage optimization problem. This is because several approaches have been proposed in the stochastic programming community focusing on improving the scalability of global search by exploiting the structure of two-stage problems. These methods include generalized Benders decomposition (Geoffrion, 1972), nonconvex Benders decomposition (Li et al., 2011; Li & Grossmann, 2019), and Lagrangian relaxation (Khajavirad & Michalek, 2009; CarøE & Schultz, 1999; Karuppiah & Grossmann, 2008). In this paper, we adopted the fundamental work of (Cao & Zavala, 2019) on the reduced-space BB scheme for two-stage optimization problems and tailed it for the MSSC task. The novelty of the approach proposed in (Cao & Zavala, 2019) is that it guarantees the convergence to the global optimum by only branching on first-stage variables (branching on second-stage variables is performed implicitly during the computation of bounds). In the context of the clustering task, the centers of clusters are regarded as the first-stage variables, while the binary variables indicating the class of each data sample are treated as the second-stage variables. It implies that the number of variables need to be partitioned on is independent of the dataset's cardinality.

Our Contributions. In this paper, we proposed a scalable deterministic global optimization algorithm for the minimum sum-of-squared clustering task. Specifically, we contribute the following benefits:

- We propose a tailed reduced space branch-and-bound clustering algorithm for the MSSC task, which only needs to branch on the centers of clusters. We also present a convergence proof of reaching the global ϵ-optimal solution using the proposed algorithm.
- We design several approaches to construct lower and upper bounds at each node in the BB scheme. The closed-form solutions to both basic lower bounding problems and basic upper bound problems are derived. Therefore, basic lower and upper bounds can be computed without solving any optimization problem. In contrast, the lower and upper bounding problems proposed in (Cao & Zavala, 2019) are computationally much more expensive to solve because each individual sup-problem is a mixed-integer nonlinear programming (MINLP) problem that needs to be solved to global optimality. Moreover, we also proposed approaches to construct tighter lower bounds than that obtained using basic lower bounding problems, such as scenario based sample grouping and Lagrangian decomposition.
- We present an open-source implementation of the proposed algorithm in `Julia`. Our algorithm and implementation enlarge the application of finding the global optimum of MSSC tasks to the scale of datasets with up to 210,000 samples (200 cores, under 3% optimality gap within the runtime of 4 hours), which is **100 times larger** than current state-of-the-art work (Aloise et al., 2012). Notably, numerical experiments on the several real-world

datasets show that our implementation can converge to a small gap ($< 0.1\%$) under 12 hours in serial or 1 hour in parallel with small number of clusters, while current state of the art need a long time (≥ 50 hours) or cannot solve. By obtaining the global clustering solution, we are also able to provide an explainable benchmark on how well the traditional k-means clustering algorithm performs.

2. Reduced-space Branch and Bound Scheme

Given a dataset $X = \{x_1, \ldots, x_S\} \in \mathbb{R}^{d\times S}$ with S samples and d attributes, a MSSC task aims to find a set of K clusters, that can minimize the Sum of Squared Errors (SSE), which is defined as:

$$\sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}} b_{s,k}||x_s - \mu_k||^2 \tag{1}$$

where $s \in \mathcal{S} := \{1, \cdots, S\}$ is the data sample set, $k \in \mathcal{K} = \{1, \cdots, K\}$ is the cluster set, $\mu := [\mu_1, \cdots, \mu_K]$ represents the center of each cluster, $b_{s,k} \in \{0, 1\}$ is equal to 1 if x_s belongs to the kth clusters, and 0, otherwise.

The k-means clustering problem can be treated as an optimization problem of the following form:

$$\min_{\mu,d,b} \sum_{s\in\mathcal{S}} d_{s,*} \tag{2a}$$
$$\text{s.t.} \quad -N(1-b_{s,k}) \leq d_{s,*} - d_{s,k} \leq N(1-b_{s,k}) \tag{2b}$$
$$d_{s,k} \geq ||x_s - \mu_k||^2 \tag{2c}$$
$$\sum_{k\in\mathcal{K}} b_{s,k} = 1 \tag{2d}$$
$$b_{s,k} \in \{0, 1\} \tag{2e}$$
$$s \in \mathcal{S}, k \in \mathcal{K} \tag{2f}$$

where $d_{s,k}$ denotes the distance between x_s and μ_k, $d_{s,*}$ represents the distance between x_s and the center of its cluster, and N is an arbitrary large value. We also define $d_s := [d_{s,1}, \cdots, d_{s,K}, d_{s,*}]$, $d := [d_1, \cdots, d_S]$, $b_s := [b_{s,1}, \cdots, b_{s,K}]$, and $b := [b_1, \cdots, b_S]$. Constraint 2d ensures that sample x_s is assigned to only one cluster, and Constraint 2b use a big-M formulation to guarantee that $d_{s,*} = d_{s,k}$ if $b_{s,k} = 1$. Problem 2[1] is a mixed-integer second order cone programming (MISOCP) problem and can be solved by off-the-shelf solvers such as Gurobi (Optimization, 2014) and CPLEX (Cplex, 2020). However, when the number of samples increases to a moderate value (e.g. $S = 100$), the problem quickly becomes intractable using these off-the-shelf solvers.

[1]In implementation, we also add the symmetric breaking constraint $\mu_{1,1} \leq \mu_{2,1} \leq \cdots \leq \mu_{k,1}$ to accelerate the solving process. However, to simplify the notation, we will not mention this constraint in the rest of the paper.

2.1. Two-stage Optimization Formulation

Problem 2 can be reformulated as a two-stage optimization problem of the form:

$$z = \min_{\mu\in M_0} \sum_{s\in\mathcal{S}} Q_s(\mu). \tag{3}$$

Here, μ are the so-called first-stage variables, and $Q_s(\mu)$ is the optimal value of the second-stage problem:

$$\begin{aligned} Q_s(\mu) = \min_{d_s, b_s} \ & d_{s,*} \\ \text{s.t.} \ & -N(1-b_{s,k}) \leq d_{s,*} - d_{s,k} \leq N(1-b_{s,k}) \\ & d_{s,k} \geq ||x_s - \mu_k||^2 \\ & \sum_{k\in\mathcal{K}} b_{s,k} = 1 \\ & b_{s,k} \in \{0, 1\} \\ & k \in \mathcal{K} \end{aligned} \tag{4}$$

Where d_s and b_s are the so-called second-stage variables. The closed set $M_0 := \{\mu \mid \mu^l \leq \mu \leq \mu^u\}$ in Equation 3 represents the bounds of centers inferred from data, that is, $\mu^l_{k,i} = \min_s X_{s,i}$, $\mu^u_{k,i} = \max_s X_{s,i}$, $\forall k \in \mathcal{K}$, $i \in \{1, \cdots, d\}$. Note that the introduction of $\mu \in M_0$ does not affect the optimal solution and is used to facilitate the discussion of BB algorithm. We denote $relint(\mathcal{C})$ and $\delta(\mathcal{C})$ as the relative interior and the diameter of set $\mathcal{C}$, respectively. Throughout this paper, the diameter of the box set M_0 is $\delta(M_0) = ||\mu^u - \mu^l||_\infty$.

It can be shown that the closed-form solution to the second-stage problem is $Q_s(\mu) = \min_k ||x_s - \mu_k||^2$. Since $Q_s(\mu)$ is the minimum of a finite number of continuous functions, we have:

Lemma 1. *$Q_s(\mu)$ is continuous on μ for all $s \in \mathcal{S}$.*

Because of the compactness of M_0 and Lemma 1, the clustering Problem 3 can attain its minimum according to the generalized Weierstrass theorem.

When solving the Problem 3 with BB algorithm, we solve the following problem at each node with respect to the partition set $M \subseteq M_0$:

$$z(M) = \min_{\mu\in M} \sum_{s\in\mathcal{S}} Q_s(\mu) \tag{5}$$

The Problem 5 is referred as the *primal node problem*. By replicating the center μ for each sample, and enforcing the non-anticipativity constraints 6b, we give the lifted Problem 5 as follow:

$$\min_{\mu_s\in M} \quad \sum_{s\in\mathcal{S}} Q_s(\mu_s) \tag{6a}$$
$$\text{s.t.} \quad \mu_s = \mu_{s+1}, s \in \{1, \cdots, S-1\} \tag{6b}$$

Formulations 5, 6 are equivalent.

2.2. Lower Bounds

In this section, we describe methods to generate lower bounds for the primal node Problem 5 and improvements to generate tighter bounds.

2.2.1. Basic Lower Bounding Problem

By relaxing the non-anticipativity constraints 6b, we can obtain the following lower bounding problem:

$$\beta(M) := \min_{\mu_s \in M} \sum_{s \in \mathcal{S}} Q_s(\mu_s) \tag{7}$$

This problem can be easily decomposed into S subproblems:

$$\beta_s(M) = \min_{\mu \in M} Q_s(\mu). \tag{8}$$

Here, $\beta(M) = \sum_{s \in \mathcal{S}} \beta_s(M)$. Since the non-anticipativity constraints are relaxed, the feasible region of 7 is a superset of the feasible region of Problem 5. Thus $\beta(M) \leq z(M)$ is always satisfied.

Because the closed-form solution to the second-stage problem is $Q_s(\mu) = \min_k ||x_s - \mu_k||^2$, we have:

$$\beta_s(M) = \min_k \min_{\mu_k \in M_k} ||x_s - \mu_k||^2, \tag{9}$$

where $M_k := \{\mu_k \mid \mu_k^l \leq \mu_k \leq \mu_k^u\}$. Therefore, subproblem $\beta_s(M)$ can be further decomposed into K subsubproblems:

$$\beta_{s,k}(M_k) = \min_{\mu_k \in M_k} ||x_s - \mu_k||^2. \tag{10}$$

Here $\beta_s(M) = \min_k \beta_{s,k}(M_k)$. Problem 10 is a Quadratic Programming (QP) problem and the closed-form solution to this problem can be derived: $\mu_{k,i} = \text{mid}\{\mu_{k,i}^l, x_{s,i}, \mu_{k,i}^u\}$, $\forall i \in \{1, \cdots, d\}$.

2.2.2. Scenario-based Sample Grouping

Although the lower bound generated using the basic lower bound problems is enough to guarantee convergence as shown in Section 3, it might not be very tight because decomposition is performed on each individual sample. Specifically, for the root node with $M = M_0$, since each subproblem only contains one data sample x_s, it is easy to verify that $\mu_k = x_s$ is the global optimal solution to Problem 10 and the corresponding $\beta(M_0)$ equal to 0. Although this lower bound can be improved as M is partitioned into smaller sets during the BB scheme, we seek to generate tighter bounds for each node.

One approach to achieve this goal is scenario-based sample grouping, that is to assign more than one sample (i.e. a group of samples) into each subproblem. Specifically, we divide the set of samples $\mathcal{S}$ into G groups $\{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_G\}$ and define the group set $\mathcal{G} = \{1, \cdots, G\}$, where $\mathcal{S}_g$ represents the subset of $\mathcal{S}$ containing samples belonging to group g, such that $\bigcup_{g=1}^{G} \mathcal{S}_g = \mathcal{S}$ and $\mathcal{S}_i \cap \mathcal{S}_g = \emptyset, \forall i, g \in \mathcal{G}, i \neq g$. Considering the lifted problem, instead of replicating the center for each sample, we can replicate it for each group and reformulate Problem 6 as follow:

$$\min_{\mu_g \in M} \quad \sum_{g \in \mathcal{G}} Q_g(\mu_g) \tag{11a}$$

$$\text{s.t.} \quad \mu_g = \mu_{g+1}, g \in \{1, \cdots, G-1\}. \tag{11b}$$

Here, $Q_g(\mu_g) = \sum_{s \in \mathcal{S}_g} Q_s(\mu_g)$. By relaxing the constraints 11b, we can obtain the following sample grouped lower bounding problem:

$$\beta^{SG}(M) := \min_{\mu_g \in M} \sum_{g \in \mathcal{G}} Q_g(\mu_g). \tag{12}$$

This problem can be easily decomposed into G subproblems:

$$\beta_g^{SG}(M) := \min_{\mu_g \in M} Q_g(\mu_g), \tag{13}$$

or, equivalently,

$$\begin{aligned} \beta_g^{SG}(M) := \min_{\mu_g, d_{\mathcal{S}_g}, b_{\mathcal{S}_g}} & \sum_{s \in \mathcal{S}_g} d_{s,*} \\ \text{s.t.} \quad & -N(1 - b_{s,k}) \leq d_{s,*} - d_{s,k} \leq N(1 - b_{s,k}) \\ & d_{s,k} \geq ||x_s - \mu_k||^2 \\ & \sum_{k \in \mathcal{K}} b_{s,k} = 1 \\ & b_{s,k} \in \{0, 1\} \\ & s \in \mathcal{S}_g, k \in \mathcal{K} \end{aligned} \tag{14}$$

with $\beta^{SG}(M) = \sum_{g \in \mathcal{G}} \beta_g^{SG}(M)$.

Compared with the basic lower bounding problems 7, non-anticipativity constraints within each group are re-enforced in Problem 12 (i.e. all samples within the same group share the same copy of centers), while non-anticipativity constraints between groups are still relaxed. Therefore, sample grouping based decomposition leads to a tighter relaxation and provides a stronger lower bound than the decomposition based on each individual sample. Therefore, we have:

Proposition 2. $\beta(M) \leq \beta^{SG}(M) \leq z(M)$.

Although sample grouping provides a tighter relaxation, the computational cost also increases dramatically compared with the basic lower bounding problem. It is because each subproblem 14 is a MISOCP problem that needs to be solved to global optimality.

The way of assigning groups strongly influences the quality of lower bounds. However, finding the optimal groups is itself an NP-hard problem. Therefore we developed a heuristic to group samples. We first divide the samples into K_{SG} clusters (default of $K * G$) with the constraints $\mu \in M$, which is solved using local optimal solvers as shown in Section 2.3.2. Then samples within each cluster is evenly distributed to different groups. In our implementation, the grouping scheme is fixed and passed through all nodes during the BB procedure.

2.2.3. Lagrangian Decomposition

Another approach to compute tighter lower bounds is through Lagrangian Decomposition, in which non-anticipativity constraints are not removed but dualized. They are multiplied by fixed Lagrange multipliers λ and added to the objective function. Here we discuss the Lagrangian Decomposition based on each individual sample, while it can also be combined with sample grouping. The relaxed problem based on Lagrangian Decomposition can be written in the following form:

$$\beta^{LD}(M,\lambda) := \min_{\mu\in M}\{\sum_{s\in\mathcal{S}} Q_s(\mu_s) + \sum_{s=1}^{S-1}\lambda_s(\mu_s - \mu_{s+1})\} \tag{15}$$

It is clear that the basic lower bounding problem 7 is a special case of problem 15 with $\lambda = 0$. Problem 15 can be decomposed into S sub-problems:

$$\beta_s^{LD}(M,\lambda) := \min_{\mu_s\in M}\{Q_s(\mu_s) + (\lambda_s - \lambda_{s-1})\mu_s\} \tag{16}$$

with $\lambda_0 = \lambda_S = 0$ and $\beta^{LD}(M,\lambda) = \sum_{s\in\mathcal{S}} \beta_s^{LD}(M,\lambda)$.

It can be shown that the solution to Problem 15 provides a lower bound to the primal node problem 6 by noticing that the optimal solution to Problem 6 is also feasible with respect to Problem 15 with the same objective value. Note that the value of λ in Problem 15 is fixed and choosing a proper value of λ may produce a tighter lower bound. To achieve the tightest lower bound, we need to solve the Lagrangian dual problem:

$$\beta^{LD}(M) = \max_{\lambda} \beta^{LD}(M,\lambda). \tag{17}$$

Solving this dual problem is itself very challenging. The community usually approaches with heuristic methods, including the Sub-gradient Method (Fisher, 1981), Volume Algorithm (Barahona & Anbil, 2000), and Progressive Hedging Algorithm (Rockafellar & Wets, 1991). In this paper, we adopt the sub-gradient method for the update of λ.

Since the lower bound computed from basic lower bounding problem satisfying $\beta(M) = \beta^{LD}(M,0)$. Thus, we have the following proposition:

Proposition 3. $\beta(M) \le \beta^{LD}(M) \le z(M)$

Compared with the basic lower bound problem, although Lagrangian Decomposition has the potential to find a tighter bound if a close-to-optimal λ is computed from the Lagrangian dual problem, the computational cost of solving a MISOCP subproblem 16 is much higher than solving a QP subproblem 10, which has a closed-form solution.

2.3. Upper Bounds

This section describes two approaches to generate upper bounds for the primal node Problem 5. Both methods are included in our implementation.

2.3.1. Basic Upper Bounding Problem

An upper bound of the primal node Problem 5 can be obtained by fixing the first stage variable μ at a candidate solution $\hat{\mu} \in M$. We denote the solution of the upper bounding problem as follow:

$$\alpha(M) = \sum_{s\in\mathcal{S}} Q_s(\hat{\mu}). \tag{18}$$

Because the closed-form solution to $Q_s(\hat{\mu})$ is known, the computation of $\alpha(M)$ is trivial without the need to solve any optimization problem. Since $\hat{\mu}$ is an arbitrary feasible solution to the primal node Problem 5, it is obviously that $z(M) \le \alpha(M), \forall\hat{\mu} \in M$. The choice of candidate solution is a critical decision. In our implementation, the algorithm will test $\hat{\mu}$ obtained from both lower bounding problems as discussed in Section 2.2 and local optimization as discussed in Section 2.3.2.

2.3.2. Local Optimal Solution

Another way of computing the upper bound is to solve the primal node Problem 5 to local optimality. One alternative formulation of the Problem 5, which is equivalent to Problem 2 with the addition of $\mu \in M$, is of the following form:

$$\begin{aligned} \min_{\mu\in M,b} \sum_{s\in\mathcal{S}}\sum_{k\in\mathcal{K}} b_{s,k}||x_s - \mu_k||^2 \\ \sum_{k\in\mathcal{K}} b_{s,k} = 1 \\ 0 \le b_{s,k} \le 1 \\ s\in\mathcal{S}, k\in\mathcal{K} \end{aligned} \tag{19}$$

The reason why $b_{s,k} \in \{0,1\}$ can be replaced by $0 \le b_{s,k} \le 1$ is because, if μ_k is fixed, a sample x_s will always be assigned to the cluster with the nearest center. This Nonlinear Programming Problem (NLP) can be easily solved using off-the-shelf solvers like Ipopt (Wächter & Biegler, 2006) to local optimality. In our implementation, we run the local solver several trails using different initial values to get the best local optimal value as the node's upper bound.

2.4. Branch-and-Bound Clustering Scheme

We adopt the framework of the reduced-space branch-and-bound scheme from (Cao & Zavala, 2019) and tail the algorithm specifically for the MSSC task. Algorithm 1 depicts the details of such scheme.

Algorithm 1 Branch-and-Bound Clustering Scheme

Initialization
Initialize the iteration index $i = 0$;
Set $\mathbb{M} \leftarrow \{M_0\}$, and tolerance $\epsilon > 0$;
Compute initial upper and lower bounds $\alpha_i = \alpha(M_0)$, $\beta_i = \beta(M_0)$;
repeat
  Node Selection
  Select a set $M \in \mathbb{M}$ satisfying $\beta(M) = \beta_i$;
  $\mathbb{M} \leftarrow \mathbb{M} \setminus \{M\}$;
  $i \leftarrow i + 1$;
  Branching
  Partition M into subsets M_1 and M_2 with $relint(M_1) \cap relint(M_2) = \emptyset$;
  Add each subset to $\mathbb{M}$ to create separated child nodes;
  Bounding
  compute $\alpha(M_1)$, $\beta(M_1)$, $\alpha(M_2)$, $\beta(M_2)$;
  $\beta_i \leftarrow \min\{\beta(M') \mid M' \in \mathbb{M}\}$;
  $\alpha_i \leftarrow \min\{\alpha_{i-1}, \alpha(M_1), \alpha(M_2)\}$;
  Remove all M' from $\mathbb{M}$ if $\beta(M') \geq \alpha_i$;
  If $|\beta_i - \alpha_i| \leq \epsilon$, STOP;
until $\mathbb{M} = \emptyset$

3. Convergence Analysis

In this section, we establish the convergence for the BB scheme constructed using only the basic lower and upper bounding problems. Since sample grouping and Lagrangian Decomposition will only provide tighter bounds, they will not break the convergence. A key feature of our BB scheme is that it only needs to branch on the space of first stage variables μ to guarantee convergence.

Our proposed BB clustering algorithm can be regarded as a rooted tree. The root node is the original variable space $\mathcal{M}_0$. This node is indexed at level 0. We denote M_{i_q} as the node at level q which is explored at iteration i_q. A node $M_{i_{q+1}}$ is a child node that connected to its parent node M_{i_q}, with $M_{i_{q+1}} \subset M_{i_q}$. The child node is at level $q+1$ and is explored at iteration i_{q+1}. We denote $\{M_{i_q}\}$ as the sequence of the partition element that represents a path of the tree from the root node to the node M_{i_q} at the level q. Since the search space is narrowing along the path, the sequence $\{\beta_i\}$ is monotonically increasing, while $\{\alpha_i\}$ is monotonically decreasing. A BB scheme is said to be convergent if $\lim_{i\to\infty} \alpha_i = \lim_{i\to\infty} \beta_i = z$. If the scheme is convergent, then it produces a global ϵ-optimal solution in a finite number of steps.

In the proof analysis, we adopt the basic results from the work in (Cao & Zavala, 2019) and the seminal work in the Chapter IV of (Horst & Tuy, 2013). Unlike these works in which the general constraints might implicitly reduce the feasible regions, any point $\mu \in M$ in the clustering problem is a feasible solution. Therefore, we modified definitions and theories accordingly in this paper.

Lemma 4 (Corollary IV.1 (Horst & Tuy, 2013))**.** *If a BB procedure is infinite, then it generates at least one infinitely decreasing sequence $\{M_{i_q}\}$ of successively refined partition elements, $M_{i_{q+1}} \subset M_{i_q}$.*

Definition 5 (Definition IV.10 (Horst & Tuy, 2013))**.** *A subdivision is called **exhaustive** if $\lim_{q\to\infty} \delta(M_{i_q}) = 0$, for all decreasing sub-sequences $\{M_{i_q}\}$ generated by the subdivision.*

It is easy to see that if the element of μ that corresponds to $\delta(M)$ is selected for partitioning, the created subdivision is guaranteed to be exhaustive.

The next several conclusions help to prove the convergence of lower bounds.

Definition 6 (Definition IV.7 (Horst & Tuy, 2013))**.** *A lower bounding operation is called **strongly consistent**, if, at every iteration, any undeleted partition set can be further refined and if any infinite decreasing sequence $\{M_{i_q}\}$ of successively refined partition elements, contains a sub-sequence $\{M_{i_{q'}}\}$, satisfying, $\lim_{q'\to\infty} \beta(M_{i_{q'}}) = z(\overline{M})$, where $\overline{M} = \bigcap_q M_{i_q}$.*

Lemma 7. *Given an exhaustive subdivision on μ, The lower bounding operation in Algorithm 1 is strongly consistent.*

Proof. With an exhaustive subdivision, M_{i_q} shrinks to a single point $\bar{\mu}$ and we thus have that $\bar{M} = \{\bar{\mu}\}$. We then prove the lemma by showing that $\lim_{q\to\infty} \beta(M_{i_q}) = z(\bar{M}) = \sum_{s\in\mathcal{S}} Q_s(\bar{\mu})$. Let $\tilde{\mu}_{i_q,s} \in \operatorname{argmin}\{Q_s(\mu_s) : \mu_s \in M_{i_q}\}$, since M_{i_q} shrinks to $\bar{\mu}$, we have $\lim_{q\to\infty} \tilde{\mu}_{i_q,s} = \bar{\mu}$. Based on the continuity of $Q_s(.)$ (Lemma 1), we have $Q_s(\bar{\mu}) = \lim_{q\to\infty} Q_s(\tilde{\mu}_{i_q,s}) = \lim_{q\to\infty} \beta_s(M_{i_q})$. Take the sum over s, we obtain $\lim_{q\to\infty} \beta(M_{i_q}) = \sum_{s\in\mathcal{S}} Q_s(\overline{M})$. □

Definition 8 (Definition IV.6 (Horst & Tuy, 2013))**.** *A selection operation is said to be **bound improving**, if, after a finite number of steps, at least one partition element where the actual lower bounding is attained is selected for further partition.*

The selection operation in Algorithm 1 is bound improving, because at each iteration, Algorithm 1 selects the node

where the actual lower bounding is attained for further partition.

Lemma 9 (Theorem IV.3 (Horst & Tuy, 2013))**.** *For a BB scheme using a lower bounding operation that is strongly consistent and using a selection operation that is bound improving, we have that* $\lim_{i\to\infty} \beta_i = z$.

Lemma 10. *Given an exhaustive subdivision on* μ*, Algorithm 1 satisfies* $\lim_{i\to\infty} \beta_i = z$.

Proof. This result can be obtained from Lemma 7 and 9. □

Finally, the convergence of the upper bounds is established through the following lemma.

Lemma 11. *Given an exhaustive subdivision on* μ*, Algorithm 1 generates a sequence* $\{\alpha_i\}$ *such that* $\lim_{i\to\infty} \alpha_i = z$.

Proof. Let $\mu^* \in M_0$ denote an optimal solution of the MSSC task. Because of the continuity of Q (implied by the continuity of Q_s), we have $Q(\mu) - Q(\mu^*) \leq K||\mu - \mu^*||$ for all $\mu \in M$. Given $\epsilon > 0$, $r = \epsilon/K$, for every point $\mu \in B_r(\mu^*)$, we have $Q(\mu) - Q(\mu^*) \leq K||\mu - \mu^*|| \leq \epsilon$ holds.

Since the subdivision is exhaustive, after a finite number of iterations i, we have, either the partition considered satisfies $M_i \subseteq B_r(\mu^*)$, or the partition M_i which contain the solution μ^* is pruned. For the first case, since $M_i \subseteq B_r(\mu^*)$, we have $Q(\mu) - Q(\mu^*) \leq \epsilon, \forall \mu \in M_i$. Then we have that $\alpha_i \leq \alpha(M_i) \leq Q(\mu^*) + \epsilon$. In the second one, since M_i is pruned, then we have $\alpha_i \leq \alpha(M_i) \leq \beta(M_i) + \epsilon$. Because $\mu^* \in M_i$, we also have $\beta(M_i) \leq Q(\mu^*)$. Hence, $\alpha_i \leq Q(\mu^*) + \epsilon$. For both cases, we have $z \leq \alpha_i \leq z + \epsilon$. Since the above conclusion holds for arbitrary $\epsilon > 0$, $\lim_{i\to\infty} \alpha_i = z$ holds. □

Combing Lemma 10 and 11, we obtain the following theorem:

Theorem 12. *Given an exhaustive subdivision on* μ*, Algorithm 1 converges in the sense that*

$$\lim_{i\to\infty} \alpha_i = \lim_{i\to\infty} \beta_i = z. \tag{20}$$

4. Computational Experiments

In this section, we evaluate the performance of our algorithm on both synthetic and real-world datasets. Our algorithm is implemented in Julia 1.5.3. We compare the computational results (in serial and parallel) against those of the classic BB algorithm implemented in the state-of-the-art global optimizer CPLEX 20.1.0 (Cplex, 2020), and classic k-means clustering algorithm implemented in Julia Package Clustering. The result of k-means clustering algorithm is generated by repeating with 100 trials using random initialization. The worst, average and best k-means objectives are recorded. Both our algorithm and CPLEX terminate with a time limit of 4 hours We also did experiments on our BB clustering algorithm (BBClst) with different convergence acceleration techniques. Here, closed-form solution (BBClst+CF), scenario-based sample grouping (BBClst+SG), and grouping based Lagrangian decomposition (BBClst+LD+SG) are applied for comparison. We also present a preliminary parallel implementation in which the subproblems in BBClst+LD+SG are assigned to multiple CPU cores. For sample grouping, we limited each group's size under $\min(162/d - k, 10 \times k)$ when decomposing the lower bounding problem. All serial experiments are executed on a 2.3GHz quad-core 10th-generation Intel Core i7 processor with 16G RAM. The parallel experiments are executed on the Niagara Cluster of Compute Canada. The complete code files can be found in `https://github.com/kingsley1989/global_kmeans`.

Algorithms are compared based on three criteria: upper bound (UB), optimality gap, and number of nodes being solved. UB represents the best found (feasible) solution reported at the termination of each algorithm. Optimality gap is measured by the difference between the best possible (lower bound or LB) and best known solutions (UB) and is calculated as $Gap \equiv \frac{UB-LB}{LB}$. The optimality gap of ϵ provides a certificate that the best found solution is not worse than $\epsilon\%$ from the optimal solution. Therefore, it serves as a benchmark index on whether searching for a better solution is still necessary (if the optimality gap is already very low, then the potential improvement is small). The optimality gap is a unique property of deterministic global optimization algorithms. Heuristic techniques such as k-means clustering algorithm do not have the capability to evaluate the quality of its solutions (in the deterministic sense). The number of nodes records how many iterations the BB procedure processed within specific time period.

Synthetic data. We first consider numerical performance on artificially generated datasets. To illustrate the scalability of the algorithms, we consider datasets with different numbers of samples. All datasets are generated with 3 Gaussian clusters randomly with $seed = 1$. Each data sample has two attributes. For each dataset, we solve two clustering problems ($k = 3$ and $k = 4$). The number of variables involved in the optimization problem increases linearly with the number of samples and clusters.

Table 1 compares the performance of our BB-based clustering algorithms (BBClst), CPLEX, and k-means algorithm. In terms of the best found (feasible) solution or UB, the BBClst based algorithms can always obtain the lowest sum-of-squares cost. For example, for dataset *Syn-2100* on problem

Table 1. Computational performance of different algorithms on synthetic datasets.

DATASETS	METHODS	$k=3$			$k=4$		
		UB	NODES	GAP(%)	UB	NODES	GAP(%)
Syn-300	k-MEANS (WORST)	6454.84	-	-	6379.28	-	-
	k-MEANS (AVERAGE)	4506.19	-	-	3416.73	-	-
	k-MEANS (BEST)	**4049.28**	-	-	**3170.97**	-	-
	CLASSIC BB (CPLEX)	6945.02	8955000	1046.8	6782.52	8181700	3981.6
	BBCLST+CF	**4049.28**	397442	7.31	**3170.97**	360753	35.67
	BBCLST+SG	**4049.28**	4475	0.88	**3170.97**	655	6.02
	BBCLST+LD+SG	**4049.28**	73	$< \mathbf{0.1}(2h)$	**3170.97**	41	**2.67**
	BBCLST+LD+SG(20CORES)	**4049.28**	73	$< \mathbf{0.1}(0.4h)$	**3170.97**	481	**0.29**
Syn-1200	k-MEANS (WORST)	24465.51	-	-	23796.32	-	-
	k-MEANS (AVERAGE)	16223.76	-	-	12578.29	-	-
	k-MEANS (BEST)	**14290.53**	-	-	**11812.94**	-	-
	CLASSIC BB (CPLEX)	26417.87	2708500	27677.8	26154.47	1649002	58723.5
	BBCLST+CF	**14290.53**	352489	7.42	**11812.94**	353012	44.73
	BBCLST+SG	**14290.53**	714	2.20	**11812.94**	169	**12.26**
	BBCLST+LD+SG	**14290.53**	32	**1.21**	**11812.94**	7	20.81
	BBCLST+LD+SG(20CORES)	**14290.53**	159	$< \mathbf{0.1}(0.6h)$	**11812.94**	177	**2.79**
Syn-2100	k-MEANS (WORST)	43141.42	-	-	41943.30	-	-
	k-MEANS (AVERAGE)	28111.86	-	-	21639.97	-	-
	k-MEANS (BEST)	**25033.48**	-	-	**20598.32**	-	-
	CLASSIC BB (CPLEX)	47349.02	930610	23709.5	47431.11	603200	333233.3
	BBCLST+CF	**25033.48**	333910	7.26	**20598.32**	335763	41.61
	BBCLST+SG	**25033.48**	233	3.20	**20598.32**	107	**12.41**
	BBCLST+LD+SG	**25033.48**	14	**2.71**	**20598.32**	3	25.14
	BBCLST+LD+SG(20CORES)	**25033.48**	322	**0.26**	**20598.32**	124	**3.49**
Syn-42000	k-MEANS (WORST)	847326.29	-	-	822601.88	-	-
	k-MEANS (AVERAGE)	570643.51	-	-	437109.88	-	-
	k-MEANS (BEST)	**501472.82**	-	-	**411815.14**	-	-
	CLASSIC BB (CPLEX)	NO FEASIBLE SOLUTION FOUND.			NO FEASIBLE SOLUTION FOUND.		
	BBCLST+CF	**501472.82**	148583	9.82	**411815.14**	139891	75.21
	BBCLST+SG	**501472.82**	4	**5.43**	411815.14	3	23.46
	BBCLST+LD+SG	**501472.82**	1	7.45	**411815.14**	1	**21.70**
	BBCLST+LD+SG(20CORES)	**501472.82**	20	**3.16**	**411815.14**	4	**12.99**

$k = 3$, our algorithm can reduce the optimal cost by 47.13% compared with the best solution returned by CPLEX. Remarkably, the best k-means objectives out of 100 trials also obtain the lowest cost. This process of repeatedly using local optimal algorithms with multiple random initialization can be viewed as a stochastic global optimization strategy. However, one key limitation of this strategy is that it cannot compute the optimality gap and evaluate if it is necessary to continue the search for better solutions. In contrast, the optimality gap provided by deterministic global optimization algorithms gives a certificate on the solution quality.

In terms of the optimality gap, CPLEX cannot converge into a comparatively low value within a budget of 4 hours, and the optimality gap remains large (over 10000%) at the end of the solution process and in extreme cases, can not find a feasible solution for problems on large dataset (e.g. *Syn-42000*) within 4 hours. On the contrary, our algorithm can always maintain a comparatively low optimality gap after four-hour execution. Specifically, for the problem with $k = 3$ on dataset *Syn-300* which has 300 samples, our algorithm ends with an optimality gap lower than 0.1% in within 2 hours in serial and within 0.4 hours in parallel (20 cores). Even when we increase the number of samples to 42000, the optimality gap remains at 5.43% in serial and 3.16% in parallel (20 cores), under the same runtime.

Comparing our algorithms using different methods to generate lower bounds, we find that, scenario-based sample grouping may be beneficial for some MSSC problems, while the Lagrangian decomposition component may accelerate or slow down the solution process depending on the number of variables and clusters.

The decomposition scheme of our algorithm enables an easy way for paralleling. To further illustrate the scalablity of our algorithm on the MSSC problem. We tested BBclst+LD+SG on the synthetic dataset with 210,000 samples in parallel of 200 cores. The results are listed in Table 2. Here, we can see that even for dataset over 200,000 samples, which is **100 times larger** the state of the art (2392 samples) by (Aloise et al., 2012),

Table 2. Computational performance of large synthetic dataset in parallel. (BBClst+LD+SG, 200 cores, $k = 3$)

DATASET	UB	NODES	GAP(%)
Syn-210000	$2.43\text{x}10^6$	6	2.55

Real-world data. We then perform numerical experiments on several real-world datasets. First, we pick three datasets with different scales to demonstrate the performance and generalization ability of our algorithm. Two of them are well known benchmark instances. The *Iris* dataset (Fisher, 1936) has 150 samples and 4 attributes, while the *Seeds* dataset (Charytanowicz et al., 2010) contains 210 samples and 7 attributes. The third dataset, *Hemicellulose*, contains 1955 experimental data samples on batch hemicellulose hydrolysis of hardwood. Each of its data sample represents a reaction condition with 7 attributes.

Table 3 summarizes the performance of different algorithms on these three datasets. In terms of the UB, our algorithms and k-means (best) attains the lowest cost for all datasets. In terms of the optimality gap, our algorithm reports a significantly lower optimality gap compared to the Classic BB method. Remarkably, our implementation BBClst+LD+SG can converge to 0.1% of the optimality gap within 1.5 hour execution for the *Iris* dataset in serial. For parallel execution with 20 cores, only 0.6 hour and 1 hour executions are needed for *Iris* and *Seeds* dataset to converge to 0.1%, respectively. In contrast, CPLEX halts at 328.63%.

Table 3. Computational performance of different algorithms on real-world datasets. ($k = 3$)

METHODS	UB	NODES	GAP(%)
Iris ($n = 150, d = 4$)			
k-MEANS (WORST)	145.45	-	-
k-MEANS (AVERAGE)	85.91	-	-
k-MEANS(BEST)	**78.85**	-	-
CLASSIC BB(CPLEX)	82.64	12313100	328.63
BBCLST+CF	**78.85**	382280	50.76
BBCLST+SG	**78.85**	876	1.58
BBCLST+LD+SG	**78.85**	31	**0.1**$(1.5h)$
BBCLST+LD+SG(20CORES)	**78.85**	31	**0.1**$(0.6h)$
Seeds ($n = 210, d = 7$)			
k-MEANS (WORST)	916.21	-	-
k-MEANS (AVERAGE)	591.46	-	-
k-MEANS(BEST)	**587.32**	-	-
CLASSIC BB(CPLEX)	626.37	6715463	850.57
BBCLST+CF	**587.32**	356273	69.49
BBCLST+SG	**587.32**	443	4.34
BBCLST+LD+SG	**587.32**	47	**0.26**
BBCLST+LD+SG(20CORES)	**587.32**	89	**0.1**$(1h)$
Hemicellulose ($n = 1,955, d = 7$)			
k-MEANS (WORST)	$16.98\text{x}10^6$	-	-
k-MEANS (AVERAGE)	$10.20\text{x}10^6$	-	-
k-MEANS(BEST)	$\mathbf{9.75\text{x}10^6}$	-	-
CLASSIC BB(CPLEX)	$18.38\text{x}10^6$	478800	4950.51
BBCLST+CF	$\mathbf{9.75\text{x}10^6}$	326896	59.48
BBCLST+SG	$\mathbf{9.75\text{x}10^6}$	74	**21.75**
BBCLST+LD+SG	$\mathbf{9.75\text{x}10^6}$	4	39.38
BBCLST+LD+SG(20CORES)	$\mathbf{9.75\text{x}10^6}$	112	**2.23**

The rest two datasets are selected from (Aloise et al., 2012) to compare the performance with current state of the art. The Padberg and Rinald's hole drilling dataset (Padberg & Rinaldi, 1991) who has 2392 samples and 2 attributes is the dataset with maximum sample size in (Aloise et al., 2012). The glass identification dataset (Dua & Graff, 2017) is also a well-know benchmark for clustering problem. It has 214 samples and 9 attributes. Table 4 expounds the results of each dataset for problem $k = 2$.

Specifically, Aloise's algorithm is superior when k is large or $n/k \approx 10$ (Aloise et al., 2012). In contrast, we offer a different approach that can process large datasets with relatively small number of clusters. As shown in Table 4, for problem $k = 2$, Aloise's algorithm cannot solve the glass identification data and is very slow for Padberg and Rinald's data. Yet, our BB clustering algorithm can even hit the gap lower than 0.1% within 12 hours on both datasets.

Table 4. Comparison on datasets with (Aloise et al., 2012). (BBClst+LD+SG, $k = 2$)

METHODS	UB	NODES	GAP(%)
Padberg and Rinald's Dataset ($n = 2,392, d = 2$)			
ALOISE ET AL.	$2.967\text{x}10^{10}$	1	i^2 $(50h)$
SERIAL	$2.967\text{x}10^{10}$	7	1.32 $(4h)$
SERIAL	$2.967\text{x}10^{10}$	253	0.1 $(11h)$
20 CORES	$2.967\text{x}10^{10}$	247	0.1 $(1h)$
Glass Identification ($n = 214, d = 9$)			
ALOISE ET AL.	CANNOT BE SOLVED		
SERIAL	819.63	85	28.65 $(4h)$
SERIAL	819.63	339	0.1 $(9h)$
20 CORES	819.63	415	0.1 $(1h)$

[2]Solved at the root node.

5. Conclusion

In this paper, we proposed a scalable global optimization algorithm for MSSC problems. This algorithm's key advantages are that branching needs to be performed only on the centers of clusters, and lower bounding problems can be decomposed into smaller subproblems. We proved that the algorithm converges to a global ϵ-optimal solution. The numerical experiments demonstrated our algorithm's ability to handle datasets with up to 200,000 samples, which improves scale of solvable MSSC problem by 100 times larger than state of the art. Numerical results also illustrate that the algorithm can provide the deterministic clustering results with good generations on diverse real-world datasets.

Acknowledgements

We appreciate the useful comments from anonymous reviewers during the improvement of this paper.

References

Agarap, A. F. and Azcarraga, A. P. Improving k-means clustering performance with disentangled internal representations. *arXiv preprint arXiv:2006.04535*, 2020.

Aloise, D. and Hansen, P. A branch-and-cut sdp-based algorithm for minimum sum-of-squares clustering. *Pesquisa Operacional*, 29(3):503–516, 2009.

Aloise, D. and Hansen, P. Evaluating a branch-and-bound rlt-based algorithm for minimum sum-of-squares clustering. *Journal of Global Optimization*, 49(3):449–465, 2011.

Aloise, D., Hansen, P., and Liberti, L. An improved column generation algorithm for minimum sum-of-squares clustering. *Mathematical Programming*, 131(1):195–220, 2012.

Barahona, F. and Anbil, R. The volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming*, 87(3):385–399, 2000.

Brusco, M. J. A repetitive branch-and-bound procedure for minimum within-cluster sums of squares partitioning. *Psychometrika*, 71(2):347–363, 2006.

Cao, Y. and Zavala, V. M. A scalable global optimization algorithm for stochastic nonlinear programs. *Journal of Global Optimization*, 75(2):393–416, 2019.

CarøE, C. C. and Schultz, R. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1-2):37–45, 1999.

Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P. A., Łukasik, S., and Żak, S. Complete gradient clustering algorithm for features analysis of x-ray images. In *Information technologies in biomedicine*, pp. 15–24. Springer, 2010.

Cplex, I. I. V20.1.0: User's manual for CPLEX. *International Business Machines Corporation*, 2020.

Diehr, G. Evaluation of a branch and bound algorithm for clustering. *SIAM Journal on Scientific and Statistical Computing*, 6(2):268–284, 1985.

Du Merle, O., Hansen, P., Jaumard, B., and Mladenovic, N. An interior point algorithm for minimum sum-of-squares clustering. *SIAM Journal on Scientific Computing*, 21(4): 1485–1505, 1999.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Fisher, M. L. The lagrangian relaxation method for solving integer programming problems. *Management science*, 27 (1):1–18, 1981.

Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

Freed, N. and Glover, F. A mixed-integer programming approach to the clustering problem. *Communications in Statistics-Simulation and Computation*, 12(5):595–607, 1983.

Gamrath, G., Anderson, D., Bestuzheva, K., Chen, W.-K., Eifler, L., Gasse, M., Gemander, P., Gleixner, A., Gottwald, L., Halbig, K., et al. The scip optimization suite 7.0. 2020.

Geoffrion, A. M. Generalized benders decomposition. *Journal of optimization theory and applications*, 10(4):237–260, 1972.

Horst, R. and Tuy, H. *Global optimization: Deterministic approaches*. Springer Science & Business Media, 2013.

Jain, A. K. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

Karuppiah, R. and Grossmann, I. E. A lagrangean based branch-and-cut algorithm for global optimization of nonconvex mixed-integer nonlinear programs with decomposable structures. *Journal of global optimization*, 41(2): 163–186, 2008.

Khajavirad, A. and Michalek, J. J. A deterministic lagrangian-based global optimization approach for quasiseparable nonconvex mixed-integer nonlinear programs. *Journal of mechanical design*, 131(5), 2009.

Komodakis, N., Paragios, N., and Tziritas, G. Clustering via lp-based stabilities. *Advances in neural information processing systems*, 21:865–872, 2008.

Koontz, W. L. G., Narendra, P. M., and Fukunaga, K. A branch and bound clustering algorithm. *IEEE Transactions on Computers*, 100(9):908–915, 1975.

Li, C. and Grossmann, I. E. A generalized benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables. *Journal of Global Optimization*, 75(2):247–272, 2019.

Li, X., Tomasgard, A., and Barton, P. I. Nonconvex generalized benders decomposition for stochastic separable mixed-integer nonlinear programs. *Journal of optimization theory and applications*, 151(3):425, 2011.

Likas, A., Vlassis, N., and Verbeek, J. J. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

Lloyd, S. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

Misener, R. and Floudas, C. A. Antigone: algorithms for continuous/integer global optimization of nonlinear equations. *Journal of Global Optimization*, 59(2-3):503–526, 2014.

Morrison, D. R., Jacobson, S. H., Sauppe, J. J., and Sewell, E. C. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

Optimization, G. Inc.,"gurobi optimizer reference manual," 2015, 2014.

Padberg, M. and Rinaldi, G. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33(1):60–100, 1991.

Peng, J. and Wei, Y. Approximating k-means-type clustering via semidefinite programming. *SIAM journal on optimization*, 18(1):186–205, 2007.

Rockafellar, R. T. and Wets, R. J.-B. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research*, 16(1):119–147, 1991.

Sağlam, B., Salman, F. S., Sayın, S., and Türkay, M. A mixed-integer programming approach to the clustering problem with an application in customer segmentation. *European Journal of Operational Research*, 173(3):866–879, 2006.

Sherali, H. D. and Desai, J. A global optimization rlt-based approach for solving the hard clustering problem. *Journal of Global Optimization*, 32(2):281–306, 2005.

Späth, H. *Cluster analysis algorithms for data reduction and classification of objects*. Horwood, 1980.

Tawarmalani, M. and Sahinidis, N. V. A polyhedral branch-and-cut approach to global optimization. *Mathematical programming*, 103(2):225–249, 2005.

Tzortzis, G. and Likas, A. The minmax k-means clustering algorithm. *Pattern Recognition*, 47(7):2505–2516, 2014.

Wächter, A. and Biegler, L. T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

Xu, J. and Lange, K. Power k-means clustering. In *International Conference on Machine Learning*, pp. 6921–6931, 2019.