TOP
https://doi.org/10.1007/s11750-024-00684-w

ORIGINAL PAPER

A bisection method for solving distance-based clustering problems globally

Peter Kirst[1] · **Tomáš Bajbar**[2] · **Mario Merkel**[2]

Received: 4 May 2023 / Accepted: 9 August 2024

Abstract
In this article, we consider distance-based clustering problems. In contrast to many approaches, we use the maximum norm instead of the more commonly used Euclidean norm to measure distances. This problem is nonsmooth and non-convex and, thus, difficult to solve to global optimality using standard approaches, which is common in cluster analysis. Therefore, we reformulate this continuous problem in light of graph-theoretical instances which enables us to construct a bisection algorithm converging to the globally minimal value of the original clustering problem by establishing valid upper and lower bounding procedures. Our numerical results indicate that our method performs well on data sets exhibiting clear cluster-pattern structure even for bigger data instances while still guaranteeing the global optimality of the computed solution. We compare our approach with the classical k-means algorithm and also discuss the limits and challenges of the proposed procedure.

Keywords Clustering problem · Global optimization · Maximal clique · Bisection method · k-means problem

Mathematics Subject Classification 90C30 · 90C26 · 90C35

1 Introduction

Clustering is a well-established unsupervised learning technique which aims to allocate a given set of data points to specific subsets called *clusters*, such that points within a cluster are similar in some sense, whereas points of different clusters may differ.

✉ Peter Kirst
peter.kirst@wur.nl

Tomáš Bajbar
tomas.bajbar@gmail.com

Mario Merkel
merkel-mario@gmx.de

[1] Wageningen University and Research, Wageningen, The Netherlands

[2] Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

Published online: 11 September 2024

From an optimization perspective, this can be achieved by means of minimizing some dissimilarity measure. Since cluster analysis is an important approach that is helpful in various applications, it is, therefore, not surprising that a large amount of literature is devoted to this topic. We refer to the surveys (Hansen and Jaumard 1997; Jain et al. 1999) as well as to the monographs (Hartigan 1975; Kogan et al. 2006) for a general introduction.

There are several different possibilities to tackle this problem. A classical approach that is widely used in practice is the k-means algorithm. Its various versions, usually considering the Euclidean norm for measuring distances, are described in Bagirov (2008); Bagirov et al. (2011); Hartigan and Wong (1979); Larrañaga et al. (1999); Likas et al. (2003); Loyd (1982); MacQueen (1976). Unfortunately, k-means algorithms do only compute locally optimal solutions and, thus, one can never be sure, whether there is a better clustering of the data points or not. This is well illustrated in Steinley (2003) and it is shown that this drawback may already occur for small-scale problems even when using professional software tools. Although implementations may have improved, since the latter was published in 2003, in principle, the difficulty of getting stuck in locally optimal points still persists. For this reason, the aim of this article is to provide a new possibility to solve a variant of this problem to global optimality.

Although, in general, it can be stated that there are much fewer algorithms to solve this problem globally compared to the large body of literature on heuristics or local solution approaches, there are also other distance-based approaches that aim at solving the problem globally. In the following we briefly mention some of these. First of all, it is interesting to see that basically all techniques that are common in the area of global optimization can be applied here as well such as dynamic programming (see, e.g., Jensen 1969; van Os and Meulman 2004), cutting plane methods (see Peng and Xia 2005), as well as reformulation–linearization-technique-based methods (RLT) as proposed in Hanif (1999); Sherali and Desai (2005). Moreover, the so-called cutting angle method is applied to clustering (see [5,6]). Recently progress has been made in solving the minimum sum-of-squares clustering problems (MSSC) globally where new techniques have been developed that combine semidefinite programming relaxations with branch-and-bound and branch-and-cut approaches (Piccialli et al. 2022b, a; Piccialli and Sudoso 2023).

Furthermore, clustering problems can be reformulated as mixed-integer linear as well as mixed-integer non-linear problems that can be solved using one of the excellent state-of-the-art solvers that are available. However, the success of these models is often somewhat limited as described, e.g., in Klein and Aronson (1991) due to the inherent combinatorial structure of these problems as well as due to many equivalent solutions. In the more recent study (Liberti and Manca 2022), an interesting idea based on the so-called random projections is applied that seems to be promising.

Among the most widely used techniques in global optimization are branch-and-bound methods. Therefore, it is not surprising that these are also applied to problems in statistics as well as data analysis in general. Possible applications including clustering can be found in Michael (2005). In addition, we mention (Koontz et al. 1975; Rao 1971) as some of the earliest approaches in that domain (see also Hand (1981)). More branch-and-bound algorithms can be found in Klein and Aronson (1991); Brusco (2003). An innovative idea of a branch-and-bound algorithm that exploits concepts

related to the so-called hierarchical clustering approaches can be found in Fränti et al. (2002). In Brusco (2006), yet another branch-and-bound procedure is proposed that is initially applied to a subset of all data points, which is subsequently enlarged.

A completely different approach to solve clustering problems to global optimality is column generation as done, for instance, in Johnson et al. (1993). The combination of this approach with a branch-and-bound method also leads to an interesting possibility for solving clustering problems to global optimality as developed in Merle et al. (1999). A considerably improved version based on this is presented in Aloise et al. (2012). An additional method based on column generation in general as well as on the two aforementioned articles, in particular, is proposed in Babaki et al. (2014). The approach is rather flexible in the sense that it is not only capable of solving clustering problems to global optimality but also allows for additional constraints to be incorporated into the model.

As opposed to the aforementioned methods, our new bisection method relies on concepts from graph theory for identifying a global solution of a clustering problem which we formulate as a non-convex continuous optimization problem. For a general introduction to graph theory, we refer to Bondy and Murty (1976); Diestel (2000); West (2001). In our method, two different subproblems arise, namely, the *maximal clique* and the *k-cover set* problem. Both are well-known and extensively studied problems settled within the areas of discrete and combinatorial optimization (Brigham and Dutton 1983; Bron and Kerbosch 1973; Karp 1972). For a description of algorithms to compute cliques of a graph in general, we refer to Johnston (1976) as well as to Pardalos and Rodgers (1992) for approaches based on classical branch-and-bound methods. As we are especially interested in the so-called maximal cliques, we also mention the well-known Bron–Kerbosch algorithm introduced in Bron and Kerbosch (1973) and, similarly, the article (Akkoyunlu 1973). Although being known to be $\mathcal{NP}$-hard, the clique cover problem can be accelerated, e.g., by data reduction techniques as described in Gramm et al. (2009). Additional clique problems are examined in Jun et al. (2020) and solved to local optimality.

The idea of applying graph-theoretical approaches to solve clustering problems is already known in the literature. Commonly, these approaches are not distance-based in the sense that distances between entities are assumed to be provided via a so-called distance matrix. In contrast, algorithms of k-means type interpret data points as elements of $\mathbb{R}^n$ and compute distances usually based on Euclidean distances. The method proposed in this article can be seen as a hybrid version. Although we also require data points in $\mathbb{R}^n$ as well as a norm to measure distances, we still take advantage of graph-theoretical ideas. However, as is common for k-means type algorithms, it is important to note that our approach does not work using arbitrary distance matrices. In fact, for our new method, the maximum norm is required here.

Graph-theoretical approaches for solving clustering problems have in common that data points are considered as vertices of a graph and the corresponding clusters are identified as certain subgraphs of this graph as, for instance, described in Johnson et al. (1993). Early and detailed introductions to graph-theoretical cluster techniques describing different algorithms for computing cliques can be found in Gary Augustson and Minker (1970) and the references therein. In Hansen and Delattre (1978), for solving a clustering problem, a minimum diameter partitioning problem is considered and

solved in terms of optimal coloring of certain subgraphs. Here, the optimal clusters correspond to the set of vertices of same color. Another clustering approach that is based on graph coloring can be found in Brusco and Cradit (2004). Alternatively, as suitable subgraphs, often *cliques* of a graph, also called complete subgraphs, are considered to identify clusters. A heuristic that exploits this idea is described in Dorndorf and Pesch (1994) and in Ben-Dor et al. (1999) the relationship between the clique and the clustering problem is exploited to solve problems regarding gene expression patterns. Considering subgraphs other than cliques, different shapes of clusters can be computed as done, e.g., in Gramm et al. (2005); Hartuv and Shamir (2000); Zhong et al. (2010).

The goal of this article is to propose a new method for the solution of the k-means problem with maximum norms. As already mentioned, we strive for a globally optimal point. Although we take advantage of graph-theoretical concepts, our approach is distance-based and, thus, we try to minimize the distances corresponding to the assignments of the data points to clusters by means of solving an optimization problem. We achieve this by means of a bisection method where in every iteration some classical graph-theoretical subproblems are solved. To the best of our knowledge, this is the first bisection method that is applied to a clustering problem. As common for k-means problems, we do not assume any prior knowledge about the cluster assignments of the given data points except that there is a certain number of clusters to be identified and that this number is given in advance. In contrast to most approaches from the literature, we measure distances in terms of maximum norms.

It should be noted that there are completely different clustering approaches such as DBSCAN (see Ester et al. 1996) or so-called hierarchical clustering approaches (see, e.g., Ward 1963 for an early reference in that area). The same holds for the numerous local and heuristic approaches, since we are only concerned with globally optimal solutions.

The article is structured as follows. In the next section, we introduce some important notions and define the global clustering optimization problem. Based on this, in Sect. 3, we explain some relations to important graph-theoretical concepts. In Sect. 4, we propose a new bisection method which uses the two aforementioned subproblems, namely the maximal clique and the k-cover set problem, to solve the original clustering problem globally. In addition, we provide a proof of convergence and briefly discuss the complexity of this approach. In Sects. 5, we discuss solution strategies for the aforementioned subproblems and we propose acceleration steps which further improve the overall computational performance of our algorithm. Finally, in Sect. 6, we compare the proposed method with a randomly initiated k-means method and we also discuss the results of the corresponding numerical experiments. The article closes with some final remarks in Sect. 7.

2 Preliminaries, notion of the problem, and basic reformulations

In this section, we briefly review some basic concepts needed throughout this article. We start by explaining some reformulations of clustering problems. The description is rather detailed to keep the exposition accessible to a broad readership ranging from

data analysis and graph theory to global optimization. At the same time, we already introduce some important ideas along a simple example.

Given a finite set of points $X = \{x^i \in \mathbb{R}^n \mid i = 1, \ldots, m\}$ with $m \in \mathbb{N}$ and given a number $k \in \mathbb{N}$, we are interested in partitioning the data set X in k different clusters $C_1, \ldots, C_k \subseteq X$ satisfying $\bigcup_{j=1}^{k} C_j = X$, so that data points within one cluster C_j are close to each other. Although we do not explicitly require the clusters $C_1, \ldots, C_k$ to be disjoint, commonly, this is fulfilled automatically for globally optimal clusterings.

To obtain a reasonable criterion for measuring the similarity of the data points within each cluster, we consider k cluster centers $z^1, \ldots, z^k \in \mathbb{R}^n$ which represent their position in $\mathbb{R}^n$. We are interested in minimal distances between each of the cluster centers and the corresponding assigned data points. More precisely, we try to position k cluster centers $z^1, \ldots, z^k$, such that the distances between the cluster centers and their assigned data points are minimized. We assign a given data point $x \in X$ to a cluster center z^j if and only if

$$\|z^j - x\| \leq \|z^l - x\| \quad \forall\, l = 1, \ldots, k$$

is fulfilled with $\|\cdot\|$ denoting an arbitrary norm. Note that we keep the exposition as general as possible here, although in the remainder of this article, we shall restrict our consideration to problems involving the maximum norm. The assignment of an arbitrary data point $x^i \in X$ to its closest distanced cluster center can be modeled by an assignment map $\sigma : \{1, \ldots, m\} \to \{1, \ldots, k\}$ defined by

$$z^{\sigma(i)} := \operatorname*{arg\,min}_{j=1,\ldots,k} \|z^j - x^i\|$$

for all $i = 1, \ldots, m$. Clearly, the assignment map σ is not unique in general and depends on the positions of the cluster centers. Once each data point is assigned to its closest distanced cluster center, then the overall quality of this particular clustering is measured by considering the norm of the vector of all minimal distances

$$\left\| \left(\|z^{\sigma(1)} - x^1\|, \ldots, \|z^{\sigma(m)} - x^m\| \right)^T \right\|.$$

Minimizing this expression leads to the unconstrained optimization problem

$$P : \min_{z^1,\ldots,z^k} \left\| \begin{pmatrix} \|z^{\sigma(1)} - x^1\| \\ \vdots \\ \|z^{\sigma(m)} - x^m\| \end{pmatrix} \right\| = \min_{z^1,\ldots,z^k} \left\| \begin{pmatrix} \min_{j=1,\ldots,k} \|z^j - x^1\| \\ \vdots \\ \min_{j=1,\ldots,k} \|z^j - x^m\| \end{pmatrix} \right\|.$$

The clustering problem P is, hence, to position k cluster centers $z^1, \ldots, z^k \in \mathbb{R}^n$, such that the objective function of the non-convex problem P is minimized. Moreover, in this article, we strive for a globally minimal point of P in contrast to many approaches from the literature.

In case a globally minimal point $(z^{1\star}, \ldots, z^{k\star})^T$ of P is given, we are able to partition the set of data points X into the corresponding k clusters $C_1, \ldots, C_k$ by

setting

$$C_j = \{x \in X \mid \|z^{j^\star} - x\| \leq \|z^l - x\|,\ l = 1, \ldots, k\}$$

for each $j = 1, \ldots, k$. Clearly, we obtain $C_j \subseteq X$ for all $j = 1, \ldots, k$, and, since each $x \in X$ belongs to at least one of the clusters $C_1, \ldots, C_k$, the property $\bigcup_{j=1}^{k} C_j = X$ is satisfied as well. Thus, identifying a globally minimal point of the problem P immediately leads to a clustering of the data set X.

Depending on the choice of the norm that is used for measuring distances, we obtain different objective functions of problem P and, hence, also different clustering results are possible in general. The most prominent method called k-means (see, e.g., Bagirov 2008; Bagirov et al. 2011; Likas et al. 2003; MacQueen 1976) is usually based on the Euclidean norm and the problem is, therefore, also called minimum sum-of-squares clustering problem (MSSC). In this article, however, we restrict our approach to maximum norms for both, the inner and the outer norms, appearing in the definition of the problem P. This norm is also known as the Chebyshev norm and is defined by $\|d\|_\infty = \max_{i=1,\ldots,n} |d_i|$ for an arbitrary $d \in \mathbb{R}^n$. This leads to the optimization problem

$$P^\infty : \min_{z^1,\ldots,z^k \in \mathbb{R}^n} \left\| \begin{pmatrix} \left\|z^{\sigma(1)} - x^1\right\|_\infty \\ \vdots \\ \left\|z^{\sigma(m)} - x^m\right\|_\infty \end{pmatrix} \right\|_\infty,$$

which is of our central interest in this article. By definition, this can be rewritten as

$$P^\infty : \min_{z^1,\ldots,z^k \in \mathbb{R}^n} \max_{i=1,\ldots,m} \min_{j=1,\ldots,k} \max_{l=1,\ldots,n} |z_l^j - x_l^i|.$$

We continue our reformulation by shifting the objective function into the constraints and, thus, we arrive at

$$\begin{aligned} \min_{z^1,\ldots,z^k \in \mathbb{R}^n,\ \alpha \in \mathbb{R}} \quad & \alpha \\ \text{s.t.} \quad & \left\| \begin{pmatrix} \min_{j=1,\ldots,k} \|z^j - x^1\|_\infty \\ \vdots \\ \min_{j=1,\ldots,k} \|z^j - x^m\|_\infty \end{pmatrix} \right\|_\infty \leq \alpha. \end{aligned}$$

Rewriting the outer norm explicitly yields

$$\begin{aligned} \min_{z^1,\ldots,z^k \in \mathbb{R}^n,\ \alpha \in \mathbb{R}} \quad & \alpha \\ \text{s.t.} \quad & \max_{i=1,\ldots,m} \min_{j=1,\ldots,k} \|z^j - x^i\|_\infty \leq \alpha. \end{aligned}$$

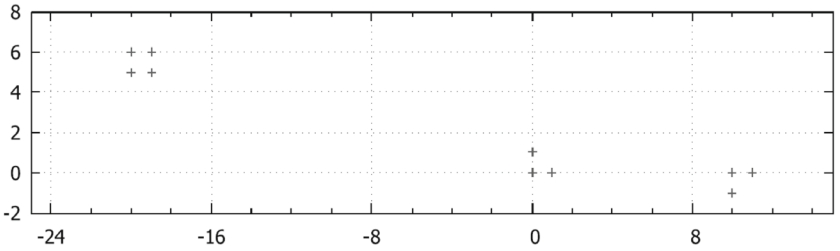


Fig. 1 Representation of the data points in X

We now continue by replacing the latter inequality constraint by m inequalities, and so, we finally obtain

$$\widetilde{P}: \quad \min_{z^1,\ldots,z^k\in\mathbb{R}^n,\ \alpha\in\mathbb{R}} \quad \alpha$$
$$\text{s.t.} \qquad \|z^{\sigma(i)} - x^i\|_\infty \leq \alpha \quad \forall i = 1,\ldots,m.$$

Thus, both problems P^∞ and $\widetilde{P}$ are equivalent in the sense that a point $(z^{1\star},\ldots,z^{k\star})$ is a globally minimal point of P^∞ if and only if there is some $\alpha^\star$, so that $(z^{1\star},\ldots,z^{k\star},\alpha^\star)$ is a globally minimal point of $\widetilde{P}$. In this case, the value $\alpha^\star \in \mathbb{R}$ attains the globally minimal value of both P^∞ and $\widetilde{P}$.

In view of the maximum norm, it is true that a data point x^i deviates from the next closest cluster center $z^{\sigma(i)}$ by at most α for any feasible point $(z^1,\ldots,z^k,\alpha)^T$ of $\widetilde{P}$. Thus, geometrically speaking, we try to position k cluster centers $z^1,\ldots,z^k$, such that k equal boxes (i.e., balls measured with respect to the maximum norm) centered at $z^1,\ldots,z^k$ have a minimal radius possible while ensuring that each data point $x \in X$ is contained in at least one of these boxes. This is illustrated in the following example.

Example 2.1 Let $X \subseteq \mathbb{R}^2$ be a set of ten data points

$$\begin{array}{llll} x^1 = (0,0)^T & x^2 = (0,1)^T & x^3 = (1,0)^T & \\ x^4 = (10,0)^T & x^5 = (11,0)^T & x^6 = (10,-1)^T & \\ x^7 = (-20,5)^T & x^8 = (-20,6)^T & x^9 = (-19,5)^T & x^{10} = (-19,6)^T. \end{array}$$

This is depicted in Fig. 1 where we intuitively recognize three separated subsets in X. Therefore, we propose a partition of X into the following three clusters:

$$C_1 = \{x^1, x^2, x^3\},\ C_2 = \{x^4, x^5, x^6\},\ C_3 = \{x^7, x^8, x^9, x^{10}\}.$$

In view of the aforementioned explanation by setting $k = 3$, we see immediately from Fig. 2 that it is possible to cover the whole dataset X by three boxes all of a side length 1 (or equivalently by $k = 3$ maximum norm balls all of a radius 0.5). Note that we cannot cover the whole dataset X by $k = 3$ boxes of a side length less than 1.

The globally minimal value of $\widetilde{P}$ is thus $v^\star = \frac{1}{2}$ with the corresponding optimal cluster centers $z^{1\star} = (0.5, 0.5)^T$, $z^{2\star} = (9.5, -0.5)^T$, $z^{3\star} = (-19.5, 5.5)^T$, which

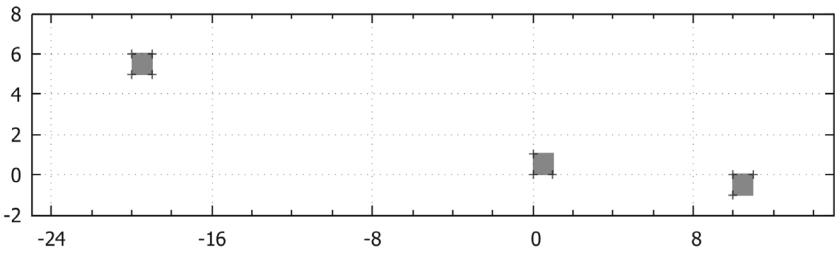


Fig. 2 Three boxes representing the global solution of the clustering problem $\widetilde{P}$

correspond to the centers of the gray-shaded boxes depicted in Fig. 2. From these considerations, it also becomes clear that the problem we solve differs from the k-means problem in the sense that optimal cluster centers are not mean values of the corresponding cluster points. The similarities between both problems, however, arise from the similar structure of the problem, with the only difference being the different norms that are used. Thus, let us stress that despite certain similarities, there are also crucial differences between the problem we solve and the k-means problem.

3 Relation to graph-theoretical concepts

In this section, we recall some concepts from graph theory and relate those to the aforementioned clustering problems. Clustering methods based on graph theory usually rely on the so-called distance matrices and in this regard, the concept is slightly more flexible compared to the distance-based k-means approaches. Note that our approach is still distance-based as already described in the previous section, since we are, in fact, using a highly specific distance matrix. Furthermore, it will become clear throughout this article that this is, indeed, crucial for our bisection method, which does not work for arbitrary distance matrices. The relation between the distance-based clustering problem and graph theory becomes clear from the following definition.

Definition 3.1 For some finite set $X \subseteq \mathbb{R}^n$ and some $\beta \geq 0$, let $G(\beta) = (X, E_\beta)$ denote an undirected graph with a set of nodes X and a set of edges E_β fulfilling

$$E_\beta = \{[x, y] \mid x, y \in X \text{ and } x \neq y \text{ and } \|x - y\|_\infty \leq \beta\}.$$

According to Definition 3.1, any graph $G(\beta) = (X, E_\beta)$ connects all data points from X which are pairwise located within a certain threshold distance β. If two given nodes $x, y \in X$ are too far away, i.e., $\|x - y\|_\infty > \beta$, then these nodes are not connected by an edge in the graph $G(\beta)$. To illustrate this, we revisit Example 2.1. Figure 3 illustrates the graph $G(2) = (X, E_2)$ for the corresponding set of data points.

A simple comparison of Figs. 2 and 3 reveals that the three clusters in the data set X identified as a global solution of the clustering problem $\widetilde{P}$ from Example 2.1 correspond to three such subsets of nodes in the graph $G(2) = (X, E_2)$ where nodes are all mutually connected by an edge in the graph $G(2)$. This gives rise to the following definition that is well known in graph theory.

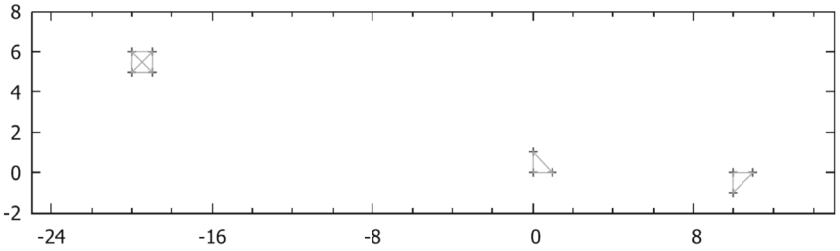


Fig. 3 Graph $G(2) = (X, E_2)$ based on the dataset X from Example 2.1

Definition 3.2 *(Clique)* Let an undirected graph $G = (X, E)$ be given. A subset $S \subseteq X$ is said to be a *clique* if and only if $[x, y] \in E$ for all $x, y \in S$ with $x \neq y$.

The next results reveal how the identification of the cliques of the graph $G(\beta) = (X, E_\beta)$ relates to solving the clustering problem P^∞.

Proposition 3.3 *Let $v^\star > 0$ denote the globally minimal value of the optimization problem $\widetilde{P}$ and let k denote the number of clusters. Then, for $\beta \geq 2v^\star$, there are k cliques $S_1, \ldots, S_k$ in $G(\beta) = (X, E_\beta)$ with*

$$\bigcup_{i=1}^{k} S_i = X.$$

Proof Let $(z^{1\star}, \ldots, z^{k\star}, v^\star)^T$ be a globally optimal point of $\widetilde{P}$. Then, as already discussed, there are clusters

$$C_i := \{x \in X \mid \|x - z^{i\star}\|_\infty \leq v^\star\}, \quad i = 1, \ldots, k$$

with $\bigcup_{i=1}^{k} C_i = X$. We show that every cluster C_i may serve as a clique in the graph $G(\beta) = (X, E_\beta)$. In fact, for $x, y \in C_i,\ i = 1, \ldots, k$ with $x \neq y$, we have

$$\|x - z^{i\star}\|_\infty \leq v^\star \text{ and } \|y - z^{i\star}\|_\infty \leq v^\star$$

and, thus, it holds $\|x - z^{i\star}\|_\infty + \|y - z^{i\star}\|_\infty \leq 2v^\star$. Applying the triangle inequality immediately yields

$$\|x - y\|_\infty \leq 2v^\star \implies \|x - y\|_\infty \leq \beta,$$

and so, $[x, y] \in E_\beta$ for all $x, y \in C_i$. Furthermore, we have $\bigcup_{i=1}^{k} C_i = X$ and so we put $S_i = C_i$ for $i = 1, \ldots, k$. Thus, we identified k cliques $S_1, \ldots, S_k$ in $G(\beta)$ which are sufficient to cover the whole dataset X, i.e., $\bigcup_{i=1}^{k} S_i = X$ and the assertion follows. □

So far, we know that each graph $G(\beta) = (X, E_\beta)$ for $\beta \geq 2v^\star$ contains k cliques which are sufficient to cover the whole dataset X where $v^\star$ denotes the globally minimal

value of the optimization problem $\widetilde{P}$. Next, we shall show that for $\beta < 2v^\star$ it is not possible to find k such cliques in the graph $G(\beta) = (X, E_\beta)$. Before proving this in Proposition 3.5, we first show the following auxiliary result stating that each clique in the graph $G(\beta) = (X, E_\beta)$ can be surrounded by a box of side length β.

Lemma 3.4 *Given $\beta \in \mathbb{R}$ and a set $S \subseteq \mathbb{R}^n$ with $\|x - y\|_\infty \leq \beta$ for all $x, y \in S$. Then, there is a box $B = [l_1, u_1] \times \cdots \times [l_n, u_n]$ with $S \subseteq B$ and $\max_{j=1,\ldots,n}(u_j - l_j) \leq \beta$.*

Proof We put

$$\begin{aligned} l_j &= \min\{x_j | x \in S\}, \quad j = 1, \ldots, n, \\ u_j &= \max\{x_j | x \in S\}, \quad j = 1, \ldots, n. \end{aligned}$$

Since

$$\|x - y\|_\infty \leq \beta \implies |x_j - y_j| \leq \beta$$

holds for all $x, y \in S$ and $j = 1, \ldots, n$, we obtain $u_j - l_j \leq \beta$ for all $j = 1, \ldots, n$. Finally the inclusion $S \subseteq B$ follows immediately from $l_j \leq x_j \leq u_j$ holding for all $j = 1, \ldots, n$ and $x \in S$. □

Now, we are ready to state the following result.

Proposition 3.5 *Let $v^\star > 0$ denote the globally minimal value of the optimization problem $\widetilde{P}$ and let k denote the number of clusters. Then, for $\beta < 2v^\star$, there are no k subsets $S_1, \ldots, S_k$ of X with*

$$\bigcup_{i=1}^{k} S_i = X,$$

such that all sets S_i, $i = 1, \ldots, k$ are cliques in the graph $G(\beta) = (X, E_\beta)$.

Proof We assume the existence of k cliques $S_1, \ldots, S_k$ in the graph $G(\beta)$ with $\bigcup_{i=1}^{k} S_i = X$ and derive a contradiction. Because of S_i, $i = 1, \ldots, k$ is assumed to be a clique in the graph $G(\beta) = (X, E_\beta)$, we have for all $x, y \in S_i$ the inequality $\|x - y\|_\infty \leq \beta$. Then, according to Lemma 3.4, there are boxes $B_i = [l_1^i, u_1^i] \times \cdots \times [l_n^i, u_n^i]$ with

$$\max_{j=1,\ldots,n} u_j^i - l_j^i \leq \beta < 2v^\star \quad \text{and} \quad S_i \subseteq B_i$$

for all $i = 1, \ldots, k$. We now put

$$z^i := \begin{pmatrix} \frac{l_1^i + u_1^i}{2} \\ \vdots \\ \frac{l_n^i + u_n^i}{2} \end{pmatrix} \quad \text{for } i = 1, \ldots, k.$$

That means for $x \in S_i \subseteq B_i$, we have

$$\|x - z^i\|_\infty = \max_{j=1,\ldots,n} \left| x_j - \frac{l_j^i + u_j^i}{2} \right|$$

$$
\begin{aligned}
&= \max_{j=1,\ldots,n} \max \left\{ x_j - \frac{l_j^i + u_j^i}{2}, \frac{l_j^i + u_j^i}{2} - x_j \right\} \\
&\leq \max_{j=1,\ldots,n} \max \left\{ u_j^i - \frac{l_j^i + u_j^i}{2}, \frac{l_j^i + u_j^i}{2} - l_j^i \right\} \\
&= \max_{j=1,\ldots,n} \max \left\{ \frac{u_j^i}{2} - \frac{l_j^i}{2}, \frac{u_j^i}{2} - \frac{l_j^i}{2} \right\} \\
&= \max_{j=1,\ldots,n} \frac{u_j^i - l_j^i}{2} \\
&\leq \frac{\beta}{2} < v^\star.
\end{aligned}
$$

Hence, we have $\|x - z^i\|_\infty \leq \frac{\beta}{2}$ for all $x \in S_i$, $i = 1, \ldots, k$ and, thus, $(z^1, \ldots, z^k, \frac{\beta}{2})^T$ is a feasible point of the optimization problem $\widetilde{P}$ with objective function value $\frac{\beta}{2} < v^\star$. This, however, contradicts $v^\star$ being a globally minimal value of $\widetilde{P}$. □

Results of this section enable us to develop an algorithm which approximates the globally minimal value $v^\star$ of the optimization problem $\widetilde{P}$ by means of a bisection method.

4 A global bisection method for solving the clustering problem

In this section, we describe our new bisection method for solving clustering problems. Moreover, we give a proof of convergence and comment on the maximum number of iterations.

4.1 Description of the algorithm and proof of convergence

The main idea is to start with an initial guess v_0 of the globally minimal value $v^\star$ of the problem $\widetilde{P}$. If there is no partition of the graph $G(2v_0)$ into k cliques $S_1, \ldots, S_k$ with $\bigcup_{i=1}^k S_i = X$, then, according to Proposition 3.3, we have $v_0 < v^\star$ and, thus, we have to increase our initial guess v_0 to some value $v_1 > v_0$. Additionally, we use v_0 as a new lower bound at the globally minimal value $v^\star$ of the problem $\widetilde{P}$.

Otherwise, if, in the graph $G(2v_0)$, there is such a partition into k cliques, then in view of Proposition 3.5, we might want to decrease our initial guess v_0 to some value $v_1 < v_0$ as a new approximation of the globally minimal value $v^\star$. Moreover, we may use v_0 as a new upper bound at the globally minimal value $v^\star$ of the problem $\widetilde{P}$.

These considerations lead to our new bisection method. Given a set of data points $X = \{x^1, \ldots, x^m\}$, a number of clusters $k \in \mathbb{N}$ and some initial lower bound $\underline{v}^0$, and an initial upper bound $\overline{v}^0$ at the value $2v^\star$, we can compute globally optimal values of a clustering problem as described formally in Algorithm 1. Appropriate initial values for lower and upper bounds are proposed below in Propositions 4.2 and 4.3, respectively.

Algorithm 1: Algorithm to solve the optimization problem $\widetilde{P}$ to global optimality

Data: $X = \{x^1, \dots, x^m\} \subseteq \mathbb{R}^n$.
Input: number of clusters $k \in \mathbb{N}$, termination tolerance $\varepsilon > 0$, initial lower bound $\underline{v}^0$ at $2v^\star$, initial upper bound $\overline{v}^0$ at $2v^\star$, $\lambda \in (0, 1)$.
Result: an ε-approximation of the globally minimal value $v^\star$ of $\widetilde{P}$ in the form of an interval $I := [\underline{v}^{\zeta^\star}, \overline{v}^{\zeta^\star}]$ with $\overline{v}^{\zeta^\star} - \underline{v}^{\zeta^\star} < \varepsilon$ and $2v^\star \in I$.

1 Set iteration counter $\zeta := 0$;
2 **while** $(\overline{v}^\zeta - \underline{v}^\zeta \geq \varepsilon)$ **do**
3 Choose $v^\zeta := \lambda \cdot \underline{v}^\zeta + (1 - \lambda) \cdot \overline{v}^\zeta$;
4 Generate graph $G(v^\zeta) = (X, E_{v^\zeta})$;
5 Try to find k cliques $S_1, \dots, S_k$ in $G(v^\zeta)$ with $\bigcup_{i=1}^k S_i = X$;
6 **if** *(X can be partitioned into such k cliques $S_1, \dots, S_k$)* **then**
7 $\overline{v}^{\zeta+1} := v^\zeta$;
8 $\underline{v}^{\zeta+1} := \underline{v}^\zeta$;
9 **else**
10 $\underline{v}^{\zeta+1} := v^\zeta$;
11 $\overline{v}^{\zeta+1} := \overline{v}^\zeta$;
12 **end**
13 Increment iteration counter ζ;
14 **end**
15 X can be partitioned into cliques $S_1, \dots, S_k$ in the graph $G(\overline{v}^\zeta)$;

In line 3, we update our approximation of the value $2v^\star$ in each iteration ζ. Given an interval $[\underline{v}^\zeta, \overline{v}^\zeta]$ containing the value $2v^\star$, in every iteration, we choose a new value v^ζ by setting

$$v^\zeta := \lambda \cdot \underline{v}^\zeta + (1 - \lambda) \cdot \overline{v}^\zeta$$

for some value $\lambda \in (0, 1)$. As usual for bisection methods, different values for λ are possible. Choosing $\lambda = 0.5$ has an advantage that the number of iterations of Algorithm 1 depends solely on the initial gap $\overline{v}^0 - \underline{v}^0$ and on the value of the termination criterion $\varepsilon > 0$. In such a case, irrespective of whether we update the lower bound $\underline{v}^\zeta$ or the upper bound $\overline{v}^\zeta$ in an iteration ζ, the gap $\overline{v}^\zeta - \underline{v}^\zeta$ is reduced by half in each iteration in contrast to other possible strategies of choosing the value $\lambda \in (0, 1)$.

At line 4, we consider the graph $G(v^\zeta) = (X, E_{v^\zeta})$, and at line 5, we decide if there is a partition of X into k cliques in the graph $G(v^\zeta)$ or not. This is done by solving the clique and the k-cover set subproblems for the graph $G(v^\zeta)$ which we discuss in detail in Sect. 5.

By means of Propositions 3.3 and 3.5, we can then update either the upper or the lower bound. In fact, in each iteration ζ, the gap between the upper bound $\overline{v}^\zeta$ and the lower bound $\underline{v}^\zeta$ is reduced, and finally, due to a constant parameter λ, both bounds converge toward each other. We summarize this result in the next theorem.

Theorem 4.1 *The sequences $(\underline{v}^\zeta)_{\zeta \in \mathbb{N}}$ and $(\overline{v}^\zeta)_{\zeta \in \mathbb{N}}$ of the non-terminating Algorithm 1 with $\varepsilon = 0$ both converge to the value $2v^\star$ where $v^\star$ denotes the globally minimal value of the optimization problem $\widetilde{P}$.*

Proof The proof follows immediately from Propositions 3.3 and 3.5 as well as from the theory of bisection methods. Indeed, the initial interval $[\underline{v}^0, \overline{v}^0]$ is divided, such

that the length of the interval decreases. In fact, we have

$$\overline{v}^{\zeta+1} - \underline{v}^{\zeta+1} \quad \leq \quad \max\{\lambda, 1-\lambda\}\left(\overline{v}^{\zeta} - \underline{v}^{\zeta}\right). \tag{1}$$

Due to $0 < \lambda < 1$, we obtain $\lim_{\zeta\to\infty}\left(\overline{v}^{\zeta} - \underline{v}^{\zeta}\right) = 0$. In addition, in view of Propositions 3.3 and 3.5, it is ensured that we always have $\underline{v}^{\zeta} \leq 2v^* \leq \overline{v}^{\zeta}$ for every iteration $\zeta \in \mathbb{N}$ and, thus, the assertion follows. □

As soon as the gap between the lower bound and the upper bound at the value $2v^\star$ becomes small enough at some iteration $\zeta^\star \in \mathbb{N}$, i.e., $\overline{v}^{\zeta^\star} - \underline{v}^{\zeta^\star} < \varepsilon$, the algorithm terminates.

Clearly, in addition to the interval $[\underline{v}^{\zeta^\star}, \overline{v}^{\zeta^\star}]$ approximating our target value $2v^\star$, we are also interested in a corresponding feasible point of the clustering problem $\widetilde{P}$. To achieve this, we consider the graph again. By Algorithm 1, this graph $G(\overline{v}^{\zeta^\star})$ contains k cliques which partition the set of data points X according to Proposition 3.3. Thus, we start with k cliques $S_1, \ldots, S_k$ in $G(\overline{v}^{\zeta^\star}) = (X, E_{\overline{v}^{\zeta^\star}})$ fulfilling $\bigcup_{i=1}^{k} S_i = X$. For all $i = 1, \ldots, k$ and for all $x, y \in S_i$, we have $[x, y] \in E_{\overline{v}^{\zeta^\star}}$, and thus, $\|x - y\|_\infty \leq \overline{v}^{\zeta^\star}$. Then, according to Lemma 3.4, by setting

$$\begin{aligned} l_j^i &= \min\{x_j | x \in S_i\}, \quad j = 1, \ldots, n, \\ u_j^i &= \max\{x_j | x \in S_i\}, \quad j = 1, \ldots, n, \end{aligned} \tag{2}$$

for each $i = 1, \ldots, k$, there are k boxes $B_i = [l_1^i, u_1^i] \times \cdots \times [l_n^i, u_n^i]$ with the property $\max_{j=1,\ldots,n}(u_j^i - l_j^i) \leq \overline{v}^{\zeta^\star}$ and $S_i \subseteq B_i$ for $i = 1, \ldots, k$. Letting

$$z^{i^\star} := \begin{pmatrix} \frac{l_1^i + u_1^i}{2} \\ \vdots \\ \frac{l_n^i + u_n^i}{2} \end{pmatrix} \quad \text{for } i = 1, \ldots, k, \tag{3}$$

we may now estimate the distance $\|x - z^{i^\star}\|_\infty$ as already done in the proof of Proposition 3.5. Hence, for each $x \in S_i \subseteq B_i$, it holds

$$\|x - z^{i^\star}\|_\infty \leq \frac{\overline{v}^{\zeta^\star}}{2}, \quad i = 1, \ldots, k,$$

and, thus, a corresponding feasible point of the clustering problem $\widetilde{P}$ is given by

$$\left(z^{1^\star}, \ldots, z^{k^\star}, \frac{\overline{v}^{\zeta^\star}}{2}\right)^T. \tag{4}$$

To apply Algorithm 1, initial valid upper and lower bounds at the globally optimal value are required. We now propose suitable possibilities to achieve this. Note that in the following result, the slightly unusual yet natural assumption $k < m$ is imposed.

Proposition 4.2 *Let $v^\star$ denote the globally minimal value of $\widetilde{P}$ and, moreover, let $k < m$. Then, a valid initial lower bound $\underline{v}^0$ at the value $2v^\star$ is given by*

$$\underline{v}^0 = \min_{\substack{x,y \in X \\ x \neq y}} \|x - y\|_\infty.$$

Proof The main idea of the proof is to derive a contradiction by showing that with $\beta < \underline{v}^0$, we need too many cliques to cover the graph $G(\beta)$, whereas from Proposition 3.3, we know that for β sufficiently large, this is not needed. Suppose it holds

$$\beta < \underline{v}^0 = \min_{\substack{x,y \in X \\ x \neq y}} \|x - y\|_\infty$$

for some $\beta \in \mathbb{R}$. Then, for two arbitrary points $x, y \in X$, $x \neq y$, we have

$$\|x - y\|_\infty \geq \underline{v}^0 > \beta,$$

and thus, each node of the graph $G(\beta) = (X, E_\beta)$ is isolated, i.e., $E_\beta = \emptyset$. Each clique in such a graph $G(\beta)$ contains at most one data point and, hence, m cliques are required to cover the dataset X. However, from Proposition 3.3, we know that for β sufficiently large, there are k cliques that cover the graph $G(\beta)$. Therefore, we have that

$$\underline{v}^0 = \min_{\substack{x,y \in X \\ x \neq y}} \|x - y\|_\infty$$

is a valid lower bound at the value $2v^\star$. □

Similarly, there is a simple possibility to initialize the upper bound at the globally optimal value.

Proposition 4.3 *Let $v^\star$ denote the globally minimal value of $\widetilde{P}$. A valid initial upper bound $\overline{v}^0$ at the value $2v^\star$ is given by*

$$\overline{v}^0 = \max_{x,y \in X} \|x - y\|_\infty.$$

Proof To show the result, we construct a feasible point of problem $\widetilde{P}$ that provides a sufficiently small objective value. This is achieved by letting

$$z_i^1 := \frac{\min_{x \in X} x_i + \max_{y \in X} y_i}{2} \qquad \text{and} \qquad z^j = 0 \text{ for } j = 2, \ldots, k.$$

Thus, for every $i = 1, \ldots, m$, we have

$$\|z^1 - x^i\| \leq \frac{\max_{x,y \in X} \|x - y\|_\infty}{2}.$$

In particular, this means that

$$\|z^{\sigma(i)} - x\|_\infty \leq \|z^1 - x^i\|_\infty \leq \frac{\overline{v}^0}{2}.$$

Therefore, in view of the definition of problem $\widetilde{P}$, we see that $\overline{v}^0/2$ is an upper bound at v^* and, thus, by $\overline{v}^*$, we obtain an upper bound at $2v^*$. □

4.2 Number of iterations and complexity of subproblems of Algorithm 1

In this section, we analyze our bisection method with regard to complexity considerations. We start by considering the number of iterations of our bisection method as given by Algorithm 1 for $\lambda = 0.5$. Depending on the initialization of the bounds $\overline{v}^0$, $\underline{v}^0$ and the termination criterion $\varepsilon > 0$, the number of iterations of our bisection method can be computed explicitly. The proof follows immediately using standard arguments of bisection methods. We present these considerations here for the sake of completeness.

Proposition 4.4 *Given a termination tolerance $\varepsilon > 0$, valid initial upper and lower bounds $\overline{v}^0$, $\underline{v}^0$ with $\overline{v}^0 - \underline{v}^0 \geq \varepsilon$, $\lambda = 0.5$ and let $\lceil \cdot \rceil$ denote the ceiling function. Then, the number of iterations of our bisection method in Algorithm 1 is at most*

$$\left\lceil \log_2 \left(\frac{\overline{v}^0 - \underline{v}^0}{\varepsilon} \right) + 1 \right\rceil.$$

Proof According to Proposition 3.3 and Proposition 3.5, in each iteration $\zeta \in \mathbb{N}$ of Algorithm 1, the gap $\overline{v}^\zeta - \underline{v}^\zeta$ is divided in half, that is

$$2(\overline{v}^{\zeta+1} - \underline{v}^{\zeta+1}) = \overline{v}^\zeta - \underline{v}^\zeta.$$

Let $\eta \in \mathbb{N}$ denote the iteration at which Algorithm 1 terminates. Then, we have $\overline{v}^\eta - \underline{v}^\eta < \varepsilon$ and the number of iterations can be determined exactly due to the equivalence

$$\begin{aligned} \frac{\overline{v}^0 - \underline{v}^0}{2^\eta} < \varepsilon &\iff \frac{\overline{v}^0 - \underline{v}^0}{\varepsilon} < 2^\eta \\ &\iff \log_2 \left(\frac{\overline{v}^0 - \underline{v}^0}{\varepsilon} \right) < \eta. \end{aligned}$$

Thus, after

$$\left\lceil \log_2 \left(\frac{\overline{v}^0 - \underline{v}^0}{\varepsilon} \right) + 1 \right\rceil$$

iterations, this is fulfilled and the algorithm terminates. □

Therefore, for $\lambda = 0.5$, the number of iterations of Algorithm 1 is logarithmic with respect to the gap $\overline{v}^0 - \underline{v}^0$. In view of the improved upper bounds as proposed in Sect. 5.3.4, there may be even fewer iterations in many cases.

It is interesting to note that neither the number of data points $m \in \mathbb{N}$, nor the number of clusters $k \in \mathbb{N}$, nor the dimension of the data points $n \in \mathbb{N}$ affects the number of iterations. However, the positions of the data points in $\mathbb{R}^n$ can be expected to have a direct impact, since the greatest and the smallest distance between two arbitrary data points in X affects the initialization of the bounds $\overline{v}^0$ and $\underline{v}^0$ as proposed in Propositions 4.2 and 4.3.

Next, we consider a single iteration $\zeta \in \mathbb{N}$ of Algorithm 1 and we discuss the complexity of the subproblems that need to be solved. As described throughout this article, we have to determine maximal cliques in the graph $G(v^\zeta)$ and, in case there are more than k maximal cliques, we also have to solve the k-cover set problem accordingly. Therefore, we take a look at the impact on the runtime through the number of data points $m \in \mathbb{N}$, the number of clusters $k \in \mathbb{N}$, and the dimension of the data points $n \in \mathbb{N}$.

The dimension n of the given data only influences the runtime needed to compute the distances

$$\|x - y\|_\infty = \max_{i=1,\dots,n} |x_i - y_i|$$

between two arbitrary data points $x, y \in X$. This lies within $\mathcal{O}(n)$, and since this has to be done for every pair of data points, thus, we have $\mathcal{O}(nm^2)$. However, this procedure only needs to be performed once and can be done in advance before executing Algorithm 1.

The number of data points $m \in \mathbb{N}$ also plays an important role during the computation of all maximal cliques by means of the Bron–Kerbosch algorithm. The recursive tree calls that are needed within this procedure increase with an increasing number of data points m. Furthermore, the maximal clique problem is known to be $\mathcal{NP}$-hard (see Karp 1972), and thus, this cannot be expected to be done efficiently.

In addition, if the number of maximal cliques $|\mathcal{S}|$ in the graph $G(v^\zeta)$ fulfills $k < |\mathcal{S}|$, we also have to consider the $\mathcal{NP}$-hard k-cover set problem accordingly. Fortunately, these are only worst-case considerations and, moreover, our numerical experiments show that upon implementation of the acceleration strategies presented in Sect. 5.3, we are able to solve clustering problems of reasonable sizes to global optimality.

Remark 4.5 We emphasize that despite the good upper bounds at the number of iterations from Proposition 4.4, still, the computational effort that might be required in every iteration can be large due to $\mathcal{NP}$-hard problems that need to be solved. Therefore, the worst-case performance must be expected to be bad.

It shall become clear from the numerical results that the scenarios leading to unmanageable optimization problems for large instances mostly deal with data sets exhibiting no clear cluster-pattern structure which makes them rather unsuitable for clustering considerations, such as scenarios with only poor (or no) cluster pattern at all. Similar observations are common in the field of global optimization, where there is often a bad worst-case performance that needs to be avoided whenever possible. Branch-

and-bound algorithms that are common in that area are, for instance, based on such a concept as well.

5 Solving the subproblems

Algorithm 1 requires at line 5 the computation of cliques for a given undirected graph $G(v^\zeta)$ as well as to evaluate whether there are k such cliques which cover the entire set of nodes X of $G(v^\zeta)$ or not. This leads to the so-called clique as well as the so-called k-cover set problems that frequently arise in graph theory and will be briefly discussed in the following two sections. For a more detailed description of these graph-theoretical problems, we refer to the literature, e.g., Bondy and Murty (1976); Diestel (2000); West (2001) and the references therein.

5.1 Computing appropriate cliques

Since the number of cliques in the graph $G(v^\zeta)$ that needs to be considered in each iteration of Algorithm 1 may be rather large, we propose an approach which considers only so-called *maximal cliques*.

Definition 5.1 (see Akkoyunlu 1973) Let an undirected graph $G = (X, E)$ and the set of all its cliques $\mathcal{C}$ be given. A clique $S \in \mathcal{C}$ is said to be *maximal* if and only if $S \not\subset S^\star$ for all $S^\star \in \mathcal{C}$.

In other words, the clique $S \in \mathcal{C}$ is maximal if and only if there is no real superset of S denoted by $S^\star$ which is also a clique in the graph G. In the following two lemmae, it is shown that in Algorithm 1, it is, in fact, sufficient to consider only maximal cliques in the graph $G(v^\zeta)$ in each iteration $\zeta \in \mathbb{N}$.

Lemma 5.2 *Let a set of data points X and a set of cliques $S_1, \ldots, S_k$ in the graph $G = (X, E)$ with*

$$\bigcup_{i=1}^{k} S_i = X.$$

be given. Then, there are also k maximal cliques in $G = (X, E)$ denoted by $S_1^\star, \ldots, S_k^\star$ with $S_i \subseteq S_i^\star$ for all $i = 1, \ldots, k$ fulfilling $\bigcup_{i=1}^{k} S_i^\star = X$.

Proof If S_i is not maximal in G then there is another clique $S_i^\star$ that is maximal and we have $S_i \subsetneqq S_i^\star$. For every set S_i that is maximal, we put $S_i^\star = S_i$. In summary, we have $S_i \subseteq S_i^\star \subseteq X$, and thus, it holds

$$\bigcup_{i=1}^{k} S_i = X \implies \bigcup_{i=1}^{k} S_i^\star = X.$$

□

Lemma 5.3 *Let a set of data points X be given, such that there is no partition of the set X into k sets $S_1, \ldots, S_k$ with $\bigcup_{i=1}^{k} S_i = X$ and all S_i, $i = 1, \ldots, k$ are cliques in $G = (X, E)$. Then, we cannot find any k maximal cliques $S_1^\star, \ldots, S_k^\star$ in $G = (X, E)$ with $\bigcup_{i=1}^{k} S_i^\star = X$.*

Proof Since each maximal clique $S_i^\star$ is a clique itself, we cannot find k maximal cliques in G which are sufficient to cover the dataset X. □

Let us stress that the maximal cliques $S_1^\star, \ldots, S_k^\star$ do not need to be mutually distinct, i.e., $S_i^\star = S_j^\star$ for $i \neq j$ is, in fact, possible. Similarly, in case a small number $\tilde{k} < k$ of maximal cliques is already sufficient to cover a graph $G = (V, E)$, we may define additional cliques $S_{\tilde{k}+1}^\star, \ldots, S_k^\star$ by, e.g., setting

$$S_i^\star := S_1^\star \quad \text{for} \quad i = \tilde{k} + 1, \ldots, k, \tag{5}$$

and so, we obtain k maximal cliques $S_1^\star, \ldots, S_k^\star$ with

$$\bigcup_{i=1}^{k} S_i^\star = X. \tag{6}$$

In summary, we can partition the set of nodes X of a graph $G(\beta) = (X, E_\beta)$ into k cliques $S_1, \ldots, S_k$ in $G(\beta)$ if and only if there are less or equal than k maximal cliques in the graph $G(\beta)$ which are sufficient to cover the set X. Therefore, it is sufficient to consider only maximal cliques in a graph $G(\beta) = (X, E_\beta)$ within our bisection method. The number of maximal cliques in a graph is much smaller than the number of cliques, because each subset of a maximal clique is a clique itself. For determining all maximal cliques of a given undirected graph algorithmically, we use the Bron–Kerbosch algorithm introduced in Bron and Kerbosch (1973). In the following, we denote the set of all maximal cliques in a graph $G(\beta) = (\bar{X}, E_\beta)$ by $\Phi_{\bar{X}}(\beta)$ where $\bar{X}$ denotes some non-empty subset of X.

Even though the maximal clique problem is $\mathcal{NP}$-hard as shown in Karp (1972) and, thus, cannot be expected to be solved efficiently within our bisection method, the Bron–Kerbosch algorithm can be accelerated by exploiting knowledge of previous iterations as we shall discuss in Sect. 5.3.

5.2 Solving the *k*-cover set problem

To successfully implement lines 5 and 6 of Algorithm 1, it is still required to have a suitable procedure to decide whether in the given set of all maximal cliques $\mathcal{S}$ of the graph $G(v^\zeta) = (X, E_{v^\zeta})$, there are k maximal cliques $S_1, \ldots, S_k \in \mathcal{S}$ that cover the whole set X, i.e., which fulfill $\bigcup_{i=1}^{k} S_i = X$. This is the well-known k-cover set problem as discussed, for instance, in Balas and Padberg (1972). Its complexity aspects have already been examined in Karp (1972), for instance. Clearly, if the number of maximal cliques $|\mathcal{S}|$ in the graph $G(v^\zeta)$ is less than or equal to k, i.e., $|\mathcal{S}| \leq k$, then there is a partition of X into k cliques of the graph $G(v^\zeta)$ which cover the set X.

For the case $|\mathcal{S}| > k$, such a straightforward conclusion cannot be drawn and, thus, a suitable method to decide whether there are k maximal cliques $S_1, \ldots, S_k \in \mathcal{S}$ with $\bigcup_{i=1}^{k} S_i = X$ is needed. To this end, we apply an integer linear problem used, e.g., in Balas and Padberg (1972), where we use binary decision variables $s_1, \ldots, s_{|\mathcal{S}|}$ which correspond to the maximal cliques $S_1, \ldots, S_{|\mathcal{S}|}$. A feasible solution with $s_i = 1$ for an $i \in \{1, \ldots, |\mathcal{S}|\}$ means that the maximal clique S_i is considered as a cluster that is needed to cover the graph, and otherwise, it is not. In contrast, if we have a feasible solution with $s_i = 0$ for an arbitrary $i \in \{1, \ldots, |\mathcal{S}|\}$, the maximal clique S_i is not considered as a cluster. The integer linear problem is given as follows:

$$
\begin{aligned}
ILP: \quad & \min_{s_1,\ldots,s_{|\mathcal{S}|}} \sum_{i=1}^{|\mathcal{S}|} s_i \\
\text{s.t.} \quad & \sum_{\substack{i: x \in S_i, \\ i=1,\ldots,|\mathcal{S}|}} s_i \geq 1, \qquad x \in X, \\
& \sum_{i=1}^{|\mathcal{S}|} s_i \leq k \\
& s_i \in \{0, 1\}, \quad i = 1, \ldots, |\mathcal{S}|.
\end{aligned}
$$

Note that the original problem from Balas and Padberg (1972) is extended by the second constraint

$$\sum_{i=1}^{|\mathcal{S}|} s_i \leq k$$

to ensure that the problem has only a feasible solution if there are at most k maximal cliques $S_1, \ldots, S_k \in \mathcal{S}$ with $\bigcup_{i=1}^{k} S_i = X$.

Finally, let us stress that for solving the clustering problem globally using our bisection Algorithm 1, it is required to determine all maximal cliques in the graph $G(v^\zeta)$ in each iteration $\zeta \in \mathbb{N}$. This problem is already $\mathcal{NP}$-hard. In addition, we have to decide whether a covering of X by k maximal cliques $S_1, \ldots, S_k$ in the graph $G(v^\zeta)$ is possible. This is known to be $\mathcal{NP}$-hard as well. Therefore, we cannot expect to apply polynomial time algorithms within our framework. For this reason, it is crucial to have suitable acceleration strategies that help to avoid worst-case performance for practical problems as is common in many procedures in global optimization in general. This is addressed in the next section.

5.3 Acceleration steps

In the following, we propose acceleration steps to improve the runtime of the subproblems arising in Algorithm 1, especially that of Bron–Kerbosch algorithm which determines all maximal cliques in each iteration of our bisection method. Although the aforementioned methods to solve the subproblems are standard procedures, these algorithms can be tailored to our needs by means of the following techniques. Thanks

to these strategies, the clustering problem can be solved globally in a reasonable time as can be seen from the numerical experiments in Sect. 6. This is discussed in the remainder of this section.

5.3.1 Using knowledge of previous iterations

Let us assume for a moment that all maximal cliques in a graph $G(\tilde{\beta}) = (X, E_{\tilde{\beta}})$ for some $\tilde{\beta} > 0$ are already known. Then, considering another graph $G(\beta) = (X, E_\beta)$ with $\beta \leq \tilde{\beta}$, according to Definition 3.1 and Definition 5.1, for two arbitrary nodes $x, y \in X$, we have

$$[x, y] \in E_\beta \implies [x, y] \in E_{\tilde{\beta}}.$$

Thus, any maximal clique S in the graph $G(\beta)$ is also a clique in the graph $G(\tilde{\beta})$ for $\beta \leq \tilde{\beta}$, although not necessarily a maximal one. Hence, in Algorithm 1, we may use the set of maximal cliques in the graph $G(\tilde{\beta})$ as a superset of the set of maximal cliques in a graph $G(\beta)$ in that case.

To take advantage of this observation, each time, we update the upper bound $\overline{v}^\zeta$ at line 11 in Algorithm 1 we save the maximal cliques which are already determined at line 6. Then, instead of calculating the set of maximal cliques $\Phi_X(v^\zeta)$ from scratch, we may determine the set of maximal cliques $\Phi_S(v^\zeta)$ for each $S \in \Phi_X(\overline{v}^\zeta)$, since the set of maximal cliques $\Phi_X(\overline{v}^\zeta)$ is already known from a previous iteration due to the inequality $v^\zeta \leq \overline{v}^\zeta$. This is a straightforward possibility to exploit knowledge of previous iterations within our bisection method.

5.3.2 Using supersets of maximal cliques

Considering a graph $G(\beta) = (X, E_\beta)$ with k cliques $S_1, \ldots, S_k$ fulfilling $\bigcup_{i=1}^k S_i = X$, we know that each point $x \in X$ is in at least one of the sets S_i, $i = 1, \ldots, k$. Then, another point $y \in X$ can be contained in the same clique only if we have

$$\|x - y\|_\infty \leq \beta.$$

This enables us to construct supersets of maximal cliques from the perspective of each individual data point $x \in X$ as given in the following definition.

Definition 5.4 Given some data set $X = \{x^1, \ldots, x^m\}$ and a graph $G(\beta) = (X, E_\beta)$ according to some parameter $\beta \geq 0$. Then, for $i = 1, \ldots, m$, we put

$$G_i := \{y : [x^i, y] \in E_\beta\} \cup \{x^i\}.$$

Each set G_i from Definition 5.4 contains a data point $y \in X$ if and only if $\|x^i - y\|_\infty \leq \beta$. In case of $\|x^i - y\|_\infty > \beta$, the data points x^i and y cannot be contained in any common clique S in the graph $G(\beta) = (X, E_\beta)$. For that reason, for each clique S in the graph $G(\beta) = (X, E_\beta)$ with $x^i \in S$, we have immediately $S \subseteq G_i$.

Thus, instead of calculating the set of all maximal cliques in $G(\beta) = (X, E_\beta)$, it may be already sufficient to calculate the set of maximal cliques $\Phi_{G_i}(\beta)$ for each

$i = 1, \ldots m$, which can help to significantly reduce runtime, since each of the sets G_i may only contain a small fraction of all given data points X.

Nevertheless, every set G_i is only a superset for each maximal clique in the graph $G(\beta) = (X, E_\beta)$ containing the point x^i and it still may include points $y, z \in X$ with $\|y - z\|_\infty > \beta$, while

$$\|x^i - y\|_\infty \leq \beta \qquad \text{and} \qquad \|x^i - z\|_\infty \leq \beta$$

is fulfilled. Still, the sets G_i are suitable to initialize the Bron–Kerbosch algorithm instead of always starting with the whole set of data points X from scratch.

5.3.3 Alternative detection of lower bounds

Let us consider a graph $G(\beta) = (X, E_\beta)$ with $\beta \geq 0$ at some iteration of Algorithm 1 and, moreover, let there be a set $L \subseteq X$ with $|L| > k$, so that we have

$$\|x - y\|_\infty > \beta \quad \forall\, x, y \in L \text{ with } x \neq y.$$

Then, we need at least $|L| > k$ cliques to cover the whole dataset X, and according to Proposition 3.3, the lower bound at the value $2v^\star$ can be increased to β immediately.

For some cases, this provides a quick possibility to determine lower bounds at the optimal value v^* without the need to compute maximal cliques and without solving the k-cover set problem.

5.3.4 Improving upper bounds

Similarly to the previous section, there is a possibility to accelerate the runtime of Algorithm 1 by improving upper bounds at the value $2v^\star$. To this end, let us assume that at some iteration $\zeta \in \mathbb{N}$, Algorithm 1 identifies at line 5 a number of k cliques $S_1, \ldots, S_k$ in $G(v^\zeta)$ with $\bigcup_{i=1}^k S_i = X$. Then, instead of updating the upper bound to the value v^ζ according to line 7 of Algorithm 1, we put

$$\overline{v}^{\zeta+1} := \max_{i=1,\ldots,k} \max_{x,y \in S_i} \|x - y\|_\infty. \tag{7}$$

Clearly, each of the cliques S_i, $i = 1, \ldots, k$ in the graph $G(v^\zeta)$ is also a clique in the graph $G(\overline{v}^{\zeta+1})$ due to

$$\|p - q\|_\infty \leq \max_{x,y \in S_i} \|x - y\|_\infty \leq \overline{v}^{\zeta+1} \quad \forall p, q \in S_i \quad \forall i = 1, \ldots, k.$$

Since $\bigcup_{i=1}^k S_i = X$ is fulfilled, in view of Proposition 3.3 the value $\overline{v}^{\zeta+1}$ is still a valid upper bound at the value $2v^\star$. Furthermore, the inequality $\overline{v}^{\zeta+1} \leq \overline{v}^\zeta$ holds, and, in addition, the upper bound update from Eq. (7) is better than the original one as proposed in line 7 of Algorithm 1. Again, this can help to improve the overall performance of our bisection method.

5.3.5 Calculation of a feasible solution using k-means algorithm

In addition to the aforementioned steps, we may also apply the well-known k-means algorithm to our set of data points X. As a result, we obtain a feasible solution for the clustering problem based on Euclidean distances However, this also yields k clusters $S_1, \ldots, S_k$ with $\bigcup_{i=1}^{k} S_i = X$. From this feasible point, we may derive a solution for our clustering problem $\widetilde{P}$ with objective value

$$\max_{i=1,\ldots,k} \max_{x,y \in S_i} \frac{\|x - y\|_\infty}{2}.$$

Since $\bigcup_{i=1}^{k} S_i = X$ is fulfilled, in view of Proposition 3.3, this also yields a valid upper bound at the value $v^\star$.

6 Numerical results

In this section, we present the results of our numerical experiments. In this section, our implementation of our clustering approach is described. The algorithm is developed in Python. A machine with an Intel(R) Core(TM) i7-8650U CPU @ 1,9 GHz processor and 32 GB RAM running Windows is used for the computation. The application is executed in a Jupyter Notebook launched in an Anaconda environment.

In addition, we use the Python package NetworkX to create the graphs in each iteration and to solve the maximal clique problem with the Bron–Kerbosch algorithm. The set-cover problem ILP as well as the MILP reformulation of our clustering problem defined below is modeled and solved by means of the Python package Pulp with the default solver `PULP_CBC_CMD` as well as by employing the `Gurobi` solver. The k-means implementation from the Python package scikit-learn is used for the acceleration step of Sect. 5.3.5. Moreover, we use the package Numpy for efficient and comprehensive data preparation, and Matplotlib is used to visualize the clustering results. Note that the original k-means algorithm solves the classical MSSC problem with Euclidean distances, whereas our novel approach aims at the solution of a slightly different problem where distances are computed using the maximum norm. Therefore, it is clear that the solutions computed by both approaches may differ. Still, both problems share certain similarities and, thus, k-means may be considered as a suitable heuristic to also solve the clustering problem with maximum norm. This is also the reason why we do report the corresponding results. Moreover, it is also clear that the k-means algorithm is typically much faster than our new approach, since the former aims at local solutions, whereas the latter does guarantee a global solution.

Our algorithm is applied to synthetic data in Sect. 6.1 as well as to real data in Sect. 6.2.

6.1 Clustering on artificial data

To examine the runtime of Algorithm 1 with respect to the number of data points m and dimension n, we test our bisection method for various data instances with $k = 3$ and

$k = 5$ clusters. For both cases $k = 3$ and $k = 5$, we generate k clusters by choosing $\frac{m}{k}$ independent points from the n-dimensional Gaussian distribution $\mathcal{N}(z^i, \sigma)$, $i = 1, \ldots, k$ for different values of $\sigma > 0$ with centers

$$z^1 = (-50, -50, \ldots)^T,\ z^2 = (0, 0, \ldots)^T,\ z^3 = (50, 50, \ldots)^T \in \mathbb{R}^n$$

for $k = 1, 2, 3$ and

$$z^4 = (-50, 50, -50, 50, \ldots)^T,\ z^5 = (50, -50, 50, -50, \ldots)^T \in \mathbb{R}^n$$

for $k = 4, 5$. Figure 4 illustrates such generated data sets $X \subseteq \mathbb{R}^2$ with $m = 1500$ points for various values of $\sigma > 0$.

Additionally, to illustrate both, the maximal size and the level of separation among all generated clusters, we consider the greatest distance between two data points within one cluster

$$\Delta_I := \max_{i=1,\ldots,k} \max_{x,y \in C_i} \|x - y\|_\infty$$

and the smallest distance between two data points from different clusters

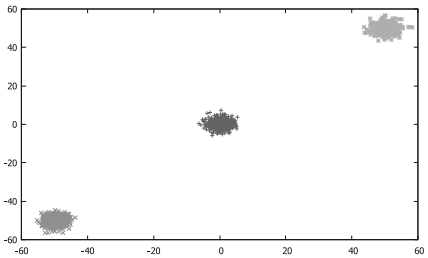
$$\Delta_O := \min_{i \neq j} \min_{x \in C_i} \min_{y \in C_j} \|x - y\|_\infty.$$

We set the termination tolerance to $\varepsilon = 10^{-7}$ and apply Algorithm 1 to the different sets of data points which are generated as described above.

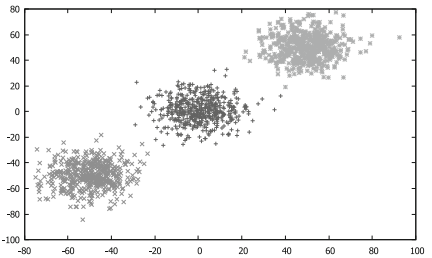
Since some very preliminary numerical tests indicated that a value of $\lambda = 0.8$ is much better suited than, for instance, $\lambda = 0.5$, we use the update strategy $v^\zeta := 0.8\underline{v}^\zeta + 0.2\overline{v}^\zeta$ which leads to a very good performance due to better working superset approximations G_i for $i = 1, \ldots, m$ at the maximal cliques in the graph $G(v^\zeta)$.

We compare our method to the well-known k-means algorithm. Note, however, that k-means only computes locally optimal points of the clustering problem with Euclidean distances, whereas we strive for a globally optimal solution of the clustering problem $\widetilde{P}$ using the maximum metric. To better compare both results, we compute two variants of the k-means. In the first one, we use the classical Euclidean-norm based k-means algorithm where the computed optimal points are finally plugged into the maximum norm objective function (denoted by $f_k^\star$ below). In addition, we also apply a k-means like algorithm which is defined completely in the maximum norm-setting (the corresponding value of the computed optimal objective is denoted by $f_{k,\infty}^*$ below). Thus, all steps are analogously to the classical k-means algorithm, where, however, the Euclidean distances are replaced by distances that are computed based on the maximum norm. More precisely, we apply a two-phase algorithm alternating the computation of the centroids and then assigning the points to the centers. In this case, however, centroids correspond to centers of the boxes and assignment of data points to the closest center is determined using the maximum norm.

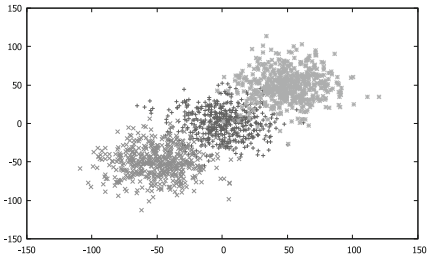
Our numerical results with $k = 3$ and $k = 5$ clusters for various data instances are presented in Tables 1 and 2. Here, we consider the case of clearly separated clusters by choosing $\sigma = 2$. The runtime of Algorithm 1 is given in column "t", whereas the



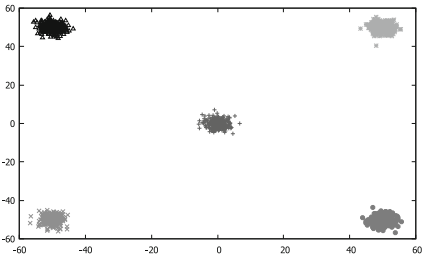
(a) X with $m = 1500$, $k = 3$, $\sigma = 2$



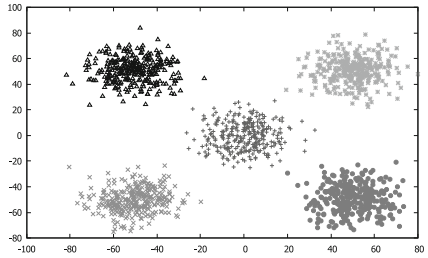
(b) X with $m = 1500$, $k = 3$, $\sigma = 10$



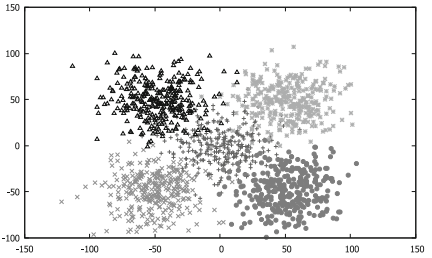
(c) X with $m = 1500$, $k = 3$, $\sigma = 20$



(d) X with $m = 1500$, $k = 5$, $\sigma = 2$



(e) X with $m = 1500$, $k = 5$, $\sigma = 10$



(f) X with $m = 1500$, $k = 3$, $\sigma = 20$

Fig. 4 Generated data set $X \subseteq \mathbb{R}^2$ with $k = 3$ and $k = 5$ clusters for varying values of $\sigma > 0$

runtime of the classical Euclidean-norm-based k-means is presented in column "t_k". Both runtime columns contain a pair of values, whereas the first value corresponds to the runtime obtained by employing the `PULP_CBC_CMD` solver and the second value corresponds to the runtime obtained from `Gurobi`. The value $f^\star$ denotes the optimal value of the objective function of $\widetilde{P}$ as computed by Algorithm 1, whereas $\tilde{f}^*$ denotes the value obtained by plugging in the optimal clusterings as computed by Algorithm 1 into a Euclidean norm instead of the maximum norm.

From Tables 1 and 2, we can see immediately that by increasing the number m of data points, the runtime of Algorithm 1 increases as expected, but still performs

Table 1 Results for $k = 3$ clearly separated clusters, i.e., $\sigma = 2$

m	n	t [s]	Δ_I	Δ_O	f^*/f_k^*	t_k [s]	$\tilde{f}^*/f_k^*$	$f^*/f_{k,\infty}^*$
120	2	0.276; 0.18	9.24	45.03	0.925	0.270; 0.177	1.038	1
120	10^2	0.130; 0.105	13.31	53.78	0.818	0.127; 0.102	1.108	1
120	10^3	0.144; 0.105	13.81	56.66	0.811	0.138; 0.1	1.096	1
120	10^4	0.225; 0.239	15.3	58.64	0.816	0.200; 0.223	1.094	1
120	10^5	1.050; 0.9	16.39	60.68	0.802	0.832; 0.75	1.094	1
1500	2	0.616; 0.422	13.5	41.52	0.859	0.289; 0.167	1.127	1
1500	10^2	0.800; 0.469	15.26	53.23	0.88	0.149; 0.095	1.064	1
1500	10^3	0.636; 0.468	16.66	56.33	0.878	0.167; 0.14	1.066	1
3000	2	1.371; 0.922	14.22	40.92	0.904	0.345; 0.178	1.078	1
3000	10^2	1.375; 1.015	16.13	53.05	0.884	0.121; 0.098	1.059	1
3000	10^3	1.985; 1.504	17.08	56.14	0.858	0.204; 0.173	1.062	1

Table 2 Results for $k = 5$ clearly separated clusters, i.e., $\sigma = 2$

m	n	t [s]	Δ_I	Δ_O	f^*/f_k^*	t_k [s]	$\tilde{f}^*/f_k^*$	$f^*/f_{k,\infty}^*$
120	2	0.768; 0.248	9.24	45.01	0.978	0.757; 0.233	1.047	1
120	10^2	0.367; 0.108	13.23	54.06	0.835	0.357; 0.104	1.094	1
120	10^3	0.421; 0.209	13.97	56.84	0.84	0.41; 0.198	1.097	1
120	10^4	0.613; 0.183	15.3	58.87	0.819	0.554; 0.17	1.097	1
120	10^5	2.575; 0.932	16.39	60.64	0.804	2.107; 0.816	1.098	1
1500	2	1.806; 0.388	13.48	41.87	0.868	0.7; 0.165	1.104	1
1500	10^2	1.775; 0.409	15.26	53.23	0.875	0.384; 0.103	1.077	1
1500	10^3	2.107; 0.41	16.3	56.35	0.863	0.425; 0.141	1.071	1
3000	2	4.395; 0.896	13.79	40.16	0.878	0.712; 0.171	1.056	1
3000	10^2	7.565; 1.138	15.54	53.11	0.86	0.475; 0.103	1.067	1
3000	10^3	6.111; 1.262	16.66	56.13	0.831	0.755; 0.191	1.064	1

well. Additionally, unlike k-means, Algorithm 1 also ensures the global optimality of the computed solution. Even for larger instances such as $m = 3000$ data points in $n = 1000$ dimensions, our new approach computes a globally minimal solution of the clustering problem with $k = 5$ clusters within a few seconds.

Nevertheless, it is worth mentioning that the underlying clusters are strictly separated in our test problems. Indeed, according to Tables 1 and 2, the smallest distance between two data points of different clusters Δ_O is much larger than the greatest distance between two arbitrary data points within one cluster Δ_I meaning that there is a clear cluster-pattern structure. In fact, for $\Delta_I < \Delta_O$, we are able to find a value $\beta \in (\Delta_I, \Delta_O)$, such that the maximal cliques in the graph $G(\beta) = (X, E_\beta)$ are clearly separated, or in other words, there is no edge $[x, y] \in E_\beta$ which connects two different clusters due to $\beta < \Delta_O \leq \|x - y\|_\infty$ for $x \in C_i,\ y \in C_j$ with $i \neq j$.

Table 3 Results for different values of σ for $m = 120$ and $n = 2$

k	σ	t [s]	Δ_I	Δ_O	f^*/f_k^*	t_k [s]	$\tilde{f}^*/f_k^*$	$f^*/f_{k,\infty}^*$
3	2	0.346; 0.179	9.24	45.03	0.925	0.343; 0.176	1.038	1
3	5	0.124; 0.087	23.11	37.59	0.925	0.119; 0.083	1.038	1
3	10	0.118; 0.102	46.22	25.17	0.925	0.112; 0.097	1.038	1
3	20	0.146; 0.102	88.04	12.17	0.921	0.142; 0.098	1.042	1
3	30	8.577; 3.919	130.37	3.32	0.825	0.125; 0.09	1.725; 1.726	0.987
5	2	0.349; 0.23	9.24	45.01	0.978	0.346; 0.228	1.047	1
5	5	0.13; 0.097	23.11	37.52	0.978	0.126; 0.094	1.047	1
5	10	0.105; 0.092	46.22	25.04	0.978	0.102; 0.089	1.047	1
5	20	12.377; 6.572	84.2	7.13	0.849	0.122; 0.088	1.302; 1.408	0.956
5	30	11.789; 6.738	107.37	4.09	0.779	0.115; 0.088	1.439; 1.522	0.876

To analyze the performance of Algorithm 1 on data sets not necessarily exhibiting clear cluster-pattern structure, we examine the impact of the standard deviation σ on the runtime of Algorithm 1 for $k = 3$ and $k = 5$ clusters with $m = 120$ two-dimensional data points in Table 3. Due to increased standard deviation of the data points within each cluster, the clusters begin to overlap. Nevertheless, we still obtain good results for cases $\Delta_I < \Delta_O$. For $\Delta_I > \Delta_O$, we observe a significant increase in runtime of Algorithm 1 unlike in the case of k-means which seems to be not affected at all. However, also Algorithm 1 terminates in a rather small amount of time (a few seconds at most) due to the implemented acceleration steps from Sect. 5.3. Moreover, as can be seen from Table 3, despite slightly larger runtimes our bisection method Algorithm 1 always identifies a global solution in all cases, whereas the k-means sometimes terminates at locally optimal points.

Finally, we present the numerical results obtained from comparing the computational performance of our proposed method with a mixed-integer-linear programming (MILP) reformulation of the underlying clustering problem which we solve to global optimality by using the Python package PULP in Google Colab environment.

To this end, we consider the following well-known MILP reformulation of the clustering problem $\widetilde{P}$ from Sect. 2:

$$
\begin{aligned}
MILP: \quad \min_{z^1,\dots,z^k \in \mathbb{R}^n, \alpha \in \mathbb{R}, y^{ij} \in \{0,1\}} \quad & \alpha \\
\text{s.t.} \quad & z_l^j - x_l^i \leq \alpha + (1 - y^{ij}) \cdot M \;\; \forall i \in [m],\ j \in [k],\ l \in [n] \\
& -z_l^j + x_l^i \leq \alpha + (1 - y^{ij}) \cdot M \;\; \forall i \in [m],\ j \in [k],\ l \in [n] \\
& \sum_{j=1}^{k} y^{ij} = 1 \qquad \forall i \in [m]
\end{aligned}
$$

with some constant $M > 0$ sufficiently large and with $[i] := \{1, \dots, i\}$ for any $i \in \mathbb{N}$.

We conducted experiments again across various dimensions, number of clusters, number of data points, and standard deviation values σ. For each such parameter

Table 4 MILP reformulation vs Algorithm 1

n	k	m	σ	MILP time [s]			Algorithm 1 time [s]		
				Min	Avg	Max	Min	Avg	Max
4	3	21	5	1.526; 0.065	1.969; 0.097	2.202; 0.13	0.201; 0.201	0.274; 0.296	0.464; 0.534
4	3	21	10	1.377; 0.065	1.95; 0.094	2.272; 0.138	0.179; 0.217	0.202; 0.232	0.246; 0.273
4	3	21	15	1.606; 0.078	2.264; 0.084	3.034; 0.103	0.16; 0.177	0.214; 0.219	0.318; 0.25
4	3	30	5	2.28; 0.096	3.672; 0.126	6.916; 0.151	0.126; 0.183	0.199; 0.217	0.24; 0.253
4	3	30	10	2.944; 0.104	3.697; 0.133	4.556; 0.211	0.142; 0.192	0.19; 0.219	0.255; 0.241
4	3	30	15	2.669; 0.098	4.088; 0.115	6.344; 0.146	0.135; 0.162	0.185; 0.219	0.216; 0.278
4	4	28	5	6.184; 0.171	12.153; 0.251	25.068; 0.35	0.083; 0.24	0.178; 0.309	0.503; 0.557
4	4	28	10	6.826; 0.157	11.381; 0.253	16.566; 0.354	0.087; 0.194	0.122; 0.285	0.227; 0.483
4	4	28	15	9.461; 0.224	13.766; 0.284	22.699; 0.363	0.087; 0.204	0.137; 0.233	0.205; 0.269
4	4	40	5	11.993; 0.42	21.925; 0.578	32.639; 0.709	0.087; 0.211	0.122; 0.232	0.233; 0.246
4	4	40	10	16.287; 0.302	20.748; 0.43	33.72; 0.697	0.085; 0.212	0.119; 0.244	0.217; 0.293
4	4	40	15	19.739; 0.213	32.763; 0.385	53.707; 0.513	0.085; 0.177	0.119; 0.265	0.175; 0.313
6	3	21	5	2.306; 0.097	2.837; 0.123	3.108; 0.144	0.144; 0.184	0.197; 0.215	0.258; 0.248
6	3	21	10	1.742; 0.104	2.305; 0.135	2.763; 0.161	0.178; 0.191	0.207; 0.213	0.233; 0.23
6	3	21	15	2.172; 0.111	2.965; 0.148	4.203; 0.187	0.121; 0.165	0.191; 0.244	0.272; 0.286
6	3	30	5	2.929; 0.16	3.902; 0.173	5.183; 0.193	0.141; 0.188	0.192; 0.208	0.222; 0.24
6	3	30	10	3.007; 0.161	4.818; 0.219	6.163; 0.293	0.138; 0.182	0.17; 0.209	0.211; 0.24
6	3	30	15	3.356; 0.143	5.281; 0.168	9.553; 0.19	0.15; 0.224	0.203; 0.239	0.286; 0.278

Table 4 continued

n	k	m	σ	MILP time [s]			Algorithm 1 time [s]		
				Min	Avg	Max	Min	Avg	Max
6	4	28	5	11.582; 0.227	14.151; 0.336	18.425; 0.463	0.084; 0.231	0.097; 0.248	0.109; 0.266
6	4	28	10	11.384; 0.27	16.551; 0.392	22.42; 0.542	0.087; 0.17	0.1; 0.222	0.113; 0.263
6	4	28	15	14.808; 0.248	18.04; 0.369	22.502; 0.414	0.086; 0.2	0.122; 0.236	0.157; 0.303
6	4	40	5	22.484; 0.511	30.601; 0.605	41.785; 0.727	0.091; 0.204	0.1; 0.258	0.115; 0.337
6	4	40	10	23.296; 0.4	39.005; 0.574	56.527; 0.839	0.087; 0.208	0.098; 0.247	0.112; 0.273
6	4	40	15	30.735; 0.455	48.86; 0.747	83.98; 1.06	0.087; 0.221	0.133; 0.283	0.242; 0.364
10	3	21	5	3.753; 0.162	4.498; 0.19	6.524; 0.244	0.153; 0.227	0.2; 0.244	0.24; 0.258
10	3	21	10	3.986; 0.16	5.179; 0.186	7.565; 0.221	0.162; 0.201	0.195; 0.231	0.235; 0.265
10	3	21	15	4.457; 0.151	5.179; 0.184	5.906; 0.249	0.194; 0.178	0.213; 0.223	0.247; 0.243
10	3	30	5	9.122; 0.199	12.132; 0.264	18.447; 0.45	0.149; 0.211	0.209; 0.235	0.273; 0.296
10	3	30	10	7.142; 0.197	8.824; 0.285	10.743; 0.531	0.145; 0.17	0.176; 0.208	0.221; 0.228
10	3	30	15	6.951; 0.265	10.446; 0.326	12.144; 0.371	0.199; 0.176	0.212; 0.23	0.222; 0.312
10	4	28	5	21.514; 0.376	25.934; 0.744	29.121; 1.296	0.087; 0.21	0.107; 0.233	0.134; 0.259
10	4	28	10	27.323; 0.295	34.768; 0.597	49.631; 1.097	0.088; 0.2	0.099; 0.221	0.114; 0.238
10	4	28	15	22.078; 0.446	38.696; 0.764	58.176; 1.225	0.087; 0.188	0.099; 0.226	0.13; 0.312
10	4	40	5	48.124; 0.732	57.583; 1.114	75.404; 1.341	0.09; 0.161	0.101; 0.217	0.114; 0.247
10	4	40	10	76.34; 0.724	90.931; 1.161	129.885; 1.539	0.088; 0.209	0.101; 0.215	0.117; 0.229
10	4	40	15	61.362; 0.701	75.875; 1.327	92.652; 1.758	0.088; 0.192	0.101; 0.225	0.113; 0.261

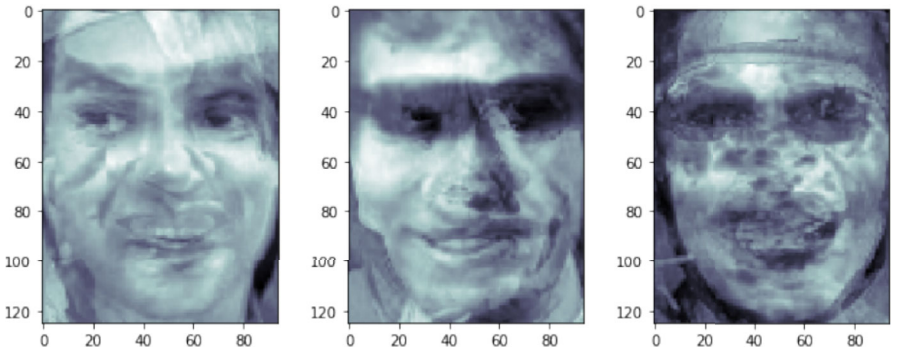


Fig. 5 Clustering result of the bisection algorithm for $k = 3$

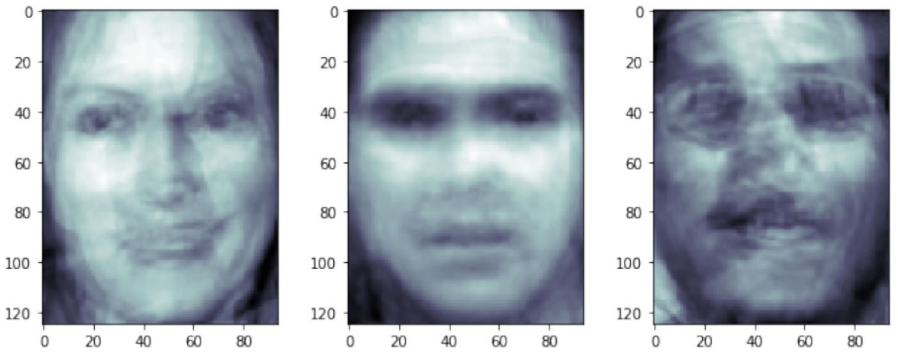


Fig. 6 Clustering result of the k-means algorithm for $k = 3$

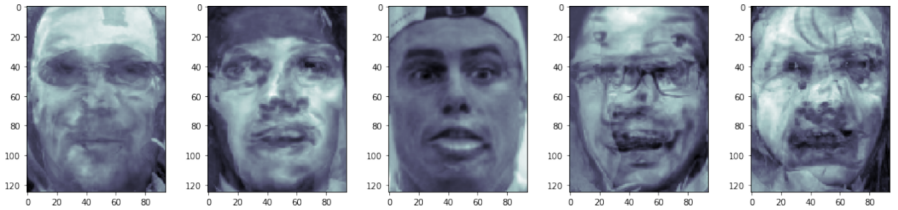


Fig. 7 Clustering result of the bisection algorithm for $k = 5$

configuration, we have randomly generated five times the corresponding data set and measured the minimum, average, and maximum execution times (in seconds) required by both approaches. The comparison reveals that Algorithm 1 consistently outperforms the (MILP) reformulation across different problem instances. Specifically, our method demonstrates significantly lower computational times, indicating its efficiency on the chosen set of problem instances when compared with the (MILP) modeling approach. Additionally, results generated with GUROBI solver (Gurobi Optimization 2024) are added.

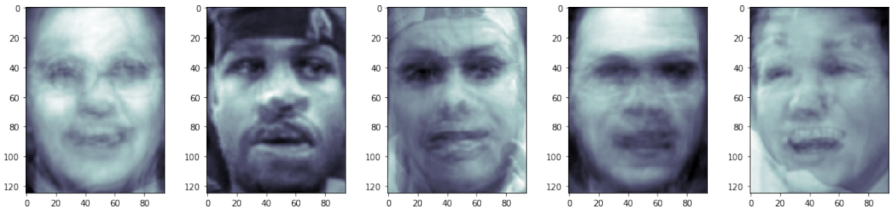


Fig. 8 Clustering result of the k-means algorithm for $k = 5$

Table 5 Results for data set "Labeled Faces in the Wild"

	$k = 3$ clusters	$k = 5$ clusters
Time for bisection method	4.0367 s	35.4832 s
Time for k-means algorithm	0.8070 s	0.7323 s

6.2 Clustering on real data

In this section, we apply our bisection method to real data. To this end, we use the data set "Labeled Faces in the Wild" (see Huang et al. 2000) that is directly available in the Python library scikit-learn.

We consider 50 arbitrarily selected grayscale images from the dataset. Each image is of the size 125×94, i.e., every image is described by a 11,750-dimensional vector. Both algorithms, k-means as well as our newly developed bisection algorithm, are applied. In the following, the results of this experiment are presented. We show the results of both algorithms for $k = 3$ and $k = 5$ clusters. The resulting cluster centers are depicted in Figs. 5, 7, 6, and 8. A notable difference is that we obtain much smoother images for the k-means algorithm and sharper ones for our new bisection method, which is expected to be a result of the different norms. Runtimes of this experiment are presented in Table 5.

Although our algorithm performs really well on all these artificial as well as real data instances (i.e., runtimes are below 1 min for all problems), it is important to note that we observe a significant increase in runtimes on instances with more data points. However, this is common for many $\mathcal{NP}$-hard problems that often occur in global optimization, where smaller instances can often be solved quickly, whereas larger ones are often impossible in a reasonable amount of time.

7 Final remarks

In this article, a new clustering algorithm for the global solution of a specific type of clustering problems is proposed. A proof of convergence is given and some numerical results illustrate the performance of the method. However, there are still some issues that need to be addressed.

First, our numerical results show that our new method performs well and, thus, seems to be a reasonable method to solve clustering problems that are defined based

on the maximum norm to global optimality. Our method is tested on real as well as on synthetic data sets to examine the performance of the method under different circumstances. A difficulty that is encountered during these tests is a drop in performance in the presence of overlapping of clusters. Although one might think that clusters should not overlap and that overlapping clusters might be a hint of assuming the existence of too many clusters within the set of data points, clearly, in practice such difficulties may occur and, thus, the need to cope with these issues arises.

Moreover, our bisection method relies on $\mathcal{NP}$-hard subproblems, namely the maximal clique and the k-cover set problem. So far, these are solved using a rather basic implementation and we expect that by replacing these algorithms with more sophisticated approaches, significant improvements in performance are possible. In particular, the integer linear problems are currently solved using the default solver of the Python package Pulp, which is known to be much weaker than commercial state-of-the-art solvers. Similarly, however, it should be mentioned that the same solver is also used in the alternative global solution approach based on an $MILP$ reformulation of the clustering problem. Thus, analogously, it could be argued that also the performance of this alternative method to solve the problem globally could be improved as well.

Finally, we would like to point out that our clustering algorithm in its current form is sensitive to outliers, since the greatest distance between a data point and its cluster center is responsible for the globally minimal value. In this article, we basically assume that a data set that is already adjusted appropriately. An approach to circumvent this issue is to enhance our method, so that it can cope with other norms, for instance, the Euclidean norm, which is less affected by this issue. However, without further considerations, this breaks Proposition 3.5, which is crucial for our approach. For this reason, major changes are required, and so, this is left for future research.

Acknowledgements The authors thank the anonymous referees for their precise and substantial remarks, which helped to significantly improve the paper. The authors gratefully acknowledge GUROBI OPTIMIZATION, LLC for providing a free academic license for their optimization software, which was used for the numerical computations in this study.

Data availability The datasets used in this study are of two types: (1) facial images "Labeled Faces in the Wild" for which the corresponding reference is provided in the present article, and (2) randomly generated point clouds as also specified in the present article. The source code for generating these datasets and reproducing the study's results is available at the GitHub Repository https://github.com/Monomachos/Clustering.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

References

Akkoyunlu EA (1973) The enumeration of maximal cliques of large graphs. SIAM J Comput 2(1):1–6

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

Babaki B, Guns T, Nijssen S (2014) Constrained clustering using column generation. In: International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research, 438–454

Bagirov AM (2008) Modified global k-means algorithm for minimum sum-of-squares clustering problems. Pattern Recogn 41(10):3192–3199

Bagirov AM, Rubinov AM (2024) Modified versions of the cutting angle method. In: Hadjisavvas and Pardalos PM (eds) Advances in convex analysis and global optimization. Kluwer Academic Publishers

Bagirov AM, Rubinov AM, Soukhoroukova NV, Yearwood J (2003) Unsupervised and supervised data classification via nonsmooth and global optimization. Top 11:1–75

Bagirov AM, Ugon J, Webb D (2011) Fast modified global k-means algorithm for incremental cluster construction. Pattern Recogn 44(4):866–876

Balas E, Padberg MW (1972) On the set-covering problem. Oper Res 20(6):1152–1161

Ben-Dor A, Shamir R, Yakhini Z (1999) Clustering gene expression patterns. J Comput Biol 6(3–4):281–297

Bondy JA, Murty U, Silva R (1976) Graph theory with applications. Macmillan, London

Brigham RC, Dutton RD (1983) On clique covers and independence numbers of graphs. Disc Math 44(2):139–144

Bron C, Kerbosch J (1973) Algorithm 457: finding all cliques of an undirected graph. Commun ACM 16(9):575–577

Brusco MJ (2003) An enhanced branch-and-bound algorithm for a partitioning problem. Br J Math Stat Psychol 56(1):83–92

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

Brusco MJ, Cradit D (2004) Graph coloring, minimum-diameter partitioning, and the analysis of confusion matrices. J Math Psychol 48(5):301–309

Diestel R (2000) Graph theory. Springer, New York

Dorndorf U, Pesch E (1994) Fast clustering algorithms. ORSA J Comput 6(2):141–153

Ester M, Kriegel H-P, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining, 96(34):226–231

Fränti P, Virmajoki O, Kaukoranta T (2002) Branch-and-bound technique for solving optimal clustering. Int Conf Pattern Recogn 2:232–235

Gary Augustson J, Minker J (1970) An analysis of some graph theoretical cluster techniques. J ACM 17:571–588

Gramm J, Guo J, Hüffner F, Niedermeier R (2005) Graph-modeled data clustering: fixed-parameter algorithms for clique generation. Theory Comput Syst 38(4):373–392

Gramm J, Guo J, Hüffner F, Niedermeier R (2009) Data reduction and exact algorithms for clique cover. J Exp Algorithmics 13(2)

Gurobi Optimization (2024) LLC. In: Gurobi optimizer reference manual. https://www.gurobi.com

Hand DJ (1981) Branch-and-bound in statistical data analysis. The Statistician 30(1):1–13

Hanif D (1999) Sherali and Warren P. Springer, Adams. A reformulation-linearization technique for solving discrete and continuous nonconvex problems

Hansen P, Delattre M (1978) Complete-link cluster analysis by graph coloring. J Am Stat Assoc 73(362):397–403

Hansen P, Jaumard B (1997) Cluster analysis and mathematical programming. Math Programm 79(1):191–215

Hartigan JA (1975) Clustering algorithms. Wiley, New York

Hartigan JA, Wong MA (1979) Algorithm AS 136: a K-means clustering algorithm. J R Stat Soc Ser C (Appl Stat) 28(1):100–108

Hartuv E, Shamir R (2000) A clustering algorithm based on graph connectivity. Inform Process Lett 76(4–6):175–181

Huang GB, Ramesh M, Berg T, Learned-Miller E (2000) Labeled Faces in the wild: a database for studying face recognition in unconstrained environments. University of Massachusetts, Amherst

Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surv 31(3):264–323
Jensen RE (1969) A dynamic programming algorithm for cluster analysis. Oper Res 17(6):1034–1057
Johnston HC (1976) Cliques of a graph: variations on the Bron–Kerbosch algorithm. Int J Comput Inform Sci 5:209–238
Johnson EL, Mehrotra A, Nemhauser GL (1993) Min-cut clustering. Math Programm 62(1):133–151
Jun W, Chu-Min Li L, Jiang JZ, Yin M (2020) Local search for diversified Top-k clique search problem. Comput Oper Res 116:104867
Karp RM (1972) Reducibility among combinatorial problems. In: Miller RE, Thatcher JW, Bohlinger JD (eds) Complexity of computer computations. The IBM Research Symposia Series. Springer, pp. 85–103
Klein G, Aronson JE (1991) Optimal clustering: a model and method. Naval Res Log 38(3):447–461
Kogan J, Nicholas C, Teboulle M (eds) (2006) Grouping multidimensional data: recent advances in clustering. Springer, Berlin
Koontz WLG, Narendra PM, Fukunaga K (1975) A branch and bound clustering algorithm. IEEE Trans Comput 100(9):908–915
Larrañaga P, Lozano JA, Peña JM (1999) An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recogn Lett 20(10):1027–1040
Liberti L, Manca B (2022) Side-constrained minimum sum-of-squares clustering: mathematical programming and random projections. J Global Optim 83(1):83–118
Likas A, Vlassis N, Verbeek JJ (2003) The global k-means clustering algorithm. Pattern Recogn 36(2):451–461
Loyd S (1982) Least squares quantization in PCM. IEEE Trans Inform Theory 28(2):129–137
MacQueen JB (1976) Some methods for classification and analysis of multivariate observations. Proc Fifth Berkel Symp Math Stat Probab 5(1):281–297
Merle OD, Hansen P, Jaumard B, Mladenović N (1999) An interior point algorithm for minimum sum-of-squares clustering. SIAM J Sci Comput 21(4):1485–1505
Michael J (2005) Brusco and stephanie stahl. In: Branch-and-Bound Applications in Combinatorial Data Analysis. Springer, New York
Pardalos PM, Rodgers GP (1992) A branch and bound algorithm for the maximum clique problem. Comput Oper Res 19(5):363–375
Peng J, Xia Y (2005) A cutting plane algorithm for the minimum sum-of-squared error clustering. In: Proceedings of the 2005 SIAM International Conference on Data Mining, pp. 150-160
Piccialli V, Sudoso AM (2023) Global optimization for cardinality-constrained minimum sum-of-squares clustering via semidefinite programming. Math Program
Piccialli V, Russo AR, Sudoso AM (2022a) An exact algorithm for semi-supervised minimum sum-of-squares clustering. Comput Oper Res 147(C)
Piccialli V, Sudoso AM, Wiegele A (2022) SOS-SDP: an exact solver for minimum sum-of-squares clustering. INFORMS J Comput 34(4):2144–2162
Rao MR (1971) Cluster analysis and mathematical programming. J Am Stat Assoc 66(335):622–626
Sherali HD, Desai J (2005) A global optimization RLT-based approach for solving the hard clustering problem. J Global Optim 32(2):281–306
Steinley D (2003) Local optima in K-means clustering: what you don't know may hurt you. Psychol Methods 8(3):294–304
van Os BJ, Meulman JJ (2004) Improving dynamic programming strategies for partitioning. J Classif 21(2):207–230
Ward JH (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58(301):236–244
West Douglas B (2001) Introduction to graph theory. Prentice Hall, Upper Saddle River
Zhong C, Miao D, Wang R (2010) A graph-theoretical clustering method based on two rounds of minimum spanning trees. Pattern Recogn 43(3):752–766