Global Solution of the Clustering Problem via Graph Theoretical Approach

Tomáš Bajbar * Peter Kirst # Mario Merkel ‡

January 18, 2020

Abstract

In this article we consider clustering problems which we model as a non-convex continuous minimization problem with the maximum norm representing the distance measure. We then 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 k-means algorithm and also discuss the limits and challenges of the proposed procedure.

Keywords: Clustering problem, global optimization, maximal clique, unsupervised learning.

AMS subject classifications: 90C26, 62H30, 68T10, 91C20

1 Introduction

Clustering is a wide-spread technique to get an overview over the structure of a given data set. In particular, clustering is a well established unsupervised learning technique which aims to allocate a given set of data points to specific subsets called *clusters* by means of minimizing some dissimilarity measure.

*tomas.bajbar@gmail.com
#peter-kirst@web.de
‡merkel-mario@gmx.de

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. 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 measure the distances in terms of the maximum norm. The goal of this article is to propose a new method for the solution of these problems. In contrast to most approaches from the literature we strive for a globally optimal point. We achieve this by means of a bisection method where in every iteration some classical graph-theoretical subproblems are solved.

Cluster analysis is an important approach that is helpful within various applications. It is thus not surprising that a large amount of literature is devoted to this topic. We refer to [12, 16] for an introduction. A classical distance-based approach for solving the clustering problem locally is the k-means algorithm. Its various versions, usually considering the Euclidean norm for measuring distances, are described in [3, 4, 17, 18, 20]. There exist also other distance-based approaches that aim at solving the problem globally such as the global *Reformulation-Linearization-Technique*-based method (RLT) as proposed in [21, 22].

We do not consider so-called hierarchical clustering, which, in contrast to our approach, does not assume prior knowledge of the number of clusters to be identified but provides this number as a part of the solution instead. Moreover, there exist several heuristic approaches to solve different tasks in cluster analysis, which we do not consider throughout this article.

Our approach relies on the application of techniques 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 [6, 9, 23]. 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 [7, 8, 15]. For a description of algorithms to compute cliques of a graph in general we refer to [14]. As we are especially interested in so-called maximal cliques, we also mention the well-known Bron-Kerbosch algorithm introduced in [8] and, similarly, the article [1]. Although being known to be $\mathcal{NP}$-hard, the clique cover problem can be accelerated e.g. by data reduction techniques as described in [11].

The idea to apply graph theoretical approaches to solve the clustering problem is already known in the literature. In general, in these methods all given data points are considered as vertices of a graph and the corresponding clusters are identified as certain subgraphs of this graph. As suitable

subgraphs, often *cliques* of a graph, also called complete subgraphs, are considered to identify clusters. To our best knowledge the term "clique" is first introduced in [19]. For instance, in [5] the relationship between the clique and the clustering problem is exploited to solve problems regarding gene expression patterns. Early and detailed introductions to graph theoretical cluster techniques describing different algorithms for computing cliques can be found in [2] and the references therein. Considering subgraphs other than cliques, different shapes of clusters can be computed as done, e.g. in [24]. In [13], for instance, instead of cliques, so-called highly connected subgraphs are interpreted as clusters. Related to this is also the approach described in [10]. In the existing literature, when approaching the clustering problem by applying graph-theoretical instances, in general, some similarity measure is considered to compute the weights of the edges in a given graph. However, in contrast to our approach, often no relationship to a certain norm such as the maximum norm is drawn.

The article is structured as follows. In Section 2 we introduce some important notions and define the global clustering optimization problem as well as some graph theoretical formulations. In Section 3 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 Sections 4 and 5 we discuss solution strategies for these subproblems together with their complexity and we propose acceleration steps which further improve the overall computational performance of our algorithm. Finally we compare the proposed method with a randomly initiated k-means method and we also discuss results of the corresponding numerical experiments. The article closes with some final remarks in Section 6.

2 Clustering problem and its graph-theoretical aspects

Given a finite set of points

$$X = \{x^i \in \mathbb{R}^n \mid i = 1, \dots, m\}$$

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 $C_1, \dots, C_k \subseteq X$ satisfying $\bigcup_{j=1}^{k} C_j = X$ so that data points within one cluster C_j are close to each other. We do not assume here that the clusters $C_1, \dots, C_k$ are disjoint.

In order to obtain a reasonable criterion for measuring the similarity of the data points within each cluster we consider k cluster centers $z^1, \dots, z^k \in \mathbb{R}^n$ which represent their position in $\mathbb{R}^n$. For obtaining a clustering result of the

best quality we consider the distances between each of the cluster centers and the corresponding assigned data points by means of an arbitrary norm. These distances serve as a basis for evaluating the goodness of our clustering result and our aim is to minimize them. More precisely, we try to position k cluster centers $z^1, \ldots, z^k$ in such a way 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

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

is fulfilled with $\|\cdot\|$ denoting an arbitrary norm. We model the assignment of an arbitrary data point $x^i \in X$ to its closest distanced cluster center by considering an assignment map $\sigma : \{1, \ldots, m\} \to \{1, \ldots, k\}$ defined by

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

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

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

we want to minimize and which 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 $z^1, \ldots, z^k \in \mathbb{R}^n$ in such a way that the objective function of the non-convex problem P is minimized. Moreover, we strive for a globally minimal point of P in contrast to many approaches from the literature.

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

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

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 $\bigcup_{j=1}^k C_j$ is satisfied as well. Thus, identifying a globally minimal

point of the problem P yields also a clustering of the data set X we are looking for.

Depending on the choice of the norm which we use for measuring distances we obtain different objective functions for the problem P and hence also different clustering results in general. The most prominent method called k-means (see, e.g. [3, 4, 18, 20]) assumes the Euclidean norm. Here we restrict our approach and consider the maximum norm 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 $\|d\|_\infty = \max_{i=1,\dots,n} |d_i|$ for an arbitrary $d \in \mathbb{R}^n$. This leads to the optimization problem

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

which is of our central interest in this article and which can also be written in the form

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

We continue our reformulation by shifting the objective function into the constraints and consider the problem

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

Rewriting the outer norm we obtain

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

and since we can further replace the latter inequality constraint by m inequalities by means of the following equivalence

$$\begin{aligned} & \max_{i=1,\dots,m} \ \min_{j=1,\dots,k} \ \|z^j - x^i\|_\infty && \leq \alpha \\ \iff \quad & \min_{j=1,\dots,k} \ \|z^j - x^i\|_\infty && \leq \alpha \qquad \forall i = 1,\dots,m \\ \iff \quad & \|z^{\sigma(i)} - x^i\|_\infty && \leq \alpha \qquad \forall i = 1,\dots,m, \end{aligned}$$

we finally obtain

$$\widetilde{P}: \quad \min_{z^1,\ldots,z^k\in\mathbb{R}^n,\ \alpha\in\mathbb{R}} \quad \alpha$$

$$\text{s.t.} \qquad \|z^{\sigma(i)} - x^i\| \leq \alpha \qquad \forall i = 1,\ldots,m.$$

It can be shown that 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 such a case, furthermore, the value $\alpha^\star \in \mathbb{R}$ is the globally minimal value of both P^∞ and $\widetilde{P}$.

Considering the optimization problem $\widetilde{P}$ 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, for solving the problem $\widetilde{P}$ globally, we try to find k cluster centers $z^1,\ldots,z^k$ in a way 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.

Example 2.1. *Let $X \subseteq \mathbb{R}^2$ be a set of 10 following points:*

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

In Figure 1 which depicts the data set X we can intuitively recognize three separated subsets in X.

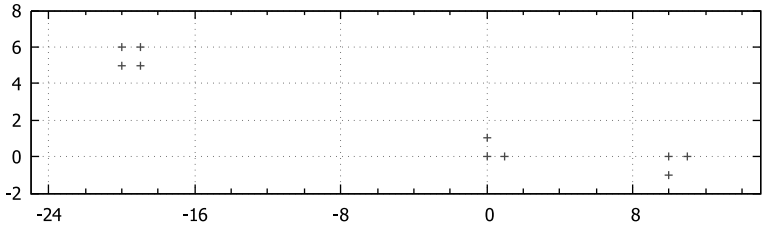


Figure 1: Representation of the data points in X

Therefore, we propose a partition of X into the following three clusters:

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

Considering the problem $\widetilde{P}$ for our data set X by setting $k = 3$, Figure 2 reveals that it is possible to cover the whole dataset X by $k = 3$ boxes all of

a side length 1 (or equivalently by $k = 3$ maximum norm balls all of a radius 0.5) and it is clear that we cannot cover the whole dataset X by $k = 3$ boxes of a side length less than 1 (or equivalently by $k = 3$ maximum norm balls of radius less than 0.5).

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 correspond to the centers of the grey-shaded boxes depicted in Figure 2.

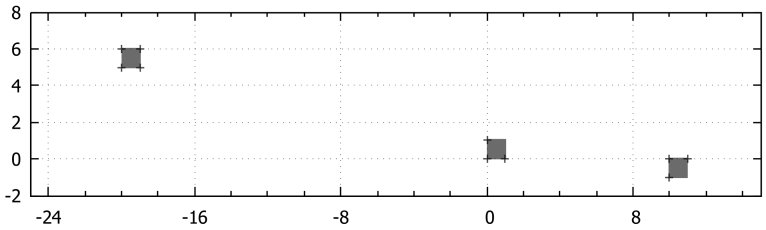


Figure 2: Three boxes representing the global solution of the clustering problem $\widetilde{P}$

Next we recall some concepts from graph theory in order to be able to propose our new approach which solves the clustering problem to global optimality by means of graph-theoretical instances.

Definition 2.2. *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 2.2 any graph $G(\beta) = (X, \beta)$ 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 $G(\beta)$. Figure 3 illustrates the graph $G(2) = (X, E_2)$ for the data set $X \subseteq \mathbb{R}^2$ from Example 2.1.

A simple comparison of Figures 2 and 3 reveals that the three clusters in the data set X identified as a global solution to the clustering problem $\widetilde{P}$ from Example 2.1 correspond to three such subsets of nodes in the graph $G(2) = (X, E_2)$ whose 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.

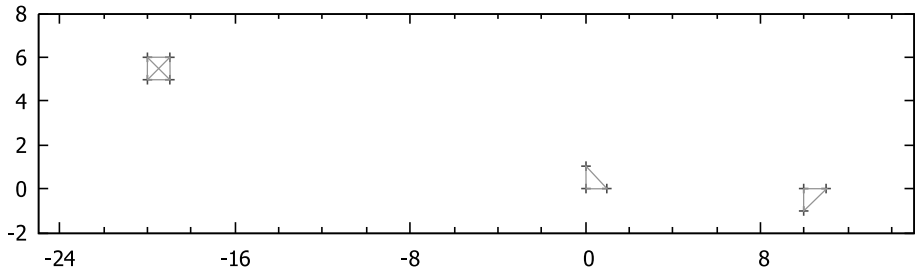


Figure 3: Graph $G(2) = (X, E_2)$ based on the dataset X from Example 2.1

Definition 2.3 (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 our problem of interest, namely the clustering problem P^∞.

Proposition 2.4. *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. We denote the optimal solution of $\widetilde{P}$ by $(z^{1\star}, \ldots, z^{k\star}, v^\star)^T$. Furthermore, 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 $\|x - z^{i\star}\|_\infty + \|y - z^{i\star}\|_\infty \leq 2v^\star$ holds. Applying the triangle inequality 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 can set $S_i = C_i$ 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 such k cliques in the graph $G(\beta) = (X, E_\beta)$. Before proving this in Proposition 2.6 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 2.5. *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

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

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 2.6. *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*

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

where 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 2.5, 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 $i = 1, \ldots, k$. Now we 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

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

In summary 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 $\frac{\beta}{2} < v^\star$ which contradicts $v^\star$ being a globally minimal value of $\widetilde{P}$. □

Using this we can propose an algorithm which approximates the globally minimal value $v^\star$ of the optimization problem $\widetilde{P}$ by means of a bisection method.

3 A global bisection method for solving the clustering problem

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 2.4 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 we apply Proposition 2.6 and 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 $\widetilde{P}$.

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 perform the following bisection procedure as we propose in Algorithm 1. Appropriate initial values for lower and upper bounds are proposed below in Proposition 3.2 and Proposition 3.3, respectively.

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

Data: $X = \{x^1, \ldots, x^m\} \subseteq \mathbb{R}^n$.
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 $2v^\star \in I$ together with the corresponding feasible point $(z^{1\star}, \ldots, z^{k\star}, \frac{\overline{v}^{\zeta^\star}}{2})^T$ of $\widetilde{P}$.

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 $S_1, \ldots, S_k$ in $G(v^\zeta)$ with $\bigcup_{i=1}^k S_i = X$;
6 **if** *(X can be partitioned into such k cliques $S_1, \ldots, S_k$)* **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 $S_1, \ldots, S_k$ in the graph $G(\overline{v}^\zeta)$;

At 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 $2v^\star$ we can choose a new value v^ζ by setting

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

for some value $\lambda \in (0, 1)$. For instance, 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 Section 4.

By means of Proposition 2.4 and Proposition 2.6 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 both bounds converge towards each other. We summarize this result in the next theorem.

Theorem 3.1. *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 Proposition 2.4 and Proposition 2.6 as well as from the theory of bisection methods. □

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.

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 solution 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 2.4. Thus we start with k cliques $S_1, \dots, S_k$ in $G(\overline{v}^{\zeta^\star}) = (X, E_{\overline{v}^{\zeta^\star}})$ fulfilling

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

For all $i = 1, \dots, k$ 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 2.5, by setting

$$\begin{array}{ll} l_j^i & = \min\{x_j | x \in S_i\}, \qquad j = 1, \dots, n, \\ u_j^i & = \max\{x_j | x \in S_i\}, \qquad j = 1, \dots, n, \end{array} \tag{3.1}$$

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

$$z^{i\star} := \begin{pmatrix} \frac{l_1^i + u_1^i}{2} \\ \vdots \\ \frac{l_n^i + u_n^i}{2} \end{pmatrix} \quad \text{for } i = 1, \dots, k \tag{3.2}$$

and estimate the distance $\|x - z^{i^\star}\|_\infty$ as already done in the proof of Proposition 2.6 as follows. 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

$$(z^{1^\star}, \ldots, z^{k^\star}, \frac{\overline{v}^{\zeta^\star}}{2})^T. \tag{3.3}$$

In the following we propose a possibility to initialize the upper bound $\overline{v}^0$ and the lower bound $\underline{v}^0$ in Algorithm 1. Note that in the following result the slightly unusual yet natural assumption $k < m$ is imposed.

Proposition 3.2. *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. 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 thus m cliques are required to cover the dataset X. However, this contradicts the assumption $k < m$. Thus, by means of Proposition 2.4

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

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

Proposition 3.3. *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. Suppose it holds $\beta \geq \overline{v}^0$ for some $\beta \in \mathbb{R}$. Then for two arbitrary data points $x, y \in X$ it holds

$$\|x - y\|_\infty \leq \overline{v}^0 \leq \beta$$

and thus the graph $G(\beta) = (X, E_\beta)$ is complete, i.e. $[x, y] \in E_\beta$ for all $x, y \in X$ with $x \neq y$ and hence the set X itself is a clique in $G(\beta)$. Then, we can find a partition of X into k cliques $S_1, \ldots, S_k$ immediately, for instance by letting $S_1 := X$ and $S_i := \emptyset$ for $i = 2, \ldots, k$. In summary

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

is a valid upper bound at the value $2v^\star$ in accordance to Proposition 2.4. □

Some iterations of Algorithm 1 are illustrated in Example A.1 in the appendix.

4 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 $G(v^\zeta)$. This leads to the so-called clique as well as the so-called k-cover set problems that typically arise in the field and will be discussed in more detail in the following two subsections. For a more detailed description of graph theoretical problems we refer to the literature [6, 9, 23] and the references therein.

4.1 Computing appropriate cliques

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

Definition 4.1 (see [1])**.** *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.

With the following two lemmas we show 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 4.2. *Let a set of data points X and a set of cliques $S_1, \dots, S_k$ 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 $S_1^\star, \dots, S_k^\star$ with $S_i \subseteq S_i^\star$ for all $i = 1, \dots, k$ 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 4.3. *Let a set of data points X be given such that there is no partition of the set X into k sets $S_1, \dots, S_k$ with $\bigcup_{i=1}^{k} S_i = X$ and all S_i, $i = 1, \dots, k$ are cliques in $G = (X, E)$. Then, we cannot find any k maximal cliques $S_1^\star, \dots, S_k^\star$ 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 highlight the fact that the maximal cliques $S_1^\star, \dots, S_k^\star$ do not need to be mutually distinct, i.e. $S_i^\star = S_j^\star$ for $i \neq j$ may occur. For instance, let us assume we are able to partition the set of data points X by already $\tilde{k} < k$ maximal cliques in a graph $G = (X, E)$. We denote these maximal cliques by $S_1^\star, \dots, S_{\tilde{k}}^\star$ and so we have

$$\bigcup_{i=1}^{\tilde{k}} S_i^\star = X. \tag{4.1}$$

Introducing additional cliques $S_{\tilde{k}+1}^\star, \dots, S_k^\star$ by e.g. setting

$$S_i^\star := S_1^\star \quad \text{for} \quad i = \tilde{k} + 1, \dots, k \tag{4.2}$$

we obtain k maximal cliques $S_1^\star, \dots, S_k^\star$ with

$$\bigcup_{i=1}^{k} S_i^\star = X. \tag{4.3}$$

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 the 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 method as described in Algorithm 2. Note that methods of this type are standard in the literature of graph theory and Algorithm 2 is presented here for the sake of completeness. Alternative approaches are also possible. For convenience, in the following we denote the set of all maximal cliques in a graph $G(\beta) = (\bar{X}, E_\beta)$ computed by Algorithm 2 by $\Phi_{\bar{X}}(\beta)$ where $\bar{X}$ denotes some nonempty subset of X.

Algorithm 2: Algorithm for determining all maximal cliques

Data: A nonempty subset $\bar{X} \subseteq X$.
Input: current guess $\beta > 0$.
Result: Set of maximal cliques $\mathcal{MC} = \{S_1, S_2, \ldots\}$ in the graph $G(\beta) = (\bar{X}, E_\beta)$.

```
isClique = true;
Generate graph G(β) = (X̄, E_β);
foreach x ∈ X̄ do
    foreach y ∈ X̄ do
        if (x ≠ y) then
            if ([x, y] ∉ E_β) then
                isClique = false;
                X̄_1 = X̄\{x};
                X̄_2 = X̄\{y};
                Compute Φ_{X̄_1}(β) using Algorithm 2 recursively;
                Compute Φ_{X̄_2}(β) using Algorithm 2 recursively;
            end
        end
    end
end
if (isClique) then
    X̄ is added to the set of maximal cliques MC using Algorithm 3
      (which adds only such cliques to MC which seem to be
      maximal);
end
```

For Algorithm 2 we consider again a set of given data points $X = \{x^1, \ldots, x^m\}$ and a current approximation β of the value $2v^\star$ with $v^\star$ denoting the globally

minimal value of the optimization problem $\widetilde{P}$.

The subset of data points under consideration is denoted by $\bar{X}$. We are interested in whether the set $\bar{X}$ already represents a maximal clique in the graph $G(\beta)$ or not. Therefore, we apply Algorithm 2 with input data $\bar{X}$ and graph $G(\beta)$ in order to check, if $\bar{X}$ is already a maximal clique.

In case $\bar{X}$ does not represent a clique, then there are at least two data points $x, y \in \bar{X}$ which are not connected in the graph $G(\beta)$, i.e. $[x, y] \notin E_\beta$. Thus we consider each pair of nodes $x, y \in \bar{X}$ with $x \neq y$ and check if there is a connection between them, i.e. $[x, y] \in E_\beta$. In case $[x, y] \notin E_\beta$ it is impossible that the nodes x and y are in the same clique. Thus $\bar{X}$ is not a clique in the graph $G(\beta)$ and we create two subsets $\bar{X}_1$ and $\bar{X}_2$ as described at the lines 8 and 9 in Algorithm 2 and we consider two nested loops at lines 10 and 11 of Algorithm 2 where the procedure is called recursively for the sets $\bar{X}_1$ and $\bar{X}_2$ as input data.

If two nodes $x, y \in \bar{X}$ are not connected to each other, then the first set of data points $\bar{X}_1$ contains all data points in $\bar{X}$ except the data point x and the second set of data points $\bar{X}_2$ contains all data points in $\bar{X}$ except the data point y. After creation of these sets the procedure is started again, recursively.

The recursion in Algorithm 2 stops if all nodes from $\bar{X}$ are connected to each other in the graph $G(\beta) = (X, E_\beta)$, i.e. $[x, y] \in E_\beta$ for all $x, y \in \bar{X}$, $x \neq y$.

So far, different kinds of cliques in a graph $G(\beta) = (X, E_\beta)$ may be determined, not necessarily solely maximal cliques. In order to ensure that eventually all identified cliques are maximal according to Definition 4.1, an additional method to add cliques to the set $\mathcal{MC}$ is called at line 17 in Algorithm 2 which is stated in Algorithm 3, formally. For any obtained maximal clique candidate $\bar{X}$ to be a maximal clique the following condition must be satisfied

$$\bar{X} \not\subset S, \quad \forall S \in \mathcal{MC}.$$

In such a case $\bar{X}$ is added to the set $\mathcal{MC}$ at line 2 in Algorithm 3 and we remove all maximal clique candidates $S \in \mathcal{MC}$ with $S \subsetneq \bar{X}$ from the set $\mathcal{MC}$ because they cannot be expected to be maximal anymore. This approach yields a pruned set of maximal clique candidates.

Algorithm 3: Algorithm excluding cliques which are not maximal

Data: Set of initial maximal clique candidates $\mathcal{MC}$.
Input: New maximal clique candidate $\bar{X}$.
Result: Updated set of maximal clique candidates $\mathcal{MC}$.

```
1 if X̄ is not a subset or equal to any set in MC then
2 |   X̄ is added to the set MC;
3 |   foreach S ∈ MC do
4 |   |   if S ⊊ X̄ then
5 |   |   |   Remove S from MC;
6 |   |   end
7 |   end
8 end
```

In the final part of the present subsection we show that Algorithm 2 in fact identifies only cliques. Furthermore, we further show that it identifies all maximal cliques in $G(\beta) = (X, E_\beta)$. This is stated formally in the next results.

Lemma 4.4. *For given inputs $\emptyset \neq \bar{X}$ and $\beta > 0$, as a result of Algorithm 2 only cliques in the graph $G(\beta) = G(\bar{X}, E_\beta)$ are identified.*

Proof. We assume that the assertion does not hold and derive a contradiction. If the result $\mathcal{MC}$ of Algorithm 2 for a given input $\emptyset \neq \bar{X} \subseteq X$ and $\beta > 0$ contains a subset $S \in \mathcal{MC}$ which is not a clique in the graph $G(\beta)$, then there are nodes $x, y \in S \subseteq \bar{X}$ with $x \neq y$ and $[x, y] \notin E_\beta$. According to lines 6 and 7 in Algorithm 2 we then have

$$isClique = false$$

and as a result of line 16 in Algorithm 2 the set S is not added to the set $\mathcal{MC}$ which contradicts our assumption. □

Lemma 4.5. *For given inputs $\emptyset \neq \bar{X}$ and $\beta > 0$, as a result of Algorithm 2 all maximal cliques in the graph $G(\beta) = G(\bar{X}, E_\beta)$ are identified.*

Proof. If $G(\beta)$ is a complete graph, then there is only one maximal clique $S = \bar{X}$ and this is identified immediately in the first iteration of the Algorithm 2.

In the following we assume that there is a maximal clique $\tilde{S}$ in the graph $G(\beta)$ with $\tilde{S} \neq \bar{X}$ which is not discovered by Algorithm 2 and we derive a contradiction. We have

$$\bar{X} = \tilde{S} \cup (\bar{X} \backslash \tilde{S})$$

and thus in order to identify $\tilde{S}$ as a maximal clique, Algorithm 2 has first to remove all nodes in $\bar{X} \backslash \tilde{S} \neq \emptyset$ from our initial set $\bar{X}$ to obtain $\tilde{S}$. In

particular, $\tilde{S}$ is a maximal clique and, thus, $\tilde{S} \cup \{x\}$ cannot be a maximal clique in the graph $G(\beta) = (\bar{X}, E_\beta)$ for all $x \in \bar{X} \backslash \tilde{S}$. More precisely, it holds

$$\forall x \in \bar{X} \backslash \tilde{S}: \quad \exists y \in \tilde{S} : [x, y] \notin E_\beta.$$

Since $\bar{X} \backslash \tilde{S} \neq \emptyset$ and $\tilde{S} \neq \emptyset$ there are two points

$$x \in \bar{X} \backslash \tilde{S}, \ y \in \tilde{S}$$

with $[x, y] \notin E_\beta$ for which Algorithm 2 at lines 7 to 11 computes $\Phi_{\bar{X} \backslash \{x\}}(\beta)$ and $\Phi_{\bar{X} \backslash \{y\}}(\beta)$. Here only $\Phi_{\bar{X} \backslash \{x\}}(\beta)$ is relevant for discovering the maximal clique $\tilde{S}$ which is identified by finitely many further recursive calls of the method. □

Hence, Algorithm 2 is a reasonable method to compute all maximal cliques in a given graph G. However, the maximal clique problem is $\mathcal{NP}$-hard as already shown in [15] and thus cannot be expected to be solved efficiently. Nevertheless, Algorithm 2 can be accelerated by exploiting knowledge of previous iterations as we shall propose in Section 4.3.

4.2 Solving the k-cover set problem

In order to successfully implement lines 5 and 6 of Algorithm 1, it is still necessary to have a reasonable 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 $S_1, \ldots, S_k \in \mathcal{S}$ which 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 and its complexity aspects have already been examined in [15], for instance. Due to

$$\bigcup_{S \in \mathcal{S}} S = X,$$

we have immediately that 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 so a 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 consider the power set $\mathcal{P}(\mathcal{S})$ of $\mathcal{S}$ and our goal is to identify a set of maximal cliques $\Sigma := \{S_1, \ldots, S_k\} \in \mathcal{P}(\mathcal{S})$ fulfilling

$$\bigcup_{j=1}^{k} S_j = X.$$

This leads to Algorithm 4 which is basically a complete enumeration of all possibilities.

Algorithm 4: Algorithm for solving the k-cover set problem

Data: $X = \{x^1, \dots, x^m\}$.
Input: Number $k \in \mathbb{N}$, set of maximal cliques $\mathcal{S}$ with $|\mathcal{S}| > k$
Result: Set of k maximal cliques $\Sigma^\star := \{S_1^\star, \dots, S_k^\star\} \in \mathcal{P}(\mathcal{S})$ with $\bigcup_{i=1}^k S_i^\star = X$ or the empty set $\emptyset$ in case this is not possible.

```
1 Σ* := ∅;
2 foreach Σ = {S_1, S_2, ...} ∈ P(S) do
3     if (|Σ| = k) then
4         if (⋃_{j=1}^k Σ_j = X) then
5             Σ* := Σ;
6             Stop;
7         end
8     end
9 end
```

In Algorithm 4 we consider each set $\Sigma \in \mathcal{P}(\mathcal{S})$ with $|\Sigma| = k$ according to the lines 2, 3 and stop as soon as a set $\Sigma \in \mathcal{P}(\mathcal{S})$ with the desired covering property is identified. However, if we cannot find any k maximal cliques that cover the whole dataset X, then the algorithm has to consider all sets $\Sigma \in \mathcal{P}(\mathcal{S})$ with $|\Sigma| = k$ which may be very time-consuming for large data sets X. Hence, the consideration of a graph $G(v^\zeta) = (X, E_{v^\zeta})$ with $v^\zeta < 2v^\star$ may lead to potentially undesirable computational effort in that case in order to apply Proposition 2.6. The k-cover set problem is also known to be $\mathcal{NP}$-hard (see [15]).

Hence, solving the clustering problem globally using our bisection Algorithm 1 cannot be seen as an easy task since we have to determine all maximal cliques in the graph $G(v^\zeta)$ in each iteration $\zeta \in \mathbb{N}$ which is itself an $\mathcal{NP}$-hard problem and, additionally, in a second step we have to decide whether a covering of X by k maximal cliques $S_1, \dots, S_k$ in the graph $G(v^\zeta)$ is possible, which is also an $\mathcal{NP}$-hard problem.

Therefore we cannot expect to apply polynomial time algorithms within our framework and thus it is very important to implement suitable strategies for runtime acceleration which we discuss in more detail in the subsequent Subsection 4.3.

4.3 Acceleration steps

In the following we propose acceleration steps for the improvement of the runtime of the subproblems arising in Algorithm 1, especially that of Algorithm 2 which determines all maximal cliques in each iteration of our bisection method. Although the aforementioned methods to solve the sub-

problems 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 Section 5.

4.3.1 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 2.2 and Definition 4.1, 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 always a clique in the graph $G(\tilde{\beta})$ for $\beta \leq \tilde{\beta}$ although not necessarily a maximal one. Hence, Algorithm 1 can 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.

Each time we update the upper bound $\overline{v}^\zeta$ at line 11 in Algorithm 1 we can save the maximal cliques which are already determined at line 6. Instead of calculating the set of maximal cliques $\Phi_X(v^\zeta)$ from scratch we can 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 $v^\zeta \leq \overline{v}^\zeta$.

4.3.2 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 set $S_i,\ i = 1, \ldots, k$. Then another point $y \in X$ may be contained in the same clique as the point x only if we have

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

Therefore, we can construct supersets of maximal cliques from the perspective of each individual data point $x \in X$ as given in the following definition.

Definition 4.6. *Given some data set $X = \{x^1, \ldots, x^m\}$ and a graph $G(\beta) = (X, E_\beta)$ for some value $\beta \geq 0$. 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 4.6 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, considering Algorithm 1, instead of calculating the set of all maximal cliques in $G(\beta) = (X, E_\beta)$, it suffices to calculate the set of maximal cliques $\Phi_{G_i}(\beta)$ for each $i = 1, \dots m$ which can significantly reduce runtime because each of the sets G_i may only contain a small fraction of all given data points X.

Nevertheless, each 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. The sets G_i may be used to initialize Algorithm 2 instead of always starting with the whole set of data points X.

4.3.3 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 2.4 the lower bound at the value $2v^\star$ can be increased to β immediately.

This provides a possibility for Algorithm 1 to recognize lower bounds at the optimal value $v^\star$ of the optimization problem $\widetilde{P}$ in many cases very fast, since in that case the solution of the maximal clique as well as the k-cover set subproblem becomes superfluous.

4.3.4 Improving upper bounds

In the following we provide another possibility to accelerate the runtime of Algorithm 1 by improving the upper bounds at the value $2v^\star$. 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 propose the update

$$\overline{v}^{\zeta+1} := \max_{i=1,\ldots,k} \max_{x,y \in S_i} \|x - y\|_\infty. \tag{4.4}$$

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

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

Since $\bigcup_{i=1}^k S_i = X$ is fulfilled, according to Proposition 2.4 the value $\overline{v}^{\zeta+1}$ is a valid upper bound at the value $2v^\star$. Furthermore, since the inequality $\overline{v}^{\zeta+1} \leq v^\zeta$ holds, the upper bound update from equation (4.4) is better than the original one as proposed in line 7 of Algorithm 1 and might improve the overall performance of our bisection method.

5 Numerical results

In this section we discuss the runtime of our bisection method by means of complexity considerations. Moreover, we present the results of our numerical experiments.

5.1 Number of iterations and complexity of subproblems of Algorithm 1

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 is then given explicitly by the following result. The proof follows immediately using standard arguments of bisection methods. We present these considerations here for the sake of completeness.

Proposition 5.1. *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 2.4 and Proposition 2.6, in each iteration $\zeta \in \mathbb{N}$ of Algorithm 1 the gap $\overline{v}^\zeta - \underline{v}^\zeta$ is divided in half, i.e.

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

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 $\overline{v}^0 - \underline{v}^0$. However, if we consider the improved upper bounds as given in Section 4.3.4 there are rather less iterations in many cases.

Observe that neither the number of data points $m \in \mathbb{N}$, 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 $\mathbb{R}^n$ have a direct impact on the number of iterations since the greatest and the smallest distance between two arbitrary data points in X affect the initialization of the bounds $\overline{v}^0$ and $\underline{v}^0$ as proposed in Propositions 3.2 and 3.3.

Next we consider one single iteration $\zeta \in \mathbb{N}$ of Algorithm 1 and we discuss the complexity of the subproblems which have to be solved. More precisely, we have to determine the maximal cliques in the graph $G(v^{\zeta})$ and in case there are more than k maximal cliques in the graph $G(v^{\zeta})$, 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 necessary for computing 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 we thus have $\mathcal{O}(nm^2)$. However, this procedure only needs to be performed once which may 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 Algorithm 2. The recursive tree calls

increase with an increasing number of data points m. The maximal clique problem is known to be $\mathcal{NP}$-hard (see [15]), and thus, in the worst case the runtime necessary for determining all maximal cliques in a given graph $G(v^\zeta)$ increases exponentially in the number of data points m.

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. In such a case we know that there are $\binom{|\mathcal{S}|}{k}$ sets of k maximal cliques in $G(v^\zeta)$ and in the worst case we have to consider all these sets while performing Algorithm 4.

However, these are only worst-case considerations and, moreover, our numerical experiments show that upon implementation of the acceleration strategies presented in Section 4.3 we are able to solve the clustering problem with $k = 3$ or $k = 5$ clusters to global optimality even for larger instances without taking too much effort in most cases (see Figure 4 in Section 5.2).

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, or data sets containing outliers.

5.2 Implementation

In this subsection our implementation of our clustering approach is described. The algorithm is developed in Java. A machine with an Intel(R) Core(TM)2 CPU 4300 @ 1,8 GHz processor and 3 GB RAM running Windows is used for the computation.

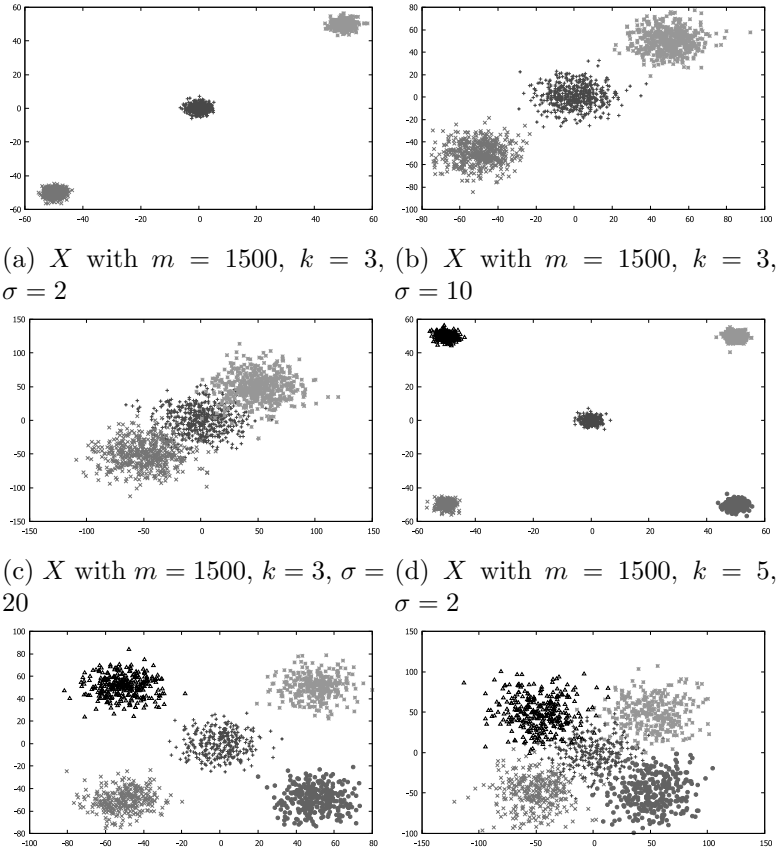
In order to examine the runtime of Algorithm 1 with respect to number of data points m and dimension n, we test our bisection method for various data instances with $k = 3$ and $k = 5$ clusters. More precisely, in both cases $k = 3$ and $k = 5$, we generate k clusters by choosing $\frac{m}{k}$ independent points from the n-dimensional Gaussian distribution $\mathcal{N}(z^i, \sigma)$, $i = 1, \dots, k$ for different values of $\sigma > 0$ and with centers

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

for $k = 1, 2, 3$ and

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

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



(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 $m = 1500$, $k = 3$, $\sigma = 20$

Figure 4: Generating data set $X \subseteq \mathbb{R}^2$ with $k = 3$ and $k = 5$ clusters for varying values $\sigma > 0$

Additionally, in order 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

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

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 $i = 1, \dots, m$ 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 whereas we strive for the global solution.

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 $\sigma = 2$. In order to weaken the tendency of k-means towards the locally optimal points we randomly initialize the algorithm 5-times and use the best result. The runtime of Algorithm 1 is given in column "time" whereas the runtime of k-means is presented in column "time$_k$". The values $f^\star$ and $f_k^\star$ denote the optimal values of objective function of $\widetilde{P}$ as computed by Algorithm 1 and by k-means, respectively. The missing values in Tables 1 and 2 indicated by '-' correspond to cases where k-means did not terminate within a given time frame.

According to Tables 1 and 2 by increasing the number m of data points the runtime of Algorithm 1 increases as expected but still performs very well and in some cases even significantly better than the 5-times randomly initiated k-means algorithm. Additionally, unlike k-means, Algorithm 1 also ensures the global optimality of the computed solution. Even for larger instances such as e.g. $m = 3000$ 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. 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

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

m	n	time $[ms]$	Δ_I	Δ_O	f^*/f_k^*	time$_k$ $[ms]$
120	2	24	11.24	44.23	1	7
120	100	45	13.06	54.02	1	31
120	1000	168	14.06	56.75	1	190
120	10000	1490	15.49	58.88	1	2001
120	100000	15954	18.55	60.76	1	27713
1500	2	2781	15.72	39.88	1	472
1500	100	5389	15.90	53.27	1	3058
1500	1000	25534	17.38	56.28	-	-
3000	2	21510	13.26	40.72	1	1443
3000	100	33181	15.90	53.11	1	10912
3000	1000	110791	17.97	56.14	-	-

distance between two arbitrary data points within one cluster Δ_I meaning 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 $i \neq j$.

In order 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 $m = 25$ two-dimensional data points in Table 3.

While we increase the 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, as can be seen from Table 3, despite significantly larger runtimes our bisection method Algorithm 1 always identifies a global solution in all cases whereas the k-means sometimes terminates at a local one.

Table 2: Results for $k = 5$ clearly separated clusters, meaning $\sigma = 2$.

m	n	time $[ms]$	Δ_I	Δ_O	f^*/f_k^*	time$_k$ $[ms]$
120	2	27	10.61	44.37	1	7
120	100	43	11.88	54.07	1	42
120	1000	147	14.36	57.02	1	248
120	10000	1327	15.03	58.93	1	2377
120	100000	16392	17.41	60.66	1	28248
1500	2	1270	12.52	42.15	1	254
1500	100	3548	14.95	53.19	1	3419
1500	1000	22476	17.06	56.24	-	-
3000	2	8138	13.84	42.20	1	1019
3000	100	17143	16.45	53.08	1	7484
3000	1000	89819	16.64	56.19	-	-

6 Final remarks

In this article, a clustering algorithm for the global solution of the clustering problem is proposed. Moreover, a proof of convergence is given and some numerical results illustrate the performance of the method. However, this article is meant as a first step in the development of bisection methods to solve the clustering problem to global optimality and, hence, there are still some issues that have to be addressed.

Firstly, our numerical results show that our new method performs well and, thus, seems to be a reasonable method to solve clustering problems to global optimality. However, so far our method is only tested on artificially generated data sets in order to examine the performance of the method under different circumstances that are generated. A difficulty of our method that is encountered during these tests is a drop in performance if overlapping of clusters is encountered. 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, probably such difficulties might occur in applications and thus the need to cope with these issues arises. This is left for future research.

Moreover, our bisection method relies on graph theoretical subproblems, namely the clique and the k-cover set problem. So far, these are solved using a very simple implementation. However, by replacing these algorithms by some more sophisticated graph theoretical approaches, we expect that there are many possibilities to improve and refine the overall performance of our

Table 3: Results for different values of σ for $m = 25$ and $n = 2$.

k	σ	time $[ms]$	Δ_I	Δ_O	f^*/f_k^*	time$_k$ $[ms]$
3	2	17	11.01	46.89	1	3
3	5	19	20.38	37.52	1	4
3	10	13950	36.55	34.18	1	4
3	15	85992	46.39	26.71	1	4
3	20	1773079	69.78	23.59	1	8
5	2	9	6.57	47.32	1	3
5	5	16	18.40	41.31	0.31	6
5	10	169	38.98	39.57	1	7
5	15	55580	58.20	29.63	1	6
5	20	73526	63.52	20.33	0.91	4

method. Again, this is left for future research.

Finally, let us remark that our clustering algorithm in its current form is very 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, this difficulty is neglected or in other words, a data set that is already adjusted appropriately is implicitly assumed in order to obtain reasonable results. 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 not that affected by this issue. However, without further considerations this breaks Proposition 2.6 which is crucial for our approach.

References

[1] E. A. Akkoyunlu. *The enumeration of maximal cliques of large graphs.* SIAM Journal on Computing, 2 (1973), 1-6

[2] J. G. Augustson and J. Minker. *An analysis of some graph theoretical cluster techniques.* Journal of the ACM, 17 (1970), 571-588

[3] A. M. Bagirov. *Modified global k-means algorithm for minimum sum-of-squares clustering problems.* Pattern Recognition, 41 (2008), 3192-3199

[4] A. M. Bagirov,J. Ugon and D. Webb. *Fast modified global k-means algorithm for incremental cluster construction.* Pattern Recognition, 44 (2011), 866-876

[5] A. Ben-Dor, R. Shamir and Z. Yakhini. *Clustering gene expression patterns.* Journal of Computational Biology, 6 (1999), 281-297

[6] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications.* Macmillan, 1976

[7] R. C. Brigham and R. D. Dutton. *On clique covers and independence numbers of graphs.* Discrete Mathematics, 44 (1983), 139-144

[8] C. Bron and J. Kerbosch. *Algorithm 457: finding all cliques of an undirected graph.* Communications of the ACM, 16 (1973), 575-577

[9] R. Diestel. *Graph Theory.* Springer, 2017

[10] J. Gramm, J. Guo, F. Hüffner and R. Niedermeier. *Graph-modeled data clustering: Fixed-parameter algorithms for clique generation.* Theory of Computing Systems, 38 (2005), 373-392

[11] J. Gramm, J. Guo, F. Hüffner and R. Niedermeier. *Data reduction and exact algorithms for clique cover.* Journal of Experimental Algorithmics, 13 (2009)

[12] P. Hansen and B. Jaumard. *Cluster analysis and mathematical programming.* Mathematical Programming, 79 (1977), 191-215

[13] E. Hartuv and R. Shamir. *A clustering algorithm based on graph connectivity.* Information Processing Letters, 76 (2000), 175-181

[14] H. C. Johnston. *Cliques of a graph: Variations on the Bron-Kerbosch algorithm.* International Journal of Computer and Information Sciences, 5 (1976), 209-238

[15] R. M. Karp. *Reducibility among combinatorial problems.* In: Miller R.E., Thatcher J.W., Bohlinger J.D. (eds) Complexity of Computer Computations. The IBM Research Symposia Series. Springer, 1972, 85-103

[16] J. Kogan, C. Nicholas and M. Teboulle. *Grouping Multidimensional Data: Recent Advances in Clustering.* Springer, 2006

[17] P. Larrañaga, J. A. Lozano and J. M. Peña. *An empirical comparison of four initialization methods for the k-means algorithm.* Pattern Recognition Letters, 20 (1999), 1027-1040

[18] A. Likas, N. Vlassis and J. J. Verbeek. *The global k-means clustering algorithm.* Pattern Recognition, 36 (2003), 451-461

[19] R. D. Luce and A. D. Perry. *A method of matrix analysis of group structures.* Psychometrika, 14 (1949), 95-116

[20] J. B. MacQueen. *Some methods for classification and analysis of multivariate observations.* Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1 (1976), 281-297

[21] H. D. Sherali and W. P. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems.* Springer, 1999

[22] H. D. Sherali and J. Desai. *A global optimization RLT-based approach for solving the hard clustering problem.* Journal of Global Optimization, 32 (2005), 281-306

[23] D. B. West. *Introduction to Graph Theory.* Prentice Hall, 1996

[24] C. Zhong, D. Miao and R. Wang. *A graph-theoretical clustering method based on two rounds of minimum spanning trees.* Pattern Recognition, 43 (2010), 752-766

A Illustration of main idea

The following example with the corresponding Figure 5 illustrates the first few iterations of Algorithm 1.

Example A.1. *We consider the data set* $X := \{x^1, \dots, x^{11}\} \subseteq \mathbb{R}^2$ *with*

$$\begin{array}{lll} x^1 = (-19, -17.5)^T & x^2 = (-25, -25.8)^T & x^3 = (-22.6, -24.7)^T \\ x^4 = (18.6, 15.4)^T & x^5 = (15, 21.6)^T & x^6 = (20.2, 25.2)^T \\ x^7 = (-0.8, -0.3)^T & x^8 = (3.9, 0.1)^T & x^9 = (-4.8, 0.5)^T \\ x^{10} = (0, 2.5)^T & x^{11} = (-2.4, -3.5)^T & \end{array}$$

as depicted in Figure 5a. Moreover, we put $k = 3$, $\lambda = 0.5$ *and* $\varepsilon = 10^{-3}$. *Computing the initial lower and upper bounds according to Propositions 3.2 and 3.3 yields*

$$\underline{v}^0 := \min_{i \neq j} \ \|x^i - x^j\|_\infty = \|x^3 - x^2\|_\infty = 2.4,$$

and

$$\overline{v}^0 := \max_{i \neq j} \ \|x^i - x^j\|_\infty = \|x^6 - x^2\|_\infty = 51,$$

which in turn implies

$$v^0 := \frac{\overline{v}^0 + \underline{v}^0}{2} = 26.7.$$

We can identify three cliques

$$S_1^0 = \{x^4, x^5, x^6, x^7, x^8, x^9, x^{10}\}, \quad S_2^0 = \{x^1, x^2, x^3, x^7, x^9, x^{11}\}, \quad S_3^0 := \emptyset$$

in the graph $G(26.7) = (X, E_{26.7})$ *covering the whole data set* X*, and hence, we update the upper bound to the value* $\overline{v}^1 := v^0 = 26.7$*. The corresponding boxes of the cliques* S_1^0, S_2^0 *and* S_3^0 *are illustrated in Figure 5b. Since the lower bound is not changed, i.e.* $\underline{v}^1 := \underline{v}^0 = 2.4$*, in the next iteration we consider the value*

$$v^1 := \frac{\overline{v}^1 + \underline{v}^1}{2} = \frac{26.7 + 2.4}{2} = 14.55.$$

This yields three cliques (see Figure 5c)

$$S_1^1 = \{x^1, x^2, x^3\}, \quad S_2^1 = \{x^4, x^5, x^6\}, \quad S_3^1 = \{x^7, \ldots, x^{11}\}$$

in the graph $G(14.55)$ *with* $S_1^1 \cup S_2^1 \cup S_3^1 = X$ *and a subsequent upper bound update* $\overline{v}^2 := v^1 = 14.55$*. Since*

$$\overline{v}^2 - \underline{v}^2 = 14.55 - 2.4 \geq 10^{-3} = \varepsilon,$$

in the next iteration we consider the value

$$v^2 := \frac{14.55 + 2.4}{2} = 8.475$$

which yields four cliques

$$S_1^2 = \{x^1, x^2, x^3\}, \quad S_2^2 = \{x^4, x^5\}, \quad S_3^2 = \{x^7, \ldots, x^{11}\}, \quad S_4^2 = \{x^5, x^6\}$$

in the graph $G(8.475)$*. An illustration of the graph* $G(8.475)$ *is given in Figure 5d and an illustration of boxes corresponding to the four cliques* $S_1^2, \ldots S_4^2$ *is given in Figure 5e.*

Since there do not exist any $k = 3$ *cliques in the graph* $G(8.475)$*, according to Proposition 2.6, we can increase the lower bound to* $\underline{v}^3 := v^2 = 8.475$*. Performing further iterations of Algorithm 1 leads to sequences* $(\underline{v}^\zeta)_{\zeta \in \mathbb{N}}$ *and* $(\overline{v}^\zeta)_{\zeta \in \mathbb{N}}$ *converging to the value* $2v^\star = 9.8 = \|x^6 - x^4\|_\infty$*, indicating that the distance between the data points* x^4 *and* x^6 *is responsible for the globally minimal objective value of our clustering problem* $\widetilde{P}$*. Finally we can identify three cliques*

$$S_1^\star = \{x^1, x^2, x^3\}, \quad S_2^\star = \{x^4, x^5, x^6\}, \quad S_3^\star = \{x^7, x^8, x^9, x^{10}, x^{11}\}$$

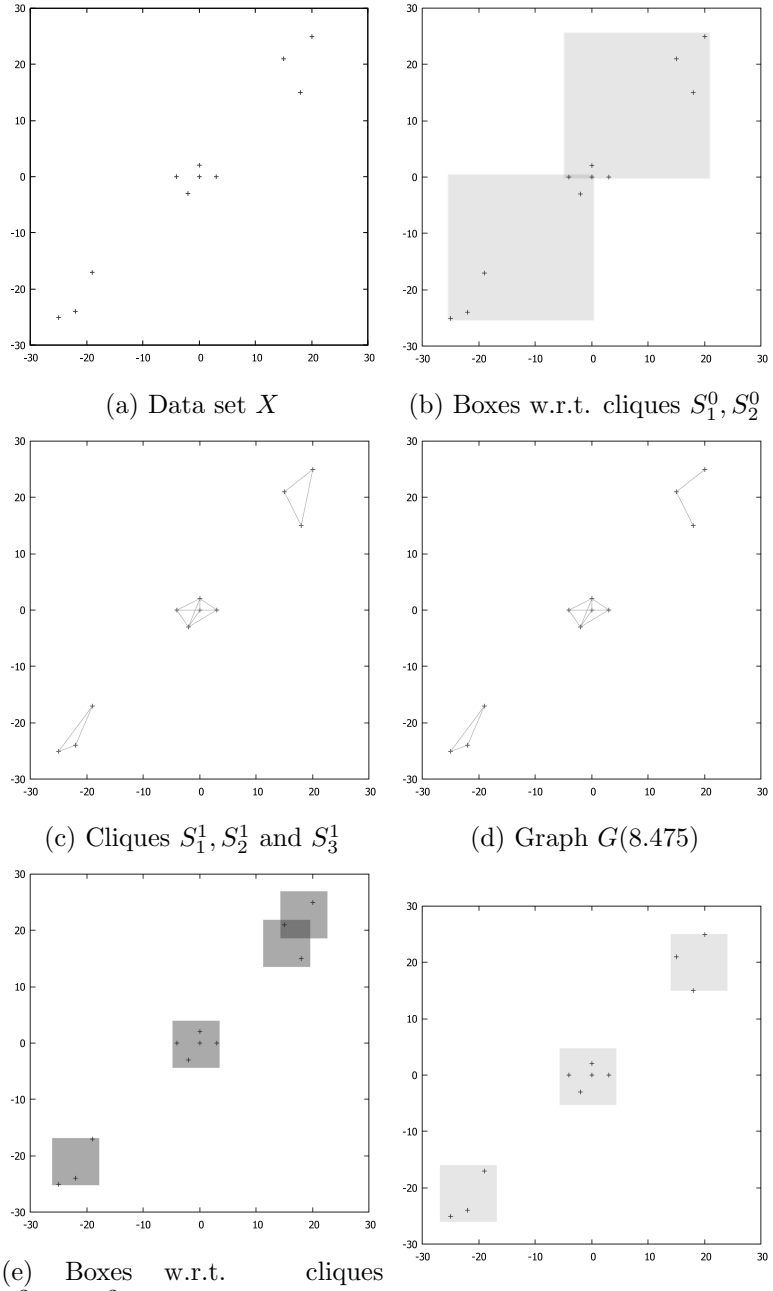
in the graph $G(2v^\star) = G(9.8)$ *and according to the construction* (3.1)-(3.3) *we can compute the optimal cluster centers*

$$z^{1\star} := (-22, -21.65)^T, \quad z^{2\star} := (17.6, 20.3)^T, \quad z^{3\star} := (-0.45, -0.5)^T$$

together with the corresponding boxes (see Figure 5f)

$$\begin{aligned} B_1^\star &:= [-25, -19] \times [-25.8, -17.5] \\ B_2^\star &:= [15, 20.2] \times [15.4, 25.2] \\ B_3^\star &:= [-4.8, 3.9] \times [-3.5, 2.5] \end{aligned}$$

which represent a globally minimal point $(z^{1\star}, z^{2\star}, z^{3\star}, v^\star)^T$ *of our clustering problem* $\widetilde{P}$*.*



(a) Data set X

(b) Boxes w.r.t. cliques S_1^0, S_2^0

(c) Cliques S_1^1, S_2^1 and S_3^1

(d) Graph $G(8.475)$

(e) Boxes w.r.t. cliques $S_1^2, \ldots, S_4^2$

(f) Optimal boxes $B_1^\star, B_2^\star$ and $B_3^\star$

Figure 5: Illustration of Algorithm 1 from Example A.1