Parameterized k-Clustering: Tractability Island

Fedor V. Fomin
Department of Informatics, University of Bergen, Norway
Fedor.Fomin@uib.no

Petr A. Golovach
Department of Informatics, University of Bergen, Norway
Petr.Golovach@uib.no

Kirill Simonov
Department of Informatics, University of Bergen, Norway
Kirill.Simonov@uib.no

Abstract

In k-Clustering we are given a multiset of n vectors $X \subset \mathbb{Z}^d$ and a nonnegative number D, and we need to decide whether X can be partitioned into k clusters $C_1, \ldots, C_k$ such that the cost

$$\sum_{i=1}^{k} \min_{c_i \in \mathbb{R}^d} \sum_{x \in C_i} \|x - c_i\|_p^p \leq D,$$

where $\|\cdot\|_p$ is the Minkowski (L_p) norm of order p. For $p = 1$, k-Clustering is the well-known k-Median. For $p = 2$, the case of the Euclidean distance, k-Clustering is k-Means. We study k-Clustering from the perspective of parameterized complexity. The problem is known to be NP-hard for $k = 2$ and it is also NP-hard for $d = 2$. It is a long-standing open question, whether the problem is fixed-parameter tractable (FPT) for the combined parameter $d+k$. In this paper, we focus on the parameterization by D. We complement the known negative results by showing that for $p = 0$ and $p = \infty$, k-Clustering is W[1]-hard when parameterized by D. Interestingly, the complexity landscape of the problem appears to be more intricate than expected. We discover a tractability island of k-Clustering: for every $p \in (0, 1]$, k-Clustering is solvable in time $2^{\mathcal{O}(D \log D)}(nd)^{\mathcal{O}(1)}$.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases clustering, parameterized complexity, k-means, k-median

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2019.14

Related Version The full version of this paper is available at `https://arxiv.org/abs/1902.08559`.

Funding This work is supported by the Research Council of Norway via the project "MULTIVAL".

1 Introduction

Recall that for $p > 0$, the *Minkowski* or *L_p-norm* of a vector $x = (x[1], \ldots, x[d]) \in \mathbb{R}^d$ is defined as

$$\|x\|_p = \left(\sum_{i=1}^{d} |x[i]|^p\right)^{1/p}.$$

Respectively, we define the *(L_p-norm) distance* between two vectors $x = (x[1], \ldots, x[d])$ and $y = (y[1], \ldots, y[d])$ as

$$\mathrm{dist}_p(x, y) = \|x - y\|_p^p = \sum_{i=1}^{d} |x[i] - y[i]|^p.$$

39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2019).
Editors: Arkadev Chattopadhyay and Paul Gastin; Article No. 14; pp. 14:1–14:15

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We also consider dist_p for $p = 0$ and $p = \infty$. For $p = 0$, dist_p is L_0 (or the Hamming) distance, that is the number of different coordinates in x and y:

$$\mathrm{dist}_0(x, y) = |\{i \in \{1, \ldots, d\} \mid x[i] \neq y[i]\}|.$$

For $p = \infty$, dist_p is L_∞-distance, which is defined as

$$\mathrm{dist}_\infty(x, y) = \max_{i \in \{1,\ldots,d\}} |x[i] - y[i]|.$$

The k-Clustering problem is defined as follows. For a given (multi) dataset of n vectors (points) $X \subset \mathbb{Z}^d$, the task is to find a partition of X into k clusters $C_1, \ldots, C_k$ minimizing the cost

$$\sum_{i=1}^{k} \min_{c_i \in \mathbb{R}^d} \sum_{x \in C_i} \mathrm{dist}_p(x, c_i),$$

intuitively, c_i is a centroid of the cluster C_i.

In particular, for $p = 1$, dist_p is the L_1-distance and the corresponding clustering problem is known as k-Median. (Often in the literature, k-Median is also used for clustering minimizing the sums of the Euclidean distances.) For $p = 2$, dist_p is the L_2 (Euclidean) distance, and then the clustering problem becomes k-Means.

Let us note that optimal clusterings for the same set of vectors can be drastically different for various values of p, as shown in Figure 1. As we show in the paper, the complexity of k-Clustering also strongly depends on the choice of p.

Figure 1 Optimal clusterings of the same set of vectors with different distances: dist_1 in the left subfigure, $\mathrm{dist}_{1/4}$ in the right subfigure. Shapes denote clusters, crosses denote cluster centroids.

k-Clustering, and especially k-Median and k-Means, are among the most prevalent problems occurring in virtually every subarea of data science. We refer to the survey of Jain [22] for an extensive overview. While in practice the most common approaches to clustering are based on different variations of Lloyd's heuristic [25], the problem is interesting from the theoretical perspective as well. In particular, there is a vast amount of literature on approximation algorithms for k-Clustering whose behavior can be analyzed rigorously, see e.g. [1, 2, 6, 8, 9, 16, 17, 20, 24, 13, 23, 10, 30].

When it comes to exact solutions, we observe the following phenomena. While heuristic algorithms for k-Clustering work surprisingly well in practice, from the perspective of the parameterized complexity, k-Clustering is intractable for all previously studied parameterizations, see Table 1. The k-Clustering problem is naturally "multivariate": in addition to the input size n, there are also parameters like space dimension d, number of clusters k or the cost of clustering D. The problem is known to be NP-complete for $k = 2$ [3, 15] and for $d = 2$ [28, 26]. By the classical work of Inaba et al. [21], in the case when both d and k are constants, k-Clustering is solvable in polynomial time $\mathcal{O}(n^{dk+1})$. It is a

long-standing open problem whether k-Clustering is FPT parameterized by $d+k$. Under ETH, the lower bound of $n^{\Omega(k)}$, even when $d=4$, was shown by Cohen-Addad et al. in [11] for the settings where the set of potential candidate centers is explicitly given as input. However the lower bound of Cohen-Addad et al. does not generalize to the settings of this paper when any point in Euclidean space can serve as a center. For the special case, when the input consists of binary vectors and the distance is Hamming, the problem is solvable in time $2^{\mathcal{O}(D\log D)}(nd)^{\mathcal{O}(1)}$ [18].

Our results and approaches. In this paper we investigate the dependence of the complexity of k-Clustering from the cost of clustering D. It appears, that adding this new "dimension" makes the complexity landscape of k-Clustering intricate and interesting. More precisely, we consider the following problem.

k-Clustering with distance dist

Input: A multiset X of n vectors in $\mathbb{Z}^d$, a positive integer k, and a nonnegative number D.

Task: Decide whether there is a partition of X into k clusters $\{C_i\}_{i=1}^k$ and k vectors $\{c_i\}_{i=1}^k$, called *centroids*, in $\mathbb{R}^d$ such that

$$\sum_{i=1}^{k}\sum_{x\in C_i}\mathrm{dist}(x,c_i)\leq D.$$

Let us remark that vector set X (like the column set of a matrix) can contain many equal vectors. Also we consider the situation when vectors from X are integer vectors, while centroid vectors are not necessarily from X. Moreover, coordinates of centroids can be reals.

Our main algorithmic result is the following theorem.

▶ **Theorem 1.** *k-Clustering with distance dist_p is solvable in time $2^{\mathcal{O}(D\log D)}(nd)^{\mathcal{O}(1)}$ for every $p\in(0,1]$.*

Thus k-Clustering when parameterized by D is fixed-parameter tractable (FPT) for Minkowski distance dist_p of order $0<p\leq 1$. In the first step of our algorithm we use color coding to reduce solution of the problem to the Cluster Selection problem, which we find interesting on its own. In Cluster Selection we have t groups of weighted vectors and the task is to select exactly one vector from each group such that the weighted cost of the composite cluster is at most D. More formally,

Cluster Selection with distance dist

Input: A set of m vectors X given together with a partition $X=X_1\cup\cdots\cup X_t$ into t disjoint sets, a weight function $w: X\to\mathbb{Z}_+$, and a nonnegative number D.

Task: Decide whether it is possible to select exactly one vector x_i from each set X_i such that the total cost of the composite cluster formed by $x_1,\ldots,x_t$ is at most D:

$$\min_{c\in\mathbb{R}^d}\sum_{i=1}^{t}w(x_i)\cdot\mathrm{dist}(x_i,c)\leq D.$$

The Cluster Selection problem is closely related to variants of the well-known Consensus Pattern problem. Namely, for the Hamming distance, the definition of Cluster Selection nearly coincides with the Colored Consensus Strings with Outliers problem studied in [7], only in the latter the alphabet is assumed to be of constant size.

Informally (see Theorem 10 for the precise statement), our reduction shows that if the distance norm satisfies some specific properties (which dist_p satisfies for all p) and if Cluster Selection is FPT parameterized by D, then so is k-Clustering. Therefore, in order to prove Theorem 1, all we need is to show that Cluster Selection is FPT parameterized by D when $p \in (0, 1]$. This is the most difficult part of the proof. Here we invoke the theorem of Marx [27] on the number of subhypergraphs in hypergraphs of bounded fractional edge cover.

Superficially, the general idea of the proof of Theorem 1 is similar to the idea behind the algorithm for Binary r-Means for L_0 from [18]. In both cases, the classical color coding technique of Alon et al. [4] is used as a preprocessing step. However, the further steps in [18] strongly exploit the fact that the data is binary. As we will see in Theorem 2, the existence of an FPT algorithm for k-Clustering in L_0 is highly unlikely. Thus the reductions from [18] cannot be applied in our case, and we need a new approach.

More precisely, for clustering in L_0 we prove the following theorem.

▶ **Theorem 2.** *With distance* dist_0*,* k-Clustering *parameterized by* $d + D$ *and* Cluster Selection *parameterized by* $d + t + D$ *are* W[1]*-hard.*

In particular, this means that up to a widely-believed assumption in complexity that FPT $\neq$ W[1], Theorem 2 rules out algorithms solving k-Clustering in time $f(d, D) \cdot n^{\mathcal{O}(1)}$ and algorithms solving Cluster Selection in L_0 in time $g(t, d, D) \cdot n^{\mathcal{O}(1)}$ for any functions $f(d, D)$ and $g(t, d, D)$. A similar hardness result holds for L_∞.

▶ **Theorem 3.** *With distance* dist_∞*,* k-Clustering *parameterized by* D *and* Cluster Selection *parameterized by* $t + D$ *are* W[1]*-hard.*

This naturally brings us to the question: What happens with k-Clustering for $p \in (1, \infty)$, especially for the Euclidean distance, that is $p = 2$. Unfortunately, we are not able to answer this question when the parameter is D only. However, we can prove that

▶ **Theorem 4.** k-Clustering *and* Cluster Selection *with distance* dist_2 *are* FPT *when parameterized by* $d + D$*.*

Thus in particular, Theorem 4 implies that k-Clustering with distance dist_2 is FPT parameterized by $d + D$. On the other hand, we prove that

▶ **Theorem 5.** Cluster Selection *with distance* dist_p *is* W[1]*-hard for every* $p \in (1, \infty)$ *when parameterized by* $t + D$ *.*

In particular, Theorem 5 yields that the approach we used to establish the tractability (with parameter D) of k-Clustering for $p = 1$ will not work for $p > 1$.

We summarize our and previously known algorithmic and hardness results for the problems k-Clustering and Cluster Selection with different distances in Table 1.

In the extended abstract, we provide a full proof of Theorems 1 and 15. Proofs of Theorems 2, 3, 4, 5, 19 and 28 can be found in the full version of this paper [19].

The remaining part of this paper is organized as follows. Section 2 contains preliminaries. In Section 3 we prove Theorem 10 which provides us with FPT Turing reduction from k-Clustering to Cluster Selection. Theorem 10 appears to be a handy tool to establish

Table 1 Complexity of k-Clustering and Cluster Selection.

dist_p	k-Clustering	Cluster Selection
$p = 0$	W[1]-hard param. $d + D$ [Thm 2] NP-c for $k = 2$ [15]	W[1]-hard param. $d + t + D$ [Thm 2]
$0 < p \leq 1$	$2^{\mathcal{O}(D \log D)}(nd)^{\mathcal{O}(1)}$ [Thm 1] NP-c for $k = 2$ when $p = 1$ [15] NP-c for $d = 2$ when $p = 1$ [28]	$2^{\mathcal{O}(D \log D)}(nd)^{\mathcal{O}(1)}$ [Thm 15] W[1]-hard param. $t + d$ for $p = 1$ [Thm 19]
$1 < p < \infty$	FPT param. $d + D$ for $p = 2$ [Thm 4] NP-c for $k = 2$ when $p = 2$ [3] NP-c for $d = 2$ when $p = 2$ [26]	FPT param. $d + D$ for $p = 2$ [Thm 4] W[1]-hard param. $t + D$ [Thm 5]
$p = \infty$	W[1]-hard param. D [Thm 3] NP-c for $k = 2$ [Thm 28]	W[1]-hard param. $t + D$ [Thm 3]

tractability of k-Clustering. In Section 4 we prove Theorem 1, the main algorithmic result of this work, stating that when $p \in (0, 1]$, k-Clustering and Cluster Selection admit FPT algorithms with parameter D. We conclude with open problems in Section 5.

2 Preliminaries and notation

Cluster notation. By a *cluster* we always mean a multiset of vectors in $\mathbb{Z}^d$. For distance dist, the *cost* of a given cluster C is the total distance from all vectors in the cluster to the optimally selected cluster centroid, $\min_{c \in \mathbb{R}^d} \sum_{x \in C} \text{dist}(x, c)$. An *optimal* cluster centroid for a given cluster C is any $c \in \mathbb{R}^d$ minimizing $\sum_{x \in C} \text{dist}(x, c)$. For most of the considered distances, we argue that an optimal cluster centroid could always be chosen among selected family of vectors (e.g. integral). Whenever we show this, we only consider optimal cluster centroids of the stated form afterwards.

Complexity. A *parameterized problem* is a language $Q \subseteq \Sigma^* \times \mathbb{N}$ where Σ^* is the set of strings over a finite alphabet Σ. Respectively, an input of Q is a pair (I, k) where $I \subseteq \Sigma^*$ and $k \in \mathbb{N}$; k is the *parameter* of the problem. A parameterized problem Q is *fixed-parameter tractable* (FPT) if it can be decided whether $(I, k) \in Q$ in time $f(k) \cdot |I|^{\mathcal{O}(1)}$ for some function f that depends of the parameter k only. Respectively, the parameterized complexity class FPT is composed by fixed-parameter tractable problems. The W-hierarchy is a collection of computational complexity classes: we omit the technical definitions here. The following relation is known amongst the classes in the W-hierarchy: $\text{FPT} = \text{W}[0] \subseteq \text{W}[1] \subseteq \text{W}[2] \subseteq \ldots \subseteq \text{W}[P]$. It is widely believed that $\text{FPT} \neq \text{W}[1]$, and hence if a problem is hard for the class $\text{W}[i]$ (for any $i \geq 1$) then it is considered to be fixed-parameter intractable. We refer to books [12, 14] for the detailed introduction to parameterized complexity.

Real computations. As is usual in computational geometry, we express the running time of algorithms in terms of number of operations over the reals. This is natural since to compute L_p-distances we have to deal with numbers of form x^p where x is an integer and p is any real number. However, in special cases the bounds hold even for more restrictive models, e.g. when $p = 1$ or $p = 2$ the algorithms operate only on integers of polynomially bounded length.

3 From k-Clustering to Cluster Selection

In this section we present a general scheme for obtaining an FPT algorithm parameterized by D, which is later applied to various distances.

First, we formalize the following intuition: there is no reason to assign equal vectors to different clusters.

▶ **Definition 6** (Initial cluster and regular partition). *For a multiset of vectors X, an inclusion-wise maximal multiset $I \subset X$ such that all vectors in I are equal is called* an initial cluster.

We say that a clustering $\{C_1, \ldots, C_k\}$ of X is regular *if for every initial cluster I there is a $i \in \{1, \ldots, k\}$ such that $I \subset C_i$.*

Now we prove that it suffices to look only for regular solutions.

▶ **Proposition 7.** *Let (X, k, D) be a yes-instance to* k-Clustering. *Then there exists a solution of (X, k, D) which is a regular clustering.*

Proof. Let us assume that the instance (X, k, D) has a solution. There are k clusters $\{C_i\}_{i=1}^k$ and k vectors $\{c_i\}_{i=1}^k$ in $\mathbb{R}^d$ such that $\sum_{i=1}^k \sum_{x \in C_i} \operatorname{dist}(x, c_i) \leq D$. Note that for every $x \in C_j$, $\operatorname{dist}(x, c_j) \geq \min_{1 \leq i \leq k} \operatorname{dist}(x, c_i)$. So if we consider a new clustering $\{C'_1, \ldots, C'_k\}$ with the same centroids, where C'_j are all vectors from X for which c_j is the closest centroid, the total distance does not increase. If we also break ties in favor of the lower index, then for any initial cluster I the same centroid c_i will be the closest, and all vectors from I will end up in C'_i, so $\{C'_1, \ldots, C'_k\}$ is a regular clustering. ◀

From now on, we consider only regular solutions.

▶ **Definition 8** (Simple and composite clusters). *We say that a cluster C is* simple *if it is an initial cluster. Otherwise, the cluster is* composite.

Next we state a property of k-Clustering with a particular distance, which is required for the algorithm. Intuitively, each unique vector adds at least some constant to the cluster cost. In the subsequent sections we show that the property holds for all distances in our consideration.

▶ **Definition 9** (α-property). *We say that a distance has the* α-property *for some $\alpha > 0$ if for any s the cost of any composite cluster which consists of s initial clusters is at least $\alpha(s-1)$.*

The Cluster Selection problem defined in the introduction is a key subroutine in our algorithm. In some cases the problem is solvable trivially, but it presents the main challenge for our main algorithmic result with the L_1 distance. The intuition to the weight function in the definition of Cluster Selection is that it represents sizes of initial clusters, that is, how many equal vectors are there.

We also need a procedure to enumerate all possible optimal cluster costs which are less than D. It may not be straightforward since not all distances in our consideration are integer. So we assume that the set of all possible optimal cluster costs which are less than D is also given in the input. Now we are ready to state the result formally.

▶ **Theorem 10.** *Assume that the α-property holds,* Cluster Selection *is solvable in time $\Phi(m, d, t, D)$, where Φ is a non-decreasing function of its arguments, and we are given the set $\mathcal{D}$ of all possible optimal cluster costs which are at most D. Then* k-Clustering *is solvable in time*

$$2^{\mathcal{O}(D \log D)}(nd)^{\mathcal{O}(1)}|\mathcal{D}|\Phi(n, d, 2D/\alpha, D).$$

Proof. By the α-property, in any solution there are at most D/α composite clusters, since each contains at least two initial clusters. Moreover, there are at most $2D/\alpha$ initial clusters in all composite clusters.

Thus by Proposition 7, solving k-Clustering is equivalent to selecting at most $T := \lceil 2D/\alpha \rceil$ initial clusters and grouping them into composite clusters such that the total cost of these clusters is at most D. We design an algorithm which, taking as a subroutine an algorithm for Cluster Selection, solves k-Clustering. An example is shown in Figure 2.

To perform the selection and grouping, our algorithm uses the color coding technique of Alon, Yuster, and Zwick from [4]. Consider the input as a family of initial clusters $\mathcal{I}$. We color initial clusters from $\mathcal{I}$ independently and uniformly at random by T colors $1, 2, \ldots, T$. Consider any solution, and the particular set of at most T initial clusters which are included into composite clusters in this solution. These initial clusters are colored by distinct colors with probability at least $\frac{T!}{T^T} \geq e^{-T}$. Now we construct an algorithm for finding a colorful solution.

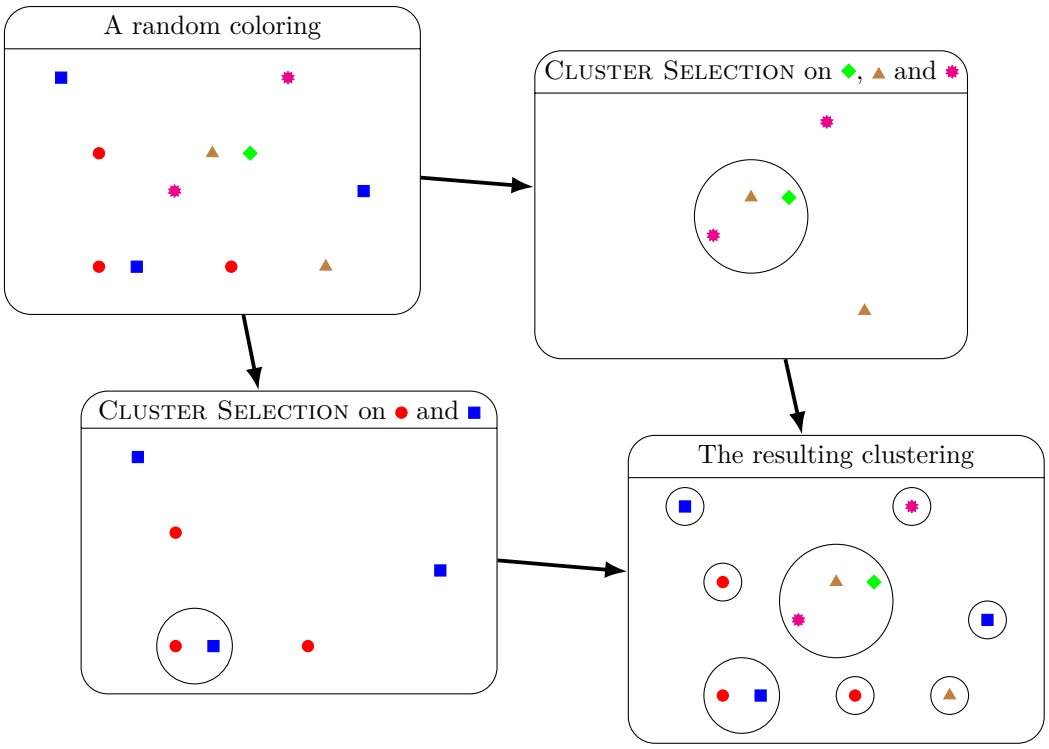

Figure 2 An illustration to the algorithm in Theorem 10. We start with a particular random coloring and a particular partition of colors $\mathcal{P} = \{P_1, P_2\}$, where $P_1 = \{●, ■\}$ and $P_2 = \{◆, ▲, ✸\}$. We make two calls to Cluster Selection with respect to P_1 and P_2 and construct the resulting clustering. In the example, all input vectors are distinct.

We consider all possible ways to split colors between clusters (some colors may be unused). Hence we consider all possible families $\mathcal{P} = \{P_1, \ldots, P_h\}$ of pairwise disjoint non-empty subsets of $\{c \in \{1, \ldots, T\} :$ there exists $J \in \mathcal{I}$ colored by $c\}$. Each family $\mathcal{P}$ corresponds to a partition of the set of colors $\{1, \ldots, T\}$ if we add one fictitious subset for colors which are not used in the composite clusters. The total number of partitions does not exceed $T^T = 2^{\mathcal{O}(D \log D)}$.

When partition $\mathcal{P}$ is fixed, we form clusters by solving instances of Cluster Selection: For each $i \in \{1, \ldots, h\}$, we take initial clusters colored by elements of P_i, bundle together those with the same color, and pass the resulting family to Cluster Selection. First note that there cannot be $P \in \mathcal{P}$ of size at most one, since then Cluster Selection has to make a simple cluster while we assume that all clusters obtained from $\mathcal{P}$ are composite. Second, the total number of clusters has to be k, the number of clusters is $|\mathcal{I}| - \sum_{P \in \mathcal{P}} |P| + |\mathcal{P}|$. For each $\mathcal{P}$ we check that both conditions hold, and if not, we discard the choice of $\mathcal{P}$ and move to the next one, before calling the Cluster Selection subroutine.

Next, we formalize how we call the Cluster Selection subroutine. We fix the set of colors $P_i = \{c_1, \ldots, c_t\}$, then take the sets $I_j = \{J \in \mathcal{I} : J \text{ is colored by } c_j\}$ for $j \in \{1, \ldots, t\}$. We turn each set of initial clusters I_j into a set of weighted vectors X_j naturally: For each $J \in I_j$, we put one vector $x \in J$ into X_j, and $w(x) := |J|$. The family of sets of vectors X_1, $\ldots$, X_t and the weight function w are the input for Cluster Selection. Then we search for the minimum cluster cost bound $d_i \leq D$ from $\mathcal{D}$, for which the instance $(X_1, \ldots, X_t, d_i)$ of Cluster Selection is a yes-instance, running each time the algorithm for Cluster Selection.

If for some i setting d_i to D leads to a no-instance, or if $\sum_{i=1}^{h} d_i > D$, then we discard the choice of the partition $\mathcal{P}$ and move to the next one. Otherwise, we report that k-Clustering has a solution and stop. Next, we prove that in this case the solution indeed exists.

We reconstruct the solution to k-Clustering as follows: For each $i \in \{1, \ldots, h\}$ the corresponding to $P_i = \{c_1, \ldots, c_t\}$ instance of Cluster Selection has a solution $\{x_1, \ldots, x_t\}$. For each $j \in \{1, \ldots, t\}$, consider the corresponding initial cluster J_j consisting of $w(x_j)$ vectors equal to x_j. For each $i \in \{1, \ldots, h\}$ we obtain a composite cluster $\cup_{j=1}^{t} J_j$, all other clusters are simple. So the total cost is $\sum_{i=1}^{h} d_i$, which is at most D. Thus, if the algorithm finds a solution, then (X, d, D) is a yes-instance.

In the opposite direction. If there is a solution to k-Clustering, then there is a regular solution, and with probability at least e^{-T} initial clusters which are parts of composite clusters in this solution are colored by distinct colors. Then, there is a partition $\mathcal{P} = \{P_1, \ldots, P_h\}$ which corresponds to this solution. This partition is obtained as follows: put into P_1 colors from the first composite cluster, into P_2 from the second and so on. At some point our algorithm checks the partition $\mathcal{P}$, and as it finds the optimal cost value for each cluster, then it is at most the cost of the corresponding cluster of the solution from which we started.

To analyze the running time, we consider $2^{\mathcal{O}(D \log D)}$ partitions $\mathcal{P}$, for each $\mathcal{P}$ we $|\mathcal{P}| = \mathcal{O}(D)$ times search for optimal d_i. And for each of $|\mathcal{D}|$ possible values [1] of d_i we make one call to the Cluster Selection algorithm, which takes time at most $\Phi(n, d, T, D)$.

To amplify the error probability to be at least $1/e$, we do $N = \lceil e^T \rceil$ iterations of the algorithm, each time with a new random coloring. As each iteration succeeds with probability at least e^{-T}, the probability of not finding a colorful solution after N iterations is at most $(1 - e^{-T})^{e^T} \leq e^{-1} < 1$. So the total running time is $2^{\mathcal{O}(D \log D)} \cdot (nd)^{\mathcal{O}(1)} |\mathcal{D}| \Phi(n, d, 2D/\alpha, D)$.

The algorithm could be derandomized by the standard derandomization technique using perfect hash families [4, 29]. So k-Clustering is solvable in the same deterministic time. ◀

[1] We could also binary search for the optimal $d_i \in \mathcal{D}$ instead, thus replacing $|\mathcal{D}|$ by $\log |\mathcal{D}|$ in the running time. However, for all choices of $\mathcal{D}$ we consider this does not make a difference.

4 Algorithms and complexity for distances with $p \in (0, 1]$

The main motivation for the results in this section is the study of k-Clustering with the L_1 distance, the case widely known as k-Medians. However, our main algorithmic result also extends to distances of order $p \in (0, 1)$ since in some sense they behave similarly to the L_1 distance.

4.1 FPT algorithm when parameterized by D

In this subsection, we prove Theorem 1: when $p \in (0, 1]$, k-Clustering admits an FPT algorithm with parameter D. First we state basic geometrical observations for cases $p = 1$ and $p \in (0, 1)$, Then we propose a general algorithm for Cluster Selection which relies only on these properties. Finally, we show how Theorem 10 could be applied.

The next two claims deal with the structure of optimal cluster centroids. We state and prove them in the case of weighted vectors where each vector has a positive integer weight given by a weight function w. The unweighted case is just a special case when the weight of each vector is one. The proofs of the claims are straightforward and are available in the full version of this paper.

First, we show that coordinates of cluster centroids could always be selected among the values present in the input, which helps greatly in enumerating cluster centroids that may be optimal.

▷ Claim 11. Assume $p \in (0, 1]$, let $C = \{x_1, \ldots, x_t\}$ be a cluster and $w : \{x_1, \cdots, x_t\} \to \mathbb{Z}_+$ be a weight function. There is an optimal (subject to the weighted distance $w(x_i) \cdot \mathrm{dist}_p(x_i, c)$) centroid c of C such that for each $i \in \{1, \ldots, d\}$, the i-th coordinate $c[i]$ of the centroid is from the values present in the input in this coordinate, that is $c[i] \in \{x_1[i], \ldots, x_t[i]\}$. Moreover, for $p = 1$ we may assume that the optimal value is a weighted median of the values present in the i-th coordinate.

In particular, by Claim 11 we may assume that the coordinates of optimal cluster centroids are integers. Then, the α-property holds with $\alpha = 1$ since at most one of the initial clusters could have distance zero to the cluster centroid, and all others have distance at least one since the cluster centroid is integral. Namely, let x be a vector in the cluster, and c be the cluster centroid, if $x \neq c$, then there is a coordinate j where x and c differ, and since they are both integral, $|x[j] - c[j]| \geq 1$, and

$$\mathrm{dist}_p(x, c) = \sum_{i=1}^{d} |x[i] - c[i]|^p \geq |x[j] - c[j]|^p \geq 1^p = 1.$$

In what follows, the expression *half of vectors by weight* means that the total weight of the corresponding set of vectors is at least half of the total weight of C.

▷ Claim 12. If at least half of the vectors by weight in the cluster C have the same value z in some coordinate i, then the optimal cluster centroid is also equal to z in this coordinate.

In order to apply Theorem 10, we need an FPT algorithm for Cluster Selection. Before obtaining it, we state some properties of hypergraphs, which we need for the algorithm.

A *hypergraph* G is a set of vertices $V(G)$ and a collection of hyperedges $E(G)$, each hyperedge is a subset of $V(G)$. If G and H are hypergraphs, we say that H *appears* at $V' \subset V(G)$ as a *subhypergraph* if there is a bijection $\pi : V(H) \to V'$ with a property that for any $E \in E(H)$ there is $E' \in E(G)$ such that $\pi(E) = E' \cap V'$, the action of π is extended to subsets of $V(H)$ in a natural way.

A *fractional edge cover* of a hypergraph H is an assignment $\psi : E(H) \to [0,1]$ such that for every $v \in V(H)$, $\sum_{E \in E(H): v \in E} \psi(E) \geq 1$. The *fractional cover number* $\rho^*(H)$ is the minimum of $\sum_{E \in E(H)} \psi(E)$ taken over all fractional edge covers ψ.

We need the following result of Marx [27] about finding occurences of one hypergraph in another.

▶ **Lemma 13** ([27]). *Let H be a hypergraph with fractional cover number $\rho^*(H)$, and let G be a hypergraph where each hyperedge has size at most ℓ. There is an algorithm that enumerates in time $|V(H)|^{\mathcal{O}(|V(H)|)} \cdot \ell^{|V(H)|\rho^*(H)+1} \cdot |E(G)|^{\rho^*(H)+1} \cdot |V(G)|^2$ every subset $V' \subset V(G)$ where H appears in G as a subhypergraph.*

Also, the following version of the Chernoff Bound will be of use.

▶ **Proposition 14** ([5]). *Let $X_1, X_2, \ldots, X_n$ be independent 0-1 random variables. Denote $X = \sum_{i=1}^{n} X_i$ and $\mu = E[X]$. Then for $0 < \beta \leq 1$,*

$$P[X \leq (1-\beta)\mu] \leq \exp(-\beta^2\mu/2),$$
$$P[X \geq (1+\beta)\mu] \leq \exp(-\beta^2\mu/3).$$

We are ready to proceed with the proof that Cluster Selection with $p \in (0,1]$ is FPT when parameterized by D.

▶ **Theorem 15.** *For every $p \in (0,1]$,* Cluster Selection *with distance* dist_p *is solvable in time $2^{\mathcal{O}(D \log D)}(md)^{\mathcal{O}(1)}$.*

Proof. First we check if any of the given vectors could be the centroid of the resulting composite cluster. When the centroid is fixed, we find the optimal solution in polynomial time by just selecting the cheapest vector with respect to this centroid from each set. If at some point we find a suitable centroid, then we return that the solution exists. If not, we may assume that the centroid is not equal to any of the given vectors. As a consequence, any vector x selected into the solution cluster contributes at least $w(x)$ to the total distance, since the centroid must be integral by Claim 11. So we may now consider only vectors of weight at most D and, moreover, the total weight of the resulting cluster is at most D.

Consider a resulting cluster C with the centroid c. There is some x_1 in C from X_1, and $\mathrm{dist}_p(x_1, c) \leq D$. So if we try all possible x_1 from X_1 (there are at most m of them), any feasible centroid is at distance at most D from at least one of them. Since x_1 and c are integral, they could be different in at most D coordinates, as $\mathrm{dist}_p(x_1, c) = \sum_{i=1}^{d} |x_1[i] - c[i]|^p \leq D$.

We try all possible $x_1 \in X_1$. After x_1 is fixed, we enumerate all subsets P of coordinates $\{1, \ldots, d\}$ where x_1 and c could differ, we show how to do it efficiently afterwards. When the subset of coordinates P is fixed, we consider all possible centroids, which are integral, equal to x_1 in all coordinates except P, and differ from x_1 by at most $D^{1/p}$ in each of coordinates from P. If $|x_1[i^*] - c[i^*]| > D^{1/p}$ for some coordinate i^*, then $\mathrm{dist}_p(x_1, c) = \sum_{i=1}^{d} |x_1[i] - c[i]|^p \geq |x_1[i^*] - c[i^*]|^p > D$, so c can not be a centroid. With restrictions stated above, there are at most $2^{\mathcal{O}(D \log D)}$ possible centroids.

It remains to show that we could enumerate all possible coordinate subsets efficiently. We reduce this task to the task of finding a specific subhypergraph and then apply Lemma 13.

▷ Claim 16. There are $2^{\mathcal{O}(D \log D)}$ coordinate subsets where x_1 and an optimal cluster centroid c could differ. There exists an algorithm which enumerates all of them in time $2^{\mathcal{O}(D \log D)}(md)^{\mathcal{O}(1)}$.

Proof. Let G be a hypergraph with $V(G) = \{1, \ldots, d\}$, one vertex for each coordinate, and for each vector x in $\cup_{j=1}^{t} X_j$ we take $w(x)$ multiple hyperedges E_x which contains exactly the coordinates where x and x_1 differ. We add an edge only if there are at most D such coordinates, otherwise x can not be in the same cluster as x_1. So hyperdeges in G are of size at most D. Since we consider only vectors of weight at most D, $|E(G)| \leq Dm$.

For a solution, let x_j be the vector selected from the corresponding X_j, for $j \in \{1, \ldots, t\}$, $C = \{x_1, \ldots, x_t\}$ be the solution cluster and c be the centroid. All vectors in C are identical in all coordinates except at most D, since if there are different values in at least $D + 1$ coordinates, the cost is at least $D + 1$. Denote this subset of coordinates as Q, c could also differ from x_1 only at Q. Denote the subset of coordinates where c differs from x_1 as P, $P \subset Q$ and so $|P| \leq D$. The solution (C, c) induces a subhypergraph H of G in the following way. Leave only hyperedges corresponding to the vectors in C, and restrict them to vertices in P. There are at most D vertices and at most D hyperedges in H, since the total weight is at most D. An example of the correspondence between input vectors and hypergraphs is given in Figure 3.

$D = 2$

v	1	2	3	4	5
x_1	0	2	1	3	2
x_2	0	**1**	1	3	**1**
x_3	**1**	2	1	3	**1**
x_4	0	2	**2**	3	2
x_5	0	2	**2**	3	**1**
c	0	2	**2**	3	2

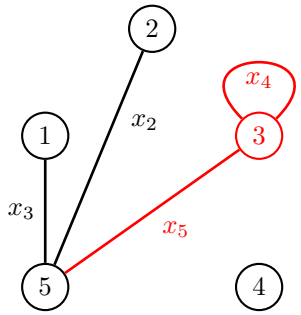

Figure 3 An illustration of the hypergraph construction in Claim 16. On the left, the vector x_1 and other input vectors x_2, …, x_5 are given. On the right, the corresponding hypergraph G. The solution is in red: on the left, the resulting cluster $\{x_1, x_4, x_5\}$ is of cost 2; on the right, the corresponding subhypergraph is H. Note that in H the hyperedge x_5 is restricted to the only vertex 3, so its size is one.

The next claim shows that the fractional cover number of H is bounded by a constant.

▷ Claim 17. Each vertex in H is covered by at least half of the hyperedges of H, and $\rho^*(H) \leq 2$.

Proof. Consider a vertex $p \in P$, and assume that less than half of the hyperedges cover p. It means that in the p-th coordinate the centroid c differs from x_1, but less than half of the vectors in C by weight differ from x_1 in this coordinate. This contradicts Claim 12.

So each vertex is covered by at least half of the hyperedges, and setting $\psi \equiv \frac{2}{|E(H)|}$ leads to $\rho^*(H) \leq 2$. ◁

In order to enumerate all possible subsets of coordinates P, we try all hypergraphs H with at most D vertices and at most D hyperedges, and if each vertex is covered by at least half of the hyperedges, we find all places where H appears in G by Lemma 13. The last step is done in $2^{\mathcal{O}(D \log D)} \cdot (md)^{\mathcal{O}(1)}$ time. However, the number of possible H could be up to $2^{\Omega(D^2)}$. The following claim, which is analogous to Proposition 6.3 in [27], shows that we could consider only hypergraphs with a logarithmic number of hyperedges.

▷ Claim 18. If $D \geq 2$, it is possible to delete all except at most $160 \ln D$ hyperedges from H so that in the resulting hypergraph H^* each vertex is covered by at least $1/4$ of the hyperedges, and $\rho^*(H^*) \leq 4$.

Proof. Denote $s = |E(H)|$, construct a new hypergraph H^* on the same vertex set $V(H)$ by independently selecting each hyperedge of H with probability $(120 \ln D)/s$. Applying Proposition 14 with $\beta = 1/3$, probability of selecting more than $160 \ln D$ hyperedges is at most $\exp((-120 \ln D)/27) < 1/D^2$. By Claim 17, each vertex v of H is covered by at least $s/2$ hyperedges, and the expected number of hyperedges covering v in H^* is at least $60 \ln D$. By Proposition 14 with $\beta = 1/3$, the probability that v is covered by less than $40 \ln D$ hyperedges in H^* is at most $\exp(-60 \ln D/18) \leq 1/D^3$. By the union bound, with probability at least $1 - 1/D^2 - D \cdot 1/D^3 > 0$ we select at most $160 \ln D$ hyperedges and each vertex is covered by at least $40 \ln D$ hyperedges. So the claim holds, and $\rho^*(H^*) \leq 4$ by setting $\psi \equiv \frac{4}{|E(H^*)|}$. ◁

Thus, if there is a subhypergraph H in G corresponding to a solution, then there is also a subhypergraph H^* in G appearing at the same subset of $V(G)$ with at most $160 \ln D$ hyperedges and where each vertex is covered by at least $1/4$ of the hyperedges. Since we only need to enumerate possible coordinate subsets, in our algorithm we try all hypergraphs of this form and apply Lemma 13 for each of them. Since there are at most $2^{\mathcal{O}(D \log D)}$ hypergraphs with at most $160 \ln D$ hyperedges and since the fractional cover number is still bounded by a constant, the total running time is $2^{\mathcal{O}(D \log D)} \cdot (md)^{\mathcal{O}(1)}$, as desired. ◁

With Claim 16 proven, the proof of the theorem is complete. ◀

Combining Theorem 10 and Theorem 15, we obtain an FPT algorithm for k-Clustering. This proves Theorem 1, which we recall here.

▶ **Theorem 1.** k-Clustering *with distance* dist_p *is solvable in time* $2^{\mathcal{O}(D \log D)}(nd)^{\mathcal{O}(1)}$ *for every* $p \in (0, 1]$.

Proof. We have an algorithm for Cluster Selection whose running time is specified by Theorem 15. By Claim 11, the α-property holds. The only missing part is to describe the way of producing the set $\mathcal{D}$ of all possible cluster costs which are at most D.

In the case $p = 1$ all distances are integral so we can take $\mathcal{D} = \{0, \ldots, D\}$.

For the general case, let $\mathcal{B} = \{a^p : a \in \{1, \ldots, \lceil D^{1/p} \rceil\}\}$. Consider a cluster $C = \{x_1, \ldots, x_t\}$ and the corresponding optimal cluster centroid c. For any $x_j \in C$, $\mathrm{dist}_p(x_j, c) = \sum_{i=1}^{d} |x_j[i] - c[i]|^p$ is a combination of elements of $\mathcal{B}$ with nonnegative integer coefficients. This is because x_j and c are integral and the cluster cost is at most D, hence $|x_j[i] - c[i]| \leq D^{1/p}$ for each $i \in \{1, \ldots, d\}$. Since weights are also integral, the whole cluster cost is a combination of distances between cluster vectors and the centroid with nonnegative integer coefficients, and so also a combination of elements of $\mathcal{B}$ with nonnegative integer coefficients. This means that we can take

$$\mathcal{D} = \left\{ \sum_{b \in \mathcal{B}} a_b \cdot b : a_b \in \mathbb{Z}, a_b \geq 0, \sum_{b \in \mathcal{B}} a_b \leq D \right\},$$

the sum of coefficients a_b is at most D since all elements of $\mathcal{B}$ are at least 1. The size of $\mathcal{D}$ is at most $|\mathcal{B}|^D = 2^{\mathcal{O}(D \log D)}$. ◀

Note that another widely studied version of k-Clustering is where centroids c_i could be selected only among the set of given vectors. Naturally, our algorithm also works in this setting since the set of possible centroids is only restricted further.

Also note that Claim 11 and Claim 12 do not hold in the case $p > 1$, and our algorithm relies heavily on the structure provided by them. Therefore, it does not seem that the algorithm could be extended to the case $p > 1$.

5 Conclusion and open problems

In this paper, we presented an FPT algorithm for k-Clustering with $p \in (0, 1]$ parameterized by D. However, for the case $p \in (1, \infty)$ we were able only to show the W[1]-hardness of Cluster Selection. While intractability of Cluster Selection does not exclude that k-Clustering could be FPT with $p \in (1, \infty)$, it indicates that the proof of this (if it is true at all) would require an approach completely different from ours. Thus an interesting and very concrete open question concerns the parameterized complexity of k-Clustering with $p \in (1, \infty)$ and parameter D.

Another open question is about the fine-grained complexity of k-Clustering when parameterized by $k + d$. For several distances, we know XP-algorithms: an $\mathcal{O}(n^{dk+1})$ algorithm by Inaba et. al. [21] for $p = 2$, as well as trivial algorithms for $p \in [0, 1]$. For the case when the possible cluster centroids are given in the input, the matching lower bound is shown in [11]. However, we are not aware of a lower bound complementing the algorithmic results in the case when any point in Euclidean space can serve as a centroid.

References

1. Marcel R. Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and nonmetric distance measures. *ACM Trans. Algorithms*, 6(4):59:1–59:26, 2010. doi:10.1145/1824777.1824779.
2. Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Approximating extent measures of points. *J. ACM*, 51(4):606–635, 2004. doi:10.1145/1008731.1008736.
3. Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, May 2009. doi:10.1007/s10994-009-5103-0.
4. Noga Alon, Raphael Yuster, and Uri Zwick. Color-Coding. *J. ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.
5. D. Angluin and L.G. Valiant. Fast probabilistic algorithms for hamiltonian circuits and matchings. *J. Computer and System Sciences*, 18(2):155–193, 1979. doi:10.1016/0022-0000(79)90045-X.
6. Mihai Badoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 250–257. ACM, 2002. doi:10.1145/509907.509947.
7. Christina Boucher, Christine Lo, and Daniel Lokshantov. Consensus Patterns (Probably) Has no EPTAS. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015*, pages 239–250, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.
8. Christos Boutsidis, Anastasios Zouzias, Michael W. Mahoney, and Petros Drineas. Randomized Dimensionality Reduction for k-Means Clustering. *IEEE Trans. Information Theory*, 61(2):1045–1062, 2015. doi:10.1109/TIT.2014.2375327.
9. Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the 47tg annual ACM symposium on Theory of Computing (STOC)*, pages 163–172. ACM, 2015.

10 Vincent Cohen-Addad. A Fast Approximation Scheme for Low-dimensional K-means. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 430–440. SIAM, 2018. URL: http://dl.acm.org/citation.cfm?id=3174304.3175298.

11 Vincent Cohen-Addad, Arnaud de Mesmay, Eva Rotenberg, and Alan Roytman. The Bane of Low-dimensionality Clustering. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 441–456. SIAM, 2018. URL: http://dl.acm.org/citation.cfm?id=3174304.3175300.

12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

13 W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation Schemes for Clustering Problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 50–58. ACM, 2003. doi:10.1145/780542.780550.

14 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

15 Uriel Feige. NP-hardness of hypercube 2-segmentation. *CoRR*, abs/1411.0821, 2014. arXiv:1411.0821.

16 Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 569–578. ACM, 2011. doi:10.1145/1993636.1993712.

17 Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA and projective clustering. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1434–1453. SIAM, 2013. doi:10.1137/1.9781611973105.

18 Fedor V. Fomin, Petr A. Golovach, and Fahad Panolan. Parameterized Low-Rank Binary Matrix Approximation. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107 of *LIPIcs*, pages 53:1–53:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.53.

19 Fedor V. Fomin, Petr A. Golovach, and Kirill Simonov. Parameterized k-Clustering: The distance matters!, 2019. arXiv:1902.08559.

20 Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 291–300. ACM, 2004.

21 Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th annual Symposium on Computational Geometry (SoCG)*, pages 332–339. ACM, 1994.

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

23 Stavros G. Kolliopoulos and Satish Rao. A Nearly Linear-Time Approximation Scheme for the Euclidean k-Median Problem. *SIAM J. Computing*, 37(3):757–782, June 2007. doi:10.1137/S0097539702404055.

24 Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2):5:1–5:32, 2010. doi:10.1145/1667053.1667054.

25 Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Trans. Information Theory*, 28(2):129–136, 1982. doi:10.1109/TIT.1982.1056489.

26 Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The Planar k-Means Problem is NP-Hard. In *Proceedings of the 3rd International Workshop on Algorithms and Computation (WALCOM)*, Lecture Notes in Comput. Sci., pages 274–285. Springer, 2009. doi:10.1007/978-3-642-00202-1_24.

27 Dániel Marx. Closest Substring Problems with Small Distances. *SIAM J. Comput.*, 38(4):1382–1410, 2008. doi:10.1137/060673898.

28 N. Megiddo and K. Supowit. On the Complexity of Some Common Geometric Location Problems. *SIAM J. Computing*, 13(1):182–196, 1984. doi:10.1137/0213014.

29 Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and Near-Optimal Derandomization. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 182–191. IEEE, 1995.

30 Christian Sohler and David P. Woodruff. Strong Coresets for k-Median and Subspace Approximation: Goodbye Dimension. In *Proceedings of the 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 802–813. IEEE, 2018. doi:10.1109/FOCS.2018.00081.