Coreset for Line Sets Clustering

Anonymous Author(s)
Affiliation
Address
email

Abstract

The input to the line-sets k-median problem is an integer $k \geq 1$, and a set $\mathcal{L} = \{L_1, \ldots, L_n\}$ of n sets of lines in $\mathbb{R}^d$. The goal is to compute a set C of k centers (points in $\mathbb{R}^d$) that minimizes the sum $\sum_{L \in \mathcal{L}} \min_{\ell \in L, c \in C} \operatorname{dist}(\ell, c)$ of Euclidean distances from each set to its closest center, where $\operatorname{dist}(\ell, c) := \min_{x \in \ell} \|x - c\|_2$. An ε-*coreset* for this problem is a weighted subset of sets in $\mathcal{L}$ that approximates this sum up to $1 \pm \varepsilon$ multiplicative factor, for every set C of k centers. We prove that *every* such input set $\mathcal{L}$ has a small ε-coreset, and provide the first coreset construction for this problem and its variants. The coreset consists of $O(\log^2 n)$ weighted line-sets from $\mathcal{L}$, and is constructed in $O(n \log n)$ time for every fixed $d, k \geq 1$ and $\varepsilon \in (0, 1)$. The main technique is based on a novel reduction to a "fair clustering" of colored points to colored centers. We then provide a coreset for this coloring problem, which may be of independent interest. Open source code and experiments are also provided.

1 Introduction

In the classic k-*means* clustering problem, the input is a set P of n points in a metric space $(\mathcal{X}, \operatorname{dist})$, and an integer $k \geq 1$. The goal is to compute a set C^* of k centers (points in $\mathcal{X}$), that minimizes the sum of squared distances over each point $p \in P$ to its nearest center in C^*, i.e., to compute

$$C^* \in \underset{C \subseteq \mathcal{X}, |C| = k}{\arg\min} \sum_{p \in P} \tilde{D}(p, C),$$

where $\tilde{D}(p, C) := \min_{c \in C} \operatorname{dist}^2(p, c)$. This problem is arguably the most common clustering problem formulation, both in industry and academy; see references e.g. in [21, 5, 31, 34].

A natural generalization is to replace this input set P of n points by a set $\mathcal{P}$ of n sets in $\mathcal{X}$. The distance from such an input set $P \in \mathcal{P}$ to the set C^* of centers can then be defined as the distance between the closest point-center pair. This problem is called *k-means for sets;* see e.g. [26] and references therein. Its goal is to compute

$$C^* \in \underset{C \subseteq \mathcal{X}, |C| = k}{\arg\min} \sum_{P \in \mathcal{P}} \tilde{D}(P, C), \tag{1}$$

where $\tilde{D}(P, C) := \min_{p \in P} \tilde{D}(p, C)$.
A special case is where every such set $P \in \mathcal{P}$ is a line ℓ in $\mathcal{X} = \mathbb{R}^d$. This problem is called k-*means for lines*; see e.g. [18, 33, 12]. Its goal is to compute a set C^* of k points in $\mathbb{R}^d$ such that

$$C^* \in \underset{C \subseteq \mathbb{R}^d, |C| = k}{\arg\min} \sum_{\ell \in \mathcal{P}} \tilde{D}(\ell, C). \tag{2}$$

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

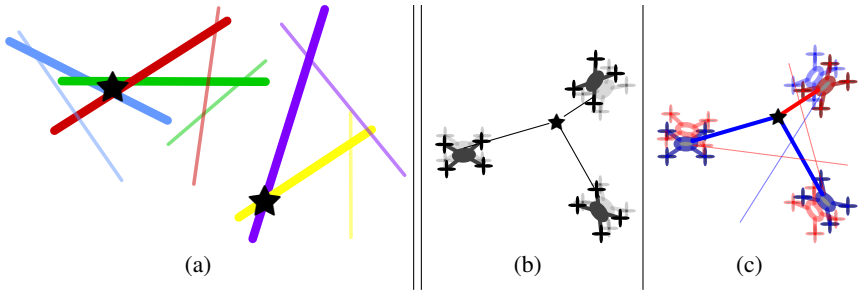


Figure 1: ***Line sets Clustering (left).*** *The $k = 2$ line-sets means (black stars) of $n = 5$ sets (in different colors), each contains a pair ($m = 2$) of lines (same color) in the plane ($\mathcal{X} = \mathbb{R}^2$). The closest line to the stars in each pair of lines is bold.* ***Application (right).*** *(b) A set of $n = 3$ drones, each equipped with an on-board 2D camera, and takes a snapshot of a point in $\mathbb{R}^3$ (the black $k = 1$ star). Due to the missing depth, each such point corresponds to a pixel in the 2D image's plane, or a line in $\mathbb{R}^3$ which is the intersection of the image's plane with the camera's pin-hole. (c) The estimated position of the drone from the noisy cameras is inaccurate, so we use $m = 2$ guesses or consecutive images over time (blue and red line). One of them (in bold) is closer to the observed point.*

This paper considers a further generalization of the last problem, where the input is a set of n *sets*, each contains *multiple* lines as follows.

Line-sets clustering. The input to the line-sets k-mean problem is an integer $k \geq 1$, and a set $\mathcal{L} = \{L_1, \ldots, L_n\}$ of n sets, where each set $L = \{\ell_1, \ldots, \ell_m\}$ in $\mathcal{L}$ consists of $m \geq 1$ lines in $\mathbb{R}^d$. The goal is to compute a set C^* of k points in $\mathbb{R}^d$ that minimizes its sum of squared distances over every set of lines in $\mathcal{L}$, i.e.,

$$C^* \in \underset{C \subseteq \mathbb{R}^d, |C|=k}{\arg\min} \sum_{L \in \mathcal{L}} \tilde{D}(L, C). \tag{3}$$

Here, $\tilde{D}(L, C) := \min_{\ell \in L} \tilde{D}(\ell, C)$ is the shortest squared distance between a point in C to its closest line in L; see Fig 1.

1.1 Motivation

The classic k-clustering of points has numerous applications, and the k line-sets clustering is its natural generalization. It is thus not surprising that these applications can be easily generalized and extended. We give examples from different research fields as follows.

General applications. A natural application is clustering of a data with missing values, as described e.g. in [12]. Here, each record in the input data-set is represented by a point in d dimensions. If an entry is missing, we may replace it by all possible values, resulting in a line in $\mathbb{R}^d$ that is parallel to one of the axes. Similarly, if the missing entry is discrete, i.e., can be one in a finite set of options, it results in a set of points. Our paper handles combination of them both. An example of this can be found in Dataset 4 in our experimental results for California housing. The entries may be the number of the apartment or floor, which corresponds to a set of values. A monthly rent is a continuous variable, which corresponds to a line. Together they correspond to line-sets. Another perspective would be of sets clustering with missing values, where most of the applications in [26] may be generalized using our work.

Computer vision. In the fundamental 3D model reconstruction problem, we are given n pixels (features). Each pixel is from a different 2D image that captures the same point in $\mathbb{R}^3$, i.e., the real world. Such a 2D pixel corresponds to a line in $\mathbb{R}^3$ that passes through the camera's pinhole and the pixel in the image plane. The goal is to estimate the location of the point in $\mathbb{R}^3$, based on the set of n 2D images. The task of estimating most probable 3D point location given a single set of lines is called *Triangulation*. Due to uncertainty and noise, feature extraction algorithms usually identify $m \geq 1$ pixels in each image as the captured point, without certainty about which of the pixels are noise and which are correct. Here, the center is this desired point in $\mathbb{R}^3$, and every image is represented by m lines. To obtain a model or a "3D point-cloud" [19, 23] we may wish to compute k points (centers) of the 3D object. See Fig. 1 (right).

We hope that this paper will open the research toward more complex and general shapes that may yield better approximations to high-ways and polygons such as k segments (as in [36, 25, 12]), arcs or k-subspaces instead of lines.

Colored point-sets clustering applications As a separate technical result we present a coreset for clustering of sets of colored points, which has its own applications. An example is the facility location problem, as in the home and work example in Section 3. Another application is text documents clustering, shown in our experiments. Here, every document is represented as a set of colored vectors, where each vector corresponds to a paragraph in a "bag of words" representation. Each color represents a different type of paragraph, e.g., introduction, summary, main section, theorem, etc. More generally, the problem is relevant where each facility is useful for a specific subset of input clients.

1.2 Generalizations and computation models

Our main technique to handle the problems above, and especially their streaming, distributed, and parallel versions, is to design data summaries (often called coresets) for the k line-sets clustering problem, and its corresponding set of queries. We may then run naive algorithms such as exhaustive search on the small coreset to extract the approximated solution, or apply existing heuristics and obtain faster results.

Definition 1.1 (Coreset)**.** *Let $k \geq 1$ be an integer, and $\mathcal{L} = \{L_1, \ldots, L_n\}$ be an input set, where each $L = \{\ell_1, \ldots, \ell_m\} \in \mathcal{L}$ consists of $m \geq 1$ lines in $\mathbb{R}^d$. For a given approximation error parameter $\varepsilon \in (0, 1)$, an* ε-coreset of $\mathcal{L}$ for k line-sets clustering *over a pseudo distance $\tilde{D}$ is a weighted set $\mathcal{S} \subseteq \mathcal{L}$ that approximates the sum of pseudo distances to $\mathcal{L}$ from* every *set C of k centers in $\mathbb{R}^d$, up to a multiplicative factor of $1 \pm \varepsilon$. More precisely, there is a* weight function $v : \mathcal{S} \to [0, \infty)$ *such that for every set $C \subseteq \mathbb{R}^d$ of size $|C| \leq k$,*

$$\left| \sum_{L \in \mathcal{L}} \tilde{D}(L, C) - \sum_{L \in \mathcal{S}} v(L) \tilde{D}(L, C) \right| \leq \varepsilon \sum_{L \in \mathcal{L}} \tilde{D}(L, C).$$

In particular, the (optimal) line-sets k-mean C^* of the weighted set $(\mathcal{S}, v)$ is a $(1 + \varepsilon)$-approximation to the k-line-sets mean of $\mathcal{L}$. This is the strongest type of coresets (sometimes called "strong coreset" [32, 39]), in the sense that it approximates *every* query C, and not just, say, the optimal solution. It is also a weighted subset of the input, unlike a sketch, linear combination, or general subsets of lines in $\mathbb{R}^d$ which are not a subset of $\mathcal{L}$; see next Section 1.3.

1.3 Related work

Clustering n points by k center points is a classic fundamental problem in machine learning [38]. Applications can be found in operations research [9, 7], statistics [21] and computational geometry [28, 4, 41], including constant factor approximations (randomized [2] and deterministic [8]). Many of the recent results[22, 14] are based on coresets (under different definitions) that can usually be computed in time that is near linear in the input size and number k of centers. The size of these coresets is usually $(k/\varepsilon)^{O(1)}$, where $\varepsilon \in (0, 1)$ is a parameter that represents the approximation error.

Coreset for clustering n lines as in Eq. (2) that contains a weighted subsets of input lines was suggested in [33]. For the case that the lines are parallel to the axes, coresets were suggested in [12]. These results also hold for j-dimensional linear subspaces, but only if they are parallel to the axes of $\mathbb{R}^d$. The main motivation is handling j missing entries in each database's record (point).

Coresets for input sets of points, lines, and fair clustering appeared relatively recently. Coreset for (point)-sets clustering as in Eq. (1), where each set consists of m points, was suggested in [26] and its size is $\left(\frac{\log n}{\varepsilon}\right)^2 k^{O(m)}$.

Our main technical result is a coreset for colored (points) clustering, where the sets and centers are colored, and each input point can be served by a center of the same color. This approach is strongly related to fair clustering, where each group is represented by a color.

Variations of fair clustering were suggested in [27, 24, 13] and references therein. Coresets for fair clustering were suggested in [37]. This paper generalizes some of these results in the sense that the input points are colored sets, but the centers are also colored.

Our coreset constructions, as many of the recent coreset constructions, is based on the Feldman-Langberg framework [29, 16, 10, 11]. This framework reduces the problem of computing a coreset – to the problem of computing the importance (sensitivity) for each of the n input points/sets.

1.4 Main contributions

We suggest the first coreset construction for line-sets clustering whose size (number of weighted sets) is sub-linear in the number n of input sets. This is by a reduction to another problem, fair clustering (colored points) with colored centers, which may be of independent interest. We then suggest a coreset for this type of fair clustering problems. More precisely, we provide the following contributions for every pseudo distance $\tilde{D}$, constant approximation error $\varepsilon \in (0,1)$, and constant integers $d, k \geq 1$. See the corresponding theorems for exact bounds.

(i) A proof that an ε-coreset $\mathcal{S} \subseteq \mathcal{L}$ of size $|\mathcal{S}| = O(\log^2 n)$ exists for *every* input set $\mathcal{L}$ of n m-sets of lines in $\mathbb{R}^d$. The coreset $\mathcal{S}$ can be computed, with high probability, in time $O(n \log n)$; see Theorem 3.6.

(ii) A proof that an ε-coreset $\mathcal{S}$ of size $|\mathcal{S}| = O(\log^2 n)$ exists for *every* input set $\mathcal{P}$ of n colored-sets in a metric space, each of size m. The coreset $\mathcal{S}$ can be computed with high probability, in time $O(n \log n)$; see Theorem 3.5.

The following results are straightforward implication of the above results, using existing merge-reduce frameworks [1]:

(iii) Support for streaming data in insertion time and memory that is poly-logarithmic in the number n of the sets seen so far in the stream.

(iv) Support for parallel computations on data that may be distributed on M machines using $1/M$ amortized insertion time.

(v) Support for deletion of sets from the stream in poly-logarithmic time during the streaming, but using memory that is linear with n.

(vi) FPTAS in time $O(n \log n)$, by running an exhaustive search on the corresponding small corsets.

Dependencies on input parameters. Our first result above is in fact an FPTAS or LTAS (linear time approximation scheme), since the asymptotic running time depends polynomially on $1/\varepsilon$, and near-linear in n. The dependency on d is polynomial, and the exponential dependency on k and m is unavoidable due to known lower bounds for special cases e.g. in [20, 26]. These are worst-case bounds, and the actual approximations errors are much smaller in real life and shown in the experimental results section bellow; see Section 4.

1.5 Novel technique: from line-sets to fair clustering

Existing coresets for point-sets clustering are heavily based on the triangle inequality between pair of points, which is not satisfied for the case of lines. On the other side, existing coresets for lines were reduced to coresets for weighted centers, which do not support sets. The main challenge in this paper is to combine these two results. This is by suggesting a novel reduction and coresets for colored-weighted centers (that includes independent interest) which borrow techniques from fair clustering. To our knowledge, this is the first link of this type between geometric shapes and fairness. We hope it will open the door for many other coresets; see Future Work.

Our reduction is based on the following three steps.

Grouping. Algorithm 3 in Section 3.2 recursively chooses a very "dense" constant fraction of sets $\mathcal{L}^m$ that are close to m center points (robust medians) $\mathcal{B}^m = \{b_1, \ldots, b_m\}$; see Definition 2.4. The ith line in each of these sets is then translated to its median b_i, for every $i \in [m]$. At this stage we reduced our problem to compute sensitivity for $|\mathcal{L}^m| \in O(n)$ sets of lines that form m "stars"; see Fig. 4(a).

Reduction to points. The distance from each line that intersects a robust median b_i (say, the origin) to a query center $c_i \in C$ is its distance to a unit vector in the same direction, weighted by the norm of the vector; see Fig. 4(b). This distance can be approximated, up to a constant factor, by replacing c_i with its projection on the sphere $c_i / \|c_i\|$ and its antipodal point $-c_i / \|c_i\|$.

Colored sets and centers. The new problem is now to compute sensitivity for n sets, each contains m *colored points* on m unit spheres; see Fig. 4(c). Each center in the set C of k queries is duplicated m times in m colors. The result is a set of $2mk$ weighted centers on m unit sphere with an additional constraint: the ith center can "serve" only the lines on the ith sphere.

2 Preliminaries

The results in this paper hold not only for squared distance functions but for other functions such as non-squared distances or M-estimators. To generalize this notion, we use the following pseudo distance over a metric space $(\mathcal{X}, \mathrm{dist})$; see many examples of such functions in [26]. From Section 3.2, i.e., for the case of input lines, we assume that $\mathcal{X} = \mathbb{R}^d$ and dist is the Euclidean norm. That is, the metric space is $(\mathbb{R}^d, ||.||_2)$ but lip below can still be any function that satisfies the following log-log Lipschitz condition.

Definition 2.1 (Pseudo-distance $\tilde{D}$)**.** *Let* $\mathrm{lip} : [0, \infty) \to [0, \infty)$ *be a* non-decreasing function *that satisfies the following condition: There is a constant* $r \in (0, \infty)$ *such that for every* $x, z > 0$ *we have* $\mathrm{lip}(zx) \leq z^r \mathrm{lip}(x)$*. Let* $(\mathcal{X}, \mathrm{dist})$ *be a metric space, and* $\tilde{D}$ *be a function that maps every pair of points* $p, c \in \mathcal{X}$ *to*

$$\tilde{D}(p, c) := \mathrm{lip}(\mathrm{dist}(p, c)).$$

For a pair of finite sets $P, C \subseteq \mathcal{X}$*, denote* $\tilde{D}(P, C) := \min_{p \in P, c \in C} \tilde{D}(p, c)$*,* $\tilde{D}(P, c) := \tilde{D}(P, \{c\})$ *and* $\tilde{D}(p, C) = \tilde{D}(\{p\}, C)$*.*

The motivation behind the last definition is that it satisfies the following pair of properties.

Lemma 2.2 (weak triangle [40])**.** *Let* $\mathcal{X}$*,* $\tilde{D}$*, and* r *be as defined in Definition 2.1. Then* $\tilde{D}$ *satisfies the following ("weak triangle") inequalities (i)–(ii) for every* $p, q, c \in \mathcal{X}$*:*

(i) For $\rho = \max\{2^{r-1}, 1\}$*,* $\quad \tilde{D}(p, q) \leq \rho(\tilde{D}(p, c) + \tilde{D}(c, q))$*.*

(ii) For $\phi = (4r - 4)^{r-1}$*,* $\quad \tilde{D}(p, c) - \tilde{D}(q, c) \leq \phi \tilde{D}(p, q) + \frac{\tilde{D}(p,c)}{4}$*.*

In the rest of the paper, for an integer $n \geq 1$ we denote $[n] = \{1, \ldots, n\}$. Also, unless otherwise stated $\tilde{D}, \mathcal{X}, r$ are as stated in Definition 2.1.

For both the line-sets clustering and the colored-sets clustering problems/coresets, the input is a set of n sets, each of size m. We call it (n, m)-set for short as follows.

Definition 2.3 ((n, m)-set)**.** *For a given integer* $m \geq 1$*, an* m-set P *is a set of* m *items, i.e.* $|P| = m$*. For an additional integer* $n \geq 1$*, an* (n, m)-set *is a set of* n m*-sets.*

Informally, a robust median for an optimization problem at hand is an element b that approximates the optimal value of this optimization problem, with some leeway on the number of input elements considered. In the context of facility location, the facility (center) needs to serve only a subset of the closest clients (input points).

Let $\mathcal{P}$ be an (n, m)-set, $C \subseteq \mathcal{X}$ and $\gamma \in (0, 1]$. We denote by $\mathrm{closest}(\mathcal{P}, C, \gamma)$ the set that is the union of $\lceil \gamma |\mathcal{P}| \rceil$ sets $P \in \mathcal{P}$ with the smallest value of $\tilde{D}(P, C)$, i.e.,

$$\mathrm{closest}(\mathcal{P}, C, \gamma) \in \underset{\mathcal{Q} \subseteq \mathcal{P} : |\mathcal{Q}| = \lceil \gamma |\mathcal{P}| \rceil}{\arg\min} \sum_{P \in \mathcal{Q}} \tilde{D}(P, C). \tag{4}$$

For simplicity of notation, we define $\mathrm{closest}(\mathcal{P}, C) := \mathrm{closest}(\mathcal{P}, C, \frac{1}{|\mathcal{P}|})$ as one of the closest items to C in $\mathcal{P}$. Here and in the rest of the paper ties are broken arbitrarily.

Definition 2.4 (Robust median [17])**.** *Let* $\mathcal{P}$ *be an* (n, m)*-set in* $\mathcal{X}$*,* $\gamma \in (0, 1]$*, and*

$$\tilde{D}^*(\mathcal{P}, \gamma) = \min_{b \in \mathcal{X}} \sum_{P \in \mathrm{closest}(\mathcal{P}, \{b\}, \gamma)} \tilde{D}(P, b).$$

For $\tau \in (0, 1)$ *and* $\alpha \geq 0$*, a point* $b \in \mathcal{X}$ *is a* (γ, τ, α)-median *for* $\mathcal{P}$ *if*

$$\sum_{P \in \mathrm{closest}(\mathcal{P}, \{b\}, (1-\tau)\gamma)} \tilde{D}(P, b) \leq \alpha \cdot \tilde{D}^*(\mathcal{P}, \gamma).$$

Definition 2.5 (Translation)**.** *Let* b *be a point in* $\mathcal{X}$*.* (i) *For every* $p \in \mathcal{X}$*, we define* $T(p, b) := b$*.* (ii) *If* $\mathcal{X} = \mathbb{R}^d$*, for every line* ℓ *in* $\mathbb{R}^d$ *we define* $T(\ell, b)$ *to be the line that is parallel to* ℓ *and intersects* b*.*

In what follows, we define the projection of a set (points or lines) over a set of points; see Fig. 2.

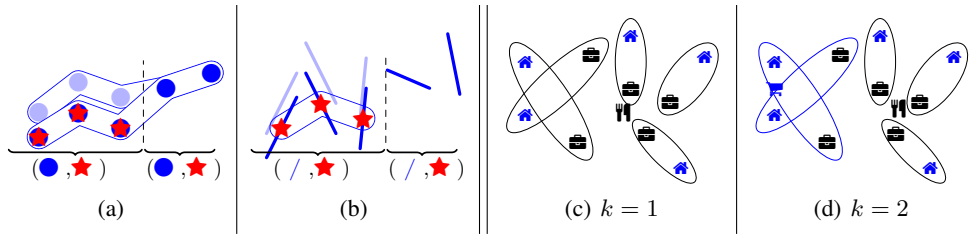


Figure 2: ***Example sets projections for Definition 2.6 (left).*** *The projection of $P =$ (top 5 points) onto the ordered set $\mathcal{B} =$ (red stars) yields $T(P, \mathcal{B}) =$ (5 dark blue points), i.e., the union of 3 points $\mathrm{proj}(P, \mathcal{B})$ (3 on the stars) with the 2 remaining points $\overline{\mathrm{proj}}(P, \mathcal{B})$.* **(b)** *The projection of $L =$ / (top 5 lines) onto the ordered set $\mathcal{B}$ (red stars) yields $T(L, \mathcal{B})$ (5 dark blue lines), i.e., the union of 3 lines $\mathrm{proj}(P, \mathcal{B})$ (translated to to the stars) with the 2 remaining lines $\overline{\mathrm{proj}}(L, \mathcal{B})$.* **Colored-set k-clustering (right).** *The input is a set of $n = 5$ pairs of points that correspond to home/work addresses (in blue and black).* **(c)** *$k = 1$ center, which is a restaurant (black fork and knife). Its distance to a person is the distance to her office (black suitcase).* **(d)** *$k = 2$ centers: a restaurant (black fork and knife) and a grocery shop (blue cart). Each person chooses the closest restaurant to her office or the closest grocery to her home.*

Definition 2.6 (Set projection)**.** *Let m, j be a pair of integers such that $1 \leq j \leq m$. Let P be either an m-set of points in $\mathcal{X}$ or an m-set of lines in $\mathcal{X} = \mathbb{R}^d$. Let $\mathcal{B} = (b_1, \ldots, b_j)$ be an ordered j-set of points in $\mathcal{X}$. Let $p_1 \in P$ denote the closest item $\{p_1\} = \mathrm{closest}(P, \{b_1\})$ to b_1. For every integer $i \in \{2, \ldots, j\}$, recursively define p_i to be the closest item $\{p_i\} = \mathrm{closest}(P \setminus \{p_1, \ldots, p_{i-1}\}, \{b_i\})$ in $P \setminus \{p_1, \ldots, p_{i-1}\}$ to b_i. We denote,*

(i) $\mathrm{proj}(P, \mathcal{B}) := \{\mathrm{T}(p_1, b_1), \ldots, \mathrm{T}(p_j, b_j)\}$ *as the j items in P that were projected onto $\mathcal{B}$.*

(ii) $\overline{\mathrm{proj}}(P, \mathcal{B}) := P \setminus \{p_1, \ldots, p_j\}$ *as the $m - j$ items in P that were not projected onto $\mathcal{B}$.*

(iii) $\mathrm{T}(P, \mathcal{B}) := \mathrm{proj}(P, \mathcal{B}) \cup \overline{\mathrm{proj}}(P, \mathcal{B})$ *as the* projection *of P onto $\mathcal{B}$.*

(iv) For projection over an empty set, we define $\overline{\mathrm{proj}}(P, \emptyset) := \mathrm{T}(P, \emptyset) := P$.

3 Sensitivity

Our main coreset construction (Algorithm 5) is a standard non-uniform sampling algorithm based on Langberg-Feldman framework [15]. The main challenge in this framework is to compute the importance of each input set, known as sensitivity. Our main contribution is an algorithm that computes this sensitivity with provable guarantees on its running time and the total sensitivity, which implies the size of the resulting coreset.

As explained in the novelty section, we provide coreset construction for line-sets, by bounding sensitivity of a constant fraction of the input lines recursively. Section 3.2 provides sensitivity bound for lines-sets clustering, by translating the input lines into robust medians, converting the translated lines into colored points, and then compute sensitives for colored-sets as in Section 3.1.

Definition 3.1 (Line-set Sensitivity)**.** *Let $\mathcal{L}$ be an (n, m)-set of lines in $\mathbb{R}^d$, and let $k \geq 1$ be an integer. We define the* sensitivity *of every such set $L \in \mathcal{L}$ of lines to be*

$$S_{\mathcal{L},k}(L) = \sup_{C \subseteq \mathbb{R}^d, |C|=k} \frac{\tilde{D}(L, C)}{\sum_{L' \in \mathcal{L}} \tilde{D}(L', C)},$$

where the supremum is over every set C of k points in $\mathbb{R}^d$ such that $\tilde{D}(L, C) > 0$.

3.1 Colored sets

In this subsection, we bound the sensitivity for the *colored sets clustering*. Here, the input is a set $\mathcal{P}$ of n m-ordered-sets (the order resembles the colors of the points) in a metric space $(\mathcal{X}, d)$ equipped with a distance function $\tilde{D}$, and an integer $k \geq 1$. The goal is to compute a colored set $\subseteq \mathcal{X} \times [m]$ of k pairs, each consisting of a point and a index (color) in $[m]$ that minimizes the sum of distances to the sets. Here, the distance to each ordered-set $P \in \mathcal{P}$ is the minimum distance from a center in to a point in P which has the same color,

$$\in \operatorname*{arg\,min}_{C' \subseteq \mathcal{X} \times [m], |C'|=k} \sum_{(p_1, \ldots, p_m) \in \mathcal{P}} \min_{(c,t) \in C'} D(c, p_t).$$

Example. Suppose we want to provide a convenient food source for n workers whose home and work addresses are represented by n pairs of GPS coordinates (points on the plane). Each person can either buy ingredients at the grocery and make a lunch box at home, or go to a restaurant during lunchtime; see Fig 2.

In order to make the reduction to sets of lines, we need coreset for this problem that supports weights. Hence, we define the weighted version of this problem.

Definition 3.2 ((n,m)-ordered-set)**.** *An m-ordered set P is an ordered-set of $m \geq 1$ points in $\mathcal{X}$. An (n,m)-ordered-set is a set of $n \geq 1$ m-ordered-sets.*

Definition 3.3 (weighted colored center)**.** *A weighted colored center in $\mathcal{X}$ is a triplet (c,w,t), where $c \in \mathcal{X}$, $w \geq 0$, and $t \geq 1$ is an integer. We define the weighted distance from an m-ordered set $P = (p_1, \ldots, p_m)$ to a colored weighted center $c' = (c,w,t)$ to be*

$$\tilde{D}(P,c') = \tilde{D}(P,(c,w,t)) = \begin{cases} w \cdot \tilde{D}(p_t, c), & \text{if } t \leq m \\ \infty, & \text{otherwise} \end{cases}.$$

The distance between a finite set C of weighted colored centers, and an m-ordered-set P is defined by $\tilde{D}(P,C) = \min_{c' \in C} \tilde{D}(P,c')$, and the cost between C and an (n,m)-ordered set $\mathcal{P}$ is the sum $\tilde{D}(\mathcal{P},C) = \sum_{P \in \mathcal{P}} \tilde{D}(P,C)$.

Overview of Algorithm 1. Given a set $\mathcal{P}$ of m-ordered-sets and an integer $k \geq 1$, Algorithm 1 computes a set $\mathcal{B}_k^m$ of center points (used only for the analyses) and a set $\mathcal{P}_k^m \subseteq \mathcal{P}$ of "similar" m-ordered-sets, which are approximately equally important for the problem at hand; see Lemma 3.4. On the ℓth iteration of the external loop (Line 2), the algorithm computes a fraction of $O(\frac{1}{k^m})$ m-sets from $\mathcal{P}_{\ell-1}^m$ that are similar in the sense that there are m dense balls of small radii, each contains at least one point from each set.

Algorithm 1: CS-DENSE$(\mathcal{P},k)$

Input : An (n,m)-ordered-set $\mathcal{P}$ in $\mathcal{X}$, and an integer $k \geq 1$.
Output : A pair $(\mathcal{P}_k^m, \mathcal{B}_k^m)$, where $\mathcal{P}_k^m \subseteq \mathcal{P}$ and $\mathcal{B}_k^m \subseteq \mathcal{X} \times [m]$; see Lemma 3.4.

1 $\mathcal{P}_1^0 := \mathcal{P}$; $\mathcal{B}_1^0 := \emptyset$; $\tau := \dfrac{1}{20}$
2 **for** $r := 1$ **to** k **do**
3 **for** $\ell := 1$ **to** m **do**
4 $\overline{\mathcal{P}}_r^{\ell-1} := \left\{ \overline{\text{proj}}(P, \mathcal{B}_r^{\ell-1}) \middle| P \in \mathcal{P}_r^{\ell-1} \right\}$ `// see Definition 2.6`
5 Compute a $\left(\dfrac{1}{2k}, \tau, 2\right)$-median $b_r^\ell \in \mathcal{X} \times [m-(\ell-1)]$ for $\overline{\mathcal{P}}_r^{\ell-1}$
249 `// see Definition 2.4, and suggested implementation in Algorithm 6.`
6 $\mathcal{P}_r^\ell := \left\{ P \in \mathcal{P}_r^{\ell-1} \;\middle|\; \overline{\text{proj}}(P, \mathcal{B}_r^{\ell-1}) \in \text{closest}\left(\overline{\mathcal{P}}_r^{\ell-1}, \{b_r^\ell\}, \frac{1-\tau}{4k}\right) \right\}$
 `// $\mathcal{P}_r^\ell$ contains every $m$-set $P$ such that $\overline{\text{proj}}(P, \mathcal{B}_r^{\ell-1})$ is in the fraction of the closest $(1-\tau)/(4k)$ sets in $\overline{\mathcal{P}}^{\ell-1}$ to the center $b_r^\ell$; see Definition 2.6.`
7 $\mathcal{B}_r^\ell := \mathcal{B}_r^{\ell-1} \bigcup \{b_r^\ell\}$
8 $\mathcal{P}_{r+1}^0 := \mathcal{P}_r^m$; $\mathcal{B}_{r+1}^0 := \mathcal{B}_r^m$
9 **Return** $(\mathcal{P}_k^m, \mathcal{B}_k^m)$

Lemma 3.4 (sensitivity of colored-sets)**.** *Let $\mathcal{P}$ be an (n,m)-ordered-set in $\mathcal{X}$, $k \geq 1$ be an integer, and let $(\mathcal{P}_k^m, \mathcal{B}_k^m)$ be the output of a call to* CS-DENSE$(\mathcal{P},k)$*; see Algorithm 1. Then $|\mathcal{P}_k^m| \in \Theta(n)$, and for every set $P \in \mathcal{P}_k^m$ and a set $C \subseteq \mathcal{X} \times [0,\infty) \times [m]$ of $|C| = k$ weighted colored centers such that $\tilde{D}(P,C) > 0$, we have $\frac{\tilde{D}(P,C)}{\sum_{P' \in \mathcal{P}} \tilde{D}(P',C)} \leq \frac{2k}{|\mathcal{P}_k^m|}$.*

Theorem 3.5. *Let $k, m \geq 1$ be constant integers, and let $\mathcal{P}$ be an (n,m)-ordered-set in $\mathcal{X}$, $\varepsilon, \delta \in (0,1)$, and let $\eta \geq \left(\frac{m^{1.5} d \log n}{\varepsilon}\right)^2 (2k)^{cmk} + \log\left(\frac{1}{\delta}\right)$ be an integer, where c is a sufficiently large constant that can be determined from the proof. There is an algorithm that given $\mathcal{P}, k, \varepsilon$ and δ return with probability at least $1-\delta$ an ε-coreset of $\mathcal{P}$ for colored-sets k-mean of size η in $n \log(n)(2k)^{O(mk)}$ time.*

3.2 Line-sets

Overview of Algorithm 2. Given an (m,m)-set $\mathcal{L}$ of lines, and a set $B \subseteq \mathbb{R}^d$ that are both satisfy the condition of Lemma C.2, and an integer $k \geq 1$. Algorithm 2 outputs a function $s : \mathcal{L} \to [0,\infty)$ that maps every set $L \in \mathcal{L}$ of lines to an upper bound $s(L)$ of its sensitivity.

Algorithm 2: GROUPED-SENSITIVITY$(\mathcal{L}, B, k)$

Input : An integer $k \geq 1$, an (n,m)-set $\mathcal{L}$ of lines, and a set $B = (b_1, \ldots, b_m)$ of m points both in $\mathbb{R}^d$ such that for every set $L = (\ell_1, \ldots, \ell_m) \in \mathcal{L}$ and every $i \in [m]$, the line ℓ_i intersects b_i.

Output : A sensitivity function $s : \mathcal{L} \to [0,\infty)$, for every set $L \in \mathcal{L}$.

For every $i \in [m]$, set $\mathbb{S}_i := \{x \in \mathbb{R}^d \mid \|x - b_i\| = 1\}$

For every $L \in \mathcal{L}$, set $P(L) := (\ell_i \cap S_i)_{i=1}^m$ `// A 2m-ordered set`

$\mathcal{P} := \{P(L) \mid L \in \mathcal{L}\}$ `// An (n,2m)-ordered set`

while $|\mathcal{P}| > 2mk$ **do**

  $(\mathcal{P}_k^m, \mathcal{B}_k^m) :=$ CS-DENSE$(\mathcal{P}, mk)$ `// see Algorithm 1`

  for *every* $P \in \mathcal{P}^m$ **do**

    $s'(P) := \dfrac{2mk}{|\mathcal{P}_k^m|}$ `// see Lemma 3.4`

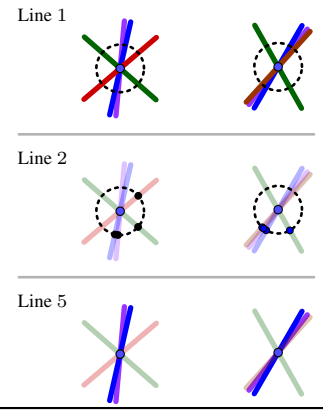
  $\mathcal{P} := \mathcal{P} \backslash \mathcal{P}_k^m$

for *every* $Q \in \mathcal{P}$ **do**

  $s'(Q) := 1$

for every $L \in \mathcal{L}$ **do** $s(L) := \sqrt{2}s'(P(L))$.

Return s



Overview of Algorithm 3 Given an (n,m)-set $\mathcal{L}$ of lines, and an integer $k \geq 1$, Algorithm 3 outputs a pair of sets $(\mathcal{L}^{m+1}, \mathcal{B}^m)$. All the m-sets in $\mathcal{L}^{m+1}$ are similar in the sense that their sensitivity with respect to original problem is small and equal.

Algorithm 3: LS-DENSE$(\mathcal{L}, k)$

Input : An (n,m)-set $\mathcal{L}$ and an integer $k \geq 1$.

Output : A pair $(\mathcal{L}^m, \mathcal{B}^m)$, where $\mathcal{L}^m \subseteq \mathcal{L}$ and $\mathcal{B}^m \subseteq \mathbb{R}^d$ is an ordered set.

$\tau := \frac{1}{20}$

$\bar{\mathcal{L}}^0 := \mathcal{L}^0 := \mathcal{L}$; $\mathcal{B}^0 := \emptyset$

for $i := 1$ **to** m **do**

  Compute a $\left(\dfrac{1}{2k}, \tau, 4\right)$-median $b^i \in \mathbb{R}^d$ for $\bar{\mathcal{L}}^{i-1}$

  `// see Algorithm 6 for suggested implementation`

  $\mathcal{L}^i := \left\{ L \in \mathcal{L}^{i-1} \;\middle|\; \overline{\mathrm{proj}}(L, \mathcal{B}^{i-1}) \in \mathrm{closest}\left(\bar{\mathcal{L}}^{i-1}, \{b^i\}, \frac{1-\tau}{4k}\right) \right\}$

  $\mathcal{B}^i := \mathcal{B}^{i-1} \bigcup \{b^i\}$

  $\bar{\mathcal{L}}^{i-1} := \{\overline{\mathrm{proj}}(L, \mathcal{B}^{i-1}) \mid L \in \mathcal{L}^{i-1}\}$ `// see Fig 2`

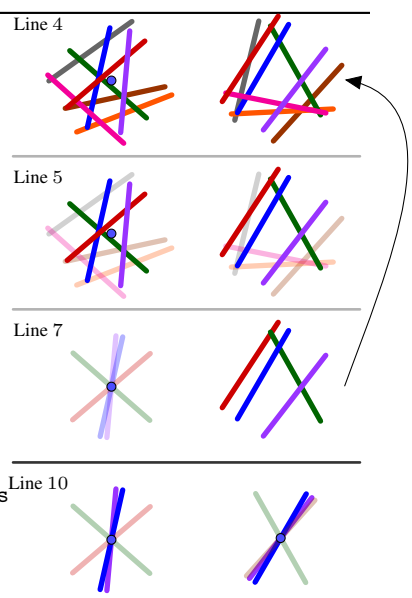
$\mathcal{L}' := \{\mathrm{proj}(L, \mathcal{B}^m) \mid L \in \mathcal{L}^m\}$

$s :=$ GROUPED-SENSITIVITY$(\mathcal{L}', \mathcal{B}^m, k)$ `// see Algorithm 2`

$\mathcal{L}^{m+1} := \arg\min_{L \in \mathcal{L}^m} s\Big(\mathrm{proj}(L, \mathcal{B}^m)\Big)$ `// L^{m+1} is a set of sets`

`// biggest cluster whose sets have equal sensitivity`

Return $(\mathcal{L}^{m+1}, \mathcal{B}^m)$



Theorem 3.6. *Let $\mathcal{L}$ be an (n,m)-set of lines in $\mathbb{R}^d$, $k \geq 1$ be an integer, $\varepsilon, \delta \in (0,1)$, and let $\eta \geq \left(\frac{m^{1.5} d \log n}{\varepsilon}\right)^2 (2k)^{cmk} + \log\left(\frac{1}{\delta}\right)$ be an integer, where c is a sufficiently large constant that can be determined from the proof. Let $(\mathcal{C}, v)$ be the output of a call to* CORESET$(\mathcal{L}, k, \eta)$*; see Algorithm 5 in the Appendix. Then, Claims (i)–(ii) hold as follows.*

(i) With probability at least $1 - \delta$, $(\mathcal{C}, v)$ is an ε-coreset of $\mathcal{L}$ for the k-line-sets of size $|\mathcal{C}| = \eta$.

(ii) The pair $(\mathcal{C}, v)$ can be computed in $n \log(n)(2k)^{O(mk)}$ time.

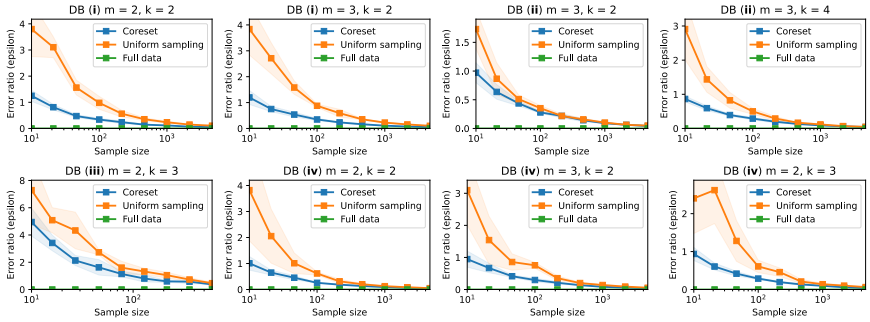


Figure 3: **Experimental results.** *See details in Section 4.*

4 Experimental Results

We implemented our coreset constriction algorithms for both Colored-sets and Line-sets. In this section we test there empirical performance on synthetic and real data. Open source is available in [6]. Since we provide the first coreset construction for the given problems, we compare our results to a single baseline - a random uniform sampling of the same number of points as in the coreset.

Colored-sets data-sets

(i) A *synthetic dataset* of size $n = 10000$ with outliers by sampling colored points from a set of Gaussians with different parameters corresponding to different colors. For each color there are two overlapping Gaussians for modeling the inliers of size 4950 each and one Gaussian more than three standard deviations away of size 100 for modeling the outliers.

(ii) The Reuters-21578 data-set from [30], which results in sets of points corresponding to each paragraph in each document. Additionally we divide paragraphs into categories: beginning (first third), main part (middle third) ending (last third). This results in each document being represented as a set of m=3 colored points corresponding to sets of paragraphs of different categories.

Line-sets data-sets

(iii) A *synthetic dataset* that consists of the colored points dataset from the previous section, after removing the colors from the points. Each point was then turned into a line by assigning it to a random direction according to similar mixture of Gaussians.

(iv) We used California housing prices data-set [35] in witch we introduced uncertainty by removing two of the 9 dimensions each point. We removed one discrete value which created sets and one continuous value which created the lines.

The experiment. For each data set, and for different values of m, k, we conduct the following experiments. For every $\sigma \in \{10 \cdot 2^i\}_{i=0}^{n}$, a coreset $\mathcal{S}_1(\sigma)$ of size σ using our algorithm, and a uniform sample $\mathcal{S}_2(\sigma)$ of the data-set of size σ. Then we generated a set $\mathcal{Q}$ of $|Q| = 500$ queries: half of them are k-means that were computed using generalization of the EM heuristic, and the rest were randomly and uniformly sampled from the ground set. Finally, for every $i \in \{1, 2\}$ we computed the maximum approximation error $\mathcal{S}_i$, $\varepsilon_i(\sigma) = \max_{Q \in \mathcal{Q}} \frac{\left|\sum_{P \in \mathcal{P}} \tilde{D}(\mathcal{S}_i, Q) - \sum_{P \in \mathcal{P}} \tilde{D}(\mathcal{P}, Q)\right|}{\sum_{P \in \mathcal{P}} \tilde{D}(\mathcal{P}, Q)}$

Results for the corresponding databases (i)–(iv) are shown in Fig. 3.

Discussion Our coresets out-preform uniform sampling in most of the experiments. As expected, when the data is uniformly distributed, the sensitivity of each point is close, and uniform sampling is already a coreset. Increasing k and m yields more isolated clusters, which explains why the error for the uniform samplings becomes higher compared to the coreset's error. As is in previous paper, the empirical error is very small compared to the theoretical worst-case bounds, even for small coresets.

5 Conclusion and Future Work

The paper suggests the first coresets for k line-sets clustering via a reduction to coresets for colored-sets clustering of independent interest. We expect that this paper will open a line of research for many possible directions. For example, projective clustering of sets. That is, replacing the *outputted* k point centers by k linear subspaces of $\mathbb{R}^d$, each of dimension $j \geq 1$. Another direction is to handle inputs sets of subspaces, each of dimension $j \geq 1$. Extending to non-linear shapes is another direction.

References

[1] Pankaj K Agarwal, Graham Cormode, Zengfeng Huang, Jeff M Phillips, Zhewei Wei, and Ke Yi. Mergeable summaries. *ACM Transactions on Database Systems (TODS)*, 38(4):1–28, 2013.

[2] Sharareh Alipour. Approximation algorithms for probabilistic k-center clustering. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1–11, 2020.

[3] Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University press, 2009.

[4] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153, 2006.

[5] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[6] Anonymous Authors. Open source code will be provided upon publication of this paper, or reviewer request. `https://github.com/`, 2022.

[7] Hanene Azzag, Gilles Venturini, Antoine Oliver, and Christiane Guinot. A hierarchical ant based clustering algorithm and its use in three real-world applications. *European Journal of Operational Research*, 179(3):906–922, 2007.

[8] Artem Barger and Dan Feldman. Deterministic coresets for k-means of big sparse data. *Algorithms*, 13(4):92, 2020.

[9] F. Bartolozzi, A. de Gaetano, E. Di Lena, S. Marino, L. Nieddu, and G. Patrizi. Operational research techniques in medical treatment and diagnosis: A review. *European Journal of Operational Research*, 121(3):435–466, 2000.

[10] Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *arXiv preprint arXiv:1612.00889*, 2016.

[11] Vladimir Braverman, Shaofeng H-C Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In *International Conference on Machine Learning*, pages 744–753. PMLR, 2019.

[12] Vladimir Braverman, Shaofeng H. C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering with missing values, 2021.

[13] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[14] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 169–182, 2021.

[15] Dan Feldman, Zahi Kfir, and Xuan Wu. Coresets for gaussian mixture models of any shape. *CoRR*, abs/1906.04895, 2019.

[16] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578, 2011.

[17] Dan Feldman and Leonard J Schulman. Data reduction for weighted and outlier-resistant clustering. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1343–1354. SIAM, 2012.

[18] Jie Gao, Michael Langberg, and Leonard J. Schulman. Clustering lines in high-dimensional space: Classification of incomplete data. *ACM Trans. Algorithms*, 7(1), December 2010.

[19] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *2011 IEEE intelligent vehicles symposium (IV)*, pages 963–968. Ieee, 2011.

[20] Sariel Har-Peled. Coresets for discrete integration and clustering. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 33–44. Springer, 2006.

[21] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[22] Lingxiao Huang and Nisheeth K Vishnoi. Coresets for clustering in euclidean spaces: Importance sampling is nearly optimal. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1416–1429, 2020.

[23] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011.

[24] Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. *arXiv preprint arXiv:1605.07139*, 2016.

[25] Ibrahim Jubran, Ernesto Evgeniy Sanches Shayda, Ilan Newman, and Dan Feldman. Coresets for decision trees of signals. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[26] Ibrahim Jubran, Murad Tukan, Alaa Maalouf, and Dan Feldman. Sets clustering. *International Conference on Machine Learning ICML*, 2020.

[27] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2012.

[28] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry*, 28(2-3):89–112, 2004.

[29] Michael Langberg and Leonard J Schulman. Universal ε-approximators for integrals. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 598–607. SIAM, 2010.

[30] David Lewis. Reuters-21578 text categorization test collection, distribution 1.0. `http://www.research/.att.com`, 1997.

[31] Jianguo Li, Weixin Wu, Tao Wang, and Yimin Zhang. One step beyond histograms: Image representation using markov stationary features. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[32] Mario Lucic, Olivier Bachem, and Andreas Krause. Strong coresets for hard and soft bregman clustering with applications to exponential family mixtures. In *Artificial intelligence and statistics*, pages 1–9. PMLR, 2016.

[33] Yair Marom and Dan Feldman. k-means clustering of lines for big data. In *Advances in Neural Information Processing Systems NeaurIPS*, pages 12817–12826, 2019.

[34] Rafail Ostrovsky, Yuval Rabani, Leonard J Schulman, and Chaitanya Swamy. The effectiveness of lloyd-type methods for the k-means problem. *Journal of the ACM (JACM)*, 59(6):1–22, 2013.

[35] R Kelley Pace and Ronald Barry. California housing dataset. `https://scikit-learn.org/stable/datasets/real_world.htmlcalifornia-housing-dataset`, 1997.

[36] Guy Rosman, Mikhail Volkov, Dan Feldman, John W Fisher III, and Daniela Rus. Coresets for k-segmentation of streaming data. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[37] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means clustering. *arXiv preprint arXiv:1812.10854*, 2018.

[38] Xiaobo Shen, Weiwei Liu, Ivor Tsang, Fumin Shen, and Quan-Sen Sun. Compressed k-means for large-scale clustering. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[39] Christian Sohler and David P Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 802–813. IEEE, 2018.

[40] Adiel Statman, Liat Rozenberg, and Dan Feldman. k-means+++: Outliers-resistant clustering. *Algorithms*, 13:311, 11 2020.

[41] Andrea Vattani. K-means requires exponentially many iterations even in the plane. *Discrete & Computational Geometry*, 45(4):596–616, 2011.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Claims from the abstract fully reflect the theorems and algorithms presented in the paper.
   (b) Did you describe the limitations of your work? [Yes] Main limitations are the defined by the complexity of the algorithms and theoretical guarantees for the size of proposed coresets. We give detailed complexity and size analysis in the paper with proofs in the supplementary materials.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We specifically address a connection of the proposed solution in the paper to the problem of Fair Clustering. We hope that further research in this direction will lead to a positive impact on the society.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We have included formal definitions and claims in Section 1 that outline the assumptions that lead to the presented results
   (b) Did you include complete proofs of all theoretical results? [Yes] Proofs for all theoretical results are included in the supplementary materials
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Detailed information for reproducibility is included in the Experimental Results section.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See the Experimental Results section.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See the Experimental Results section.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the Experimental Results section.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We have used open datasets in the Experimental Results section, with proper citations
   (b) Did you mention the license of the assets? [Yes] We properly cite the widely available public datasets as required by their usage policy.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We will publish the open source code of the proposed algorithms.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We were not directly collecting any personal data

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We have included only widely used publicly available datasets in the experimental section of the research.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We have not used crowdsourcing.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]