An Empirical Evaluation of k-Means Coresets*

Abstract

Coresets are among the most popular paradigms for summarizing data. In particular, there exist many high performance coresets for clustering problems such as k-means in both theory and practice. Curiously, there exists little work on comparing the quality of available k-means coresets.

In this paper we perform such an evaluation. First, we show that it is computationally hard to compare the quality of not only two different coreset algorithms, but also of two different output of a (randomized) coreset algorithm. In order to perform an empirical evaluation, we therefore have to work with heuristics. To this end, we propose and analyse a benchmark for coreset comparison. Using this benchmark and real-world data sets, we conduct an exhaustive evaluation of the most commonly used coreset implementations.

1 Introduction

The design an analysis of scalable algorithms has become an important research area over the past two decades. This is particularly important in data analysis, where even polynomial running time might not be enough to handle proverbial *big data* sets. One of the main approaches to deal with the scalability issue is to compress or sketch large data sets into smaller, more manageable ones. The aim of such compression methods is to preserve the properties of the original data, up to some small error, while significantly reducing the number of data points.

Among the most popular and successful paradigms in this line of research are *coresets*. Informally, given a data set A, a coreset $S \subset A$ with respect to a given set of queries Q and query function $f : A \times Q \to \mathbb{R}_{\geq 0}$ approximates the behaviour of A for all queries up to some multiplicative distortion D via

$$\sup_{q \in Q} \max\left(\frac{f(S,q)}{f(A,q)}, \frac{f(A,q)}{f(S,q)}\right) \leq D.$$

Coresets have been applied to a number of problems such as computational geometry [2, 6], linear algebra [24, 28], and machine learning [30, 33]. But the by far most intensively studied an arguably most successful applications of the coreset framework are k-clustering problem.

Here we are given n points A with (potential unit) weights $w : A \to \mathbb{R}_{\geq 0}$ in some metric space with distance function dist and aim to find k centers C such that

$$\mathrm{cost}_A(C) := \frac{1}{n} \sum_{p \in A} \min_{c \in C} w(p) \cdot \mathrm{dist}^z(p, c)$$

is minimized. The most popular variant of this problem is probably the k-means problem in d-dimensional Euclidean space where $z = 2$ and $\mathrm{dist}(x, y) = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$.

A (k, ε)-coreset is now a subset $\Omega \subset A$ with weights $w : \Omega \to \mathbb{R}_{\geq 0}$ such that for any set of k centers C

$$\sup_{C} \max\left(\frac{\mathrm{cost}_A(C)}{\mathrm{cost}_\Omega(C)}, \frac{\mathrm{cost}_\Omega(C)}{\mathrm{cost}_A(C)}\right) \leq 1 + \varepsilon. \tag{1.1}$$

In a long line of work spanning the last 20 years[4, 5, 7, 14, 17, 21, 20, 23, 5, 26, 36], the size of coresets has been steadily improved with the current state of the art yielding a coreset with $\tilde{O}(k\varepsilon^{-4})$ points for a distortion $D \leq (1 + \varepsilon)$ due to Cohen-Addad, Saulpic, and Schwiegelshohn [11].

While we have a good grasp of the theoretical guarantees of these algorithms, our understanding of the empirical performance is somewhat lacking. This is due to two main reasons.

- Experiments are geared towards optimization: Often experiments on coresets are conducted as follows. First, compute coreset(s) with the available algorithm(s). Then run an optimization algorithm. The *best* coreset algorithm is considered to be the one resulting in the clustering with smallest cost.
- Evaluating the quality of a coreset is hard: Given two point sets A and B, it is hard to determine the distortion when considering B as a candidate coreset of A with respect to k-clustering problems in most metrics. Thus, while we can default to the

*The full version of the paper can be accessed at `https://arxiv.org/abs/1902.09310`

worst case guarantees from theory, it is difficult to compare the output of two coreset algorithms for a given data set.

These two reasons are related. Due to the difficulty of evaluating coresets, comparing the outcome of an optimization algorithm is a simple and reasonable alternative inasmuch as no large cost clustering becomes a low cost clustering if the coreset computation was successful. Nevertheless, this method of comparison has the drawback that it is more likely to measure the performance of the underlying optimization problem, rather than evaluating coresets. Moreover, coresets as defined in Eq. (1.1) have a guarantee for *all* candidate solutions C. A relaxed guarantee that only preserves optimal or nearly optimal clusterings are known as weak coresets, which often are easier to compute. Thus, if a candidate coreset algorithm outputs a weak coreset, it may have a bad worst-case distortion which would not be visible when evaluating it on the outcome of an optimization algorithm.

Thus, the purpose of this paper is to systematically evaluate the distortion of various coreset algorithms. On real world data sets, we observe that while all coreset algorithms are generally able to find a solution with good cost, there is a stark difference in the distortion, which shows a differences between optimization and compression are readily observable in practise. In addition, we propose a benchmark for k-means coresets in Euclidean spaces. We argue why this benchmark has properties that make it both hard for all known coreset constructions. We also show how to efficiently estimate the distortion of a candidate coreset on the benchmark.

2 Coreset Algorithms

Though the algorithms vary in details, coreset constructions come in one of the following two flavours:

1. Movement-based constructions: Such algorithms compute a clustering with T points such that $\text{cost}_A(T) \ll \text{OPT}$, where OPT is the cost of an optimal k-means clustering. The coreset guarantee then follows as a consequence of the triangle inequality. These algorithms all have an exponential dependency on the dimension d, and therefore have been overtaken by sampling-based methods. Nevertheless, these constructions are more robust to various constrained clustering formulations [22, 35] and continue to be popular. Examples from theory include [19, 21].

2. Importance sampling: Points are sampled proportionate to their sensitivity which for a point p is defined as $sens(p) := \sup_C \frac{\min_{c\in C} \text{dist}^2(p,c)}{\text{cost}_A(C)}$ and weighted by their inverse sampling probability. In terms of theoretical performance, sensitivity sampling has largely replaced movement based constructions, see for example [15, 27].

Of course, there exist algorithms that draw on techniques from both, see for example [11]. In what follows, we will survey implementations of various coreset constructions that we will evaluate later.

StreamKM++ [1] The popular k-means++ algorithm [3] computes a set of centers K by iteratively sampling a point proportionate to $\min_{q\in K} \text{dist}^2(p,q)$ and adding it to K. The procedure terminates once the desired number of centers has been reached. The first center is typically picked uniformly at random. The StreamKM++ paper runs the k-means++ algorithms for T iterations, where T is the desired coreset size. At the end, every point q in K is weighted by the number of points in A closest to it. While the construction has elements of important sampling, the analysis is largely movement-based.

BICO [18] Combines the very fast, but poor quality clustering algorithm BIRCH [40] with the movement-based analysis from [19, 21]. The clustering is organized by way of a hierarchical decomposition: When adding a point p to one of the coreset points T at level i, it first finds the closest point q in T. If p is too far away from q, a new center is opened at p. Otherwise p is either added to q, or, if adding p to q increases the clustering cost of q beyond a certain threshold, the algorithm attempts to add p to the child-clusters of q. The procedure then continues recursively.

Sensitivity Sampling [14] The simplest implementation of sensitivity sampling first computes an $(O(1), O(1))$ bicriteria K approximation[1], for example by running k-means++ for $2k$ iterations [39]. Let K be the $2k$ clustering thus computed and let K_i be an arbitrary cluster of K with center q_i. Subsequently, the algorithm picks $T - 2k$ points proportionate to $\frac{\text{dist}^2(p,q)}{\text{cost}_{K_i}(\{q_i\})} + \frac{1}{|K_i|}$. Let $|\hat{K}_i|$ be the estimated number of points in the sample. Finally, the algorithm weighs each q_i by $(1+\varepsilon)\cdot|K_i| - |\hat{K}_i|$.

Group Sampling [11] First, the algorithm computes an $O(1)$ approximation (or a bicriteria approximation) K. Subsequently, the algorithm is prepro-

[1] An (α, β) bicriteria approximation computes an α approximation using $\beta \cdot k$ many centers.

cesses the input into groups such that (1) for any two points $p, p' \in K_i$, their cost is identical up to constant factors and (2) for any two clusters K_i, K_j, their cost is identical up to constant factors. In every group, Group-Sampling now samples points proportionate to their cost. The authors of [11] that there always exist a partition into $\log^2 1/\varepsilon$ groups. Points not contained in a group are snapped to their closest center q in K. q is weighted by the number of points snapped to it.

2.1 Dimension Reduction Finally, we also combine coreset constructions with a variety of dimension reduction techniques. Since the seminal paper by Feldman, Schmidt, and Sohler [16], most coreset algorithms have used some form of dimension reduction to eliminate the dependency on d, either by explicitly computing a low-dimensional embedding, see for example [16, 37], or by using the existence of a suitable embedding in the analysis [11, 23].

In particular, movement-based coresets often have an exponential dependency on the dimension, which can be alleviated with some form of dimension reduction, both in theory [35] and in practise [25].

Here the are two main techniques.

Principal Component Analysis: Feldman, Schmidt, and Sohler [16] showed that projecting an input A onto the first $O(k/\varepsilon^2)$ principal components is a coreset, albeit in low dimension. The analysis was subsequently tightened by [8] and extended to other center based cost functions by [36]. Although it's target dimension is generally worse than those based on random projections and terminal embeddings, there is nevertheless reasons for using PCA regardless: It removes noise and thus may make it easier to compute a high quality coreset.

Terminal Embeddings: Given a set of points A in $\mathbb{R}^d$, a terminal embedding $f : \mathbb{R}^d \to \mathbb{R}^m$ preserves the pairwise distance between any point $p \in A$ and any point $q \in \mathbb{R}^d$ up to a $(1 \pm \varepsilon)$ factor. The statement is related to the famous Johnson-Lindenstrauss lemma but it is stronger as it does not apply to only the pairwise distances of A. Nevertheless, the same target dimension is sufficient. Terminal embeddings were studied by [13, 29, 34], with Narayanan and Nelson [34] achieving an optimal target dimension of $O(\varepsilon^{-2} \log n)$, where n is the number of points. For applications to coresets, we refer to [4, 11, 23].

For an overview on practical aspects of dimension reduction, we refer to Venkatsubramanian and Wang [38].

3 Hardness of Coreset Evaluation and a Benchmark

In this section, we first show that it is in general co-NP hard to evaluate the coreset distortion, given two point sets A and B. Thereafter we describe the benchmark and it's properties.

PROPOSITION 3.1. *Given two point sets A and B in $\mathbb{R}^d$ and a sufficiently small (constant) $\varepsilon > 0$, it is co-NP hard to decide whether A is a (k, ε)-coreset of B.*

Proof. First, we recall that for some ε_0 and candidate clustering cost V, it is NP-hard to decide whether there exists a clustering with cost in $[V, (1+\varepsilon_0) \cdot V]$. Therefore, it is co-NP-hard to decide whether there exists a set of centers C such that $\text{cost}_A(C) \geq (1 + \varepsilon_0) \cdot \text{cost}_B(C)$. ∎

We remark that the possible values for ε_0 are determined by the current APX-hardness results. Assuming NP≠P, $\varepsilon_0 \approx 1.07$ and assuming UCG, $\varepsilon_0 \approx 1.17$ [10, 9] for k-means in Euclidean spaces.

3.1 Benchmark Construction In this section, we describe our benchmark. The benchmark has a parameters α which controls the number of points and dimensions. For a given value of k the benchmark consists of $n = k^\alpha$ points and $d = \alpha \cdot k$ rows. It is recursively constructed as follows.

Denote by $\mathbb{1}_k$ the k-dimensional all 1 vector and by v_i the k dimensional vector with entries $(v_i)_j = \begin{cases} -\frac{1}{k} & \text{if } i \neq j \\ \frac{k-1}{k} & \text{if } i = j \end{cases}$. For $\ell \leq \alpha$, recursively define the k^ℓ dimensional vector $v_i^\ell = \begin{bmatrix} (v_i^{\ell-1})_1 \cdot \mathbb{1}_k \\ (v_i^{\ell-1})_2 \cdot \mathbb{1}_k \\ \vdots \\ (v_i^{\ell-1})_1 \cdot \mathbb{1}_k \end{bmatrix}$. Finally, set the column $t = a \cdot k + b$, $a \in \{0, \ldots \alpha - 1\}$ and $b \in \{1, \ldots k\}$, of A to be k^a stacks of $v_b^{\alpha-a}$.

To get a better feel for the construction, we have given two example inputs in Section 3.1. Chris: TODO: Omar could you please rearrange the matrix on the right side in the figure such that it is consistent with the notation? We are generating them left to right, this matrix is written right to left.

We now summarize the key properties of the benchmark. To this end, we require a few notions. Let A be the input matrix. We slightly abuse notation an refer

$$\begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \qquad \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \end{bmatrix}$$

Figure 1: Benchmark construction for $k = 2$ and $\alpha = 3$ (left) and $k = 3$ and $\alpha = 2$ (right).

to A_i as both the ith point as well as the ith row of the matrix A. For a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$, we define that the $n \times k$ indicator matrix $\tilde{X}$ induced by $\mathcal{C}$ via

$$\tilde{X}_{i,j} = \begin{cases} 1 & \text{if } A_i \in C_j \\ 0 & \text{else.} \end{cases}$$

Furthermore, we will also use the $n \times k$ normalized clustering matrix X defined as

$$\tilde{X}_{i,j} = \begin{cases} \frac{1}{\sqrt{|C_i|}} & \text{if } A_i \in C_j \\ 0 & \text{else.} \end{cases}$$

We also recall the following lemma which will allow us to express the k-means cost of a clustering $\mathcal{C}$ with optimally chosen centers in terms of the cost of X and A.

LEMMA 3.1. (FOLKLORE) *Let A be an arbitrary set of points and let $\mu(A) = \frac{1}{|A|}\sum_{p\in A} p$ be the mean. Then for any point c*

$$\sum_{p\in A} \|p - c\|^2 = \sum_{p\in A} \|p - \mu(A)\|^2 + |A| \cdot \|\mu(A) - c\|^2.$$

This lemma proves that for any given cluster C_j, the mean is the optimal choice of center. We also note that any two distinct columns of X are orthogonal. Furthermore $\mathbf{1}\mathbf{1}^T A$ copies the mean into every entry of A. Combining these two observations, we see that the matrix XX^TA maps the ith row of A onto the mean of the cluster it is assigned to. Finally, define the Frobenius norm of an $n \times d$ A by $\|A\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{d} A_{i,j}^2}$. Then the k-means cost of the clustering $\mathcal{C}$ is precisely

$$\|A - XX^TA\|_F^2.$$

Finally, we require the following distance measure on clusterings as proposed by Meila [31, 32]. Given two clusterings $\mathcal{C}$ and $\mathcal{C}'$, the $k \times k$ confusion matrix M is defined as

$$M_{i,j} = |C_i \cap C'_j|.$$

Furthermore for the indicator matrices $\tilde{X}$ and $\tilde{X}'$ induced by $\mathcal{C}$ and $\mathcal{C}'$ we have the identity $M = \tilde{X}^T\tilde{X}'$. Denote by Π_k the set of all permutations over k elements. Then the distance between $\mathcal{C}$ and $\mathcal{C}'$ is defined as

$$d(\mathcal{C}, \mathcal{C}') = 1 - \frac{1}{n}\max_{\pi \in \Pi_k} \sum_{i=1}^{k} M_{i,\pi(i)}.$$

Observe that for clusters that are identical, their distance is 0. The maximum distance between any two k clusterings is always $\frac{k-1}{k}$.

We are now ready to state the desired properties of our benchmark. The benchmark was designed to generate many clusterings such that

1. The distance between these clustering is maximized.
2. The clusterings have equal cost.
3. The clusterings are induced by a set of centers in $\mathbb{R}^d$.

The first and second property ensure that (equally good) solutions we use for evaluation are spread out through the solution space. In other words, we are less likely to only focus on a set of solutions $\mathcal{S}$ for which a low distortion on one $S \in \mathcal{S}$ implies a low distortion for all elements of $\mathcal{S}$. The third property is important as these are the only clusterings the (standard) coreset guarantee has to apply to.

The solutions we consider as given as follows. For the columns $a \cdot k + 1, \ldots (a + 1) \cdot k$, we define the clustering $\mathcal{C}^a = \{C_1^a, \ldots C_k^a\}$ with $A_i \in C_j^a$ if and only if $A_{i,j} > 0$. Let $\tilde{X}^a$ and X^a denote the indicator matrix and clustering matrix, respectively, as induced by $\mathcal{C}^a$.

FACT 3.1. *For $a \neq a'$, we have $d(\mathcal{C}^a, \mathcal{C}^{a'}) = 1 - 1/k$.*

Proof. Consider an arbitrary vector v_i^ℓ. By construction, the positive entries of v_i^ℓ range from $k^{\ell-1} \cdot i + 1$ to $k^{\ell-1} \cdot (i+1)$. Similarly, the positive entries for the vector $v_j^{\ell-1}$ range from range from $k^{\ell-2} \cdot j + 1$ to $k^{\ell-2} \cdot (j+1)$. Therefore, concatenating $v_j^{\ell-1}$ k times into a vector v', v' and v_i^ℓ can share at most one positive coordinate. Inductively, the same holds true for any concatenation of vectors $v_j^{\ell-h}$. Thus, the two clusters induced by the columns formed by concatenating the vectors v can share only a $1/k$ fraction of the points. Since each cluster consists of exactly $k^\alpha/k = k^{\alpha-1}$ points, the confusion matrix M only has entries $\frac{n}{k^2}$ and for any permutation π, we have $d(\mathcal{C}^a, \mathcal{C}^{a'}) = 1 - 1/k$. ∎

FACT 3.2. *For all $a, a' \in \{0, \dots \alpha - 1\}$, we have*

$$\|A - X^a (X^a)^T A\|_F^2 = (\alpha - 1) \cdot (k - 1) \cdot k^{\alpha - 1}.$$

Proof. Without loss of generality, assume $a = 0$; the proof otherwise holds by rearranging rows and columns of A due to the preceding fact. We first note that for any point $A_i \in C_i$, the coordinates $A_{i,\ell}$ are identical for $\ell < k$. Furthermore for the column $\ell \geq k$, we have by construction $\sum_{A_i \in C_j} A_{i,\ell} = k^{\alpha-1} \cdot \frac{k-1}{k} + (k^\alpha - k^{\alpha-1})\frac{1}{k} = k^{\alpha-1} \cdot (\frac{k-1}{k} - (k-1)\frac{1}{k}) = 0$. Therefore, the mean of C_j satisfies $\mu(C_j)_\ell = \begin{cases} A_{i,\ell} & \text{if } \ell < k \\ 0 & \text{else.} \end{cases}$. Thus, $(A - X^a (X^a)^T A)_{i,\ell} = \begin{cases} A_{i,\ell} & \text{if } \ell \geq k \\ 0 & \text{if } \ell < k \end{cases}$ and the cost is $(\alpha - 1) \cdot k \cdot \left(k^{\alpha-1} \cdot \left(\frac{k-1}{k}\right)^2 + k^\alpha - k^{\alpha-1} \cdot \left(\frac{1}{k}\right)^2 \right) = (\alpha - 1)(k-1)k^{\alpha-1}$. ∎

Finally, we show that the means for the clustering $\mathcal{C}^a$ also induce $\mathcal{C}^a$.

FACT 3.3. *For a clustering $\mathcal{C}^a$, let $\mu(C_j^a)$ denote the mean of cluster C_j^a. Then every point of is assigned to its closest center.*

Proof. Again, we assume without loss of generality $a = 0$. Let A_i be an arbitrary point of cluster C_h^a and consider the mean $\mu(C_j^a)_\ell = \begin{cases} A_{i,\ell} & \text{if } \ell < k \\ 0 & \text{else.} \end{cases}$ of cluster C_j^a. By definition, the positive coordinates of A_i are not equal to the positive coordinates of $\mu(C_j^a)$. The only difference in coordinates between the means of $\mu(C_j^a)$ and $\mu(C_h^a)$ are the first k coordinates, as the rest are 0. But here the coordinates of $\mu(C_h^a)$ and A_i are identical, hence $\mu(C_j^a)$ cannot be closer to A_i. ∎

4 Experiments

In this section, we describe our experimental evaluation of four widely used algorithms on real-world datasets and our benchmark. The goal is to assess the quality of the generated k-means coresets. On that account, we measure the distortion. In addition, we want to measure how noise reduction affects the coreset quality. For that, we create PCA transformed version for each of the real-world datasets. As the evaluated algorithms are randomized, we repeat each experiment ten times, reporting the mean distortion and variance for different values of k.

4.1 Algorithms The evaluated algorithms include `SensitivitySample` [12], `GroupSample` [12], `BICO` [18], and `StreamKM++` [1]. BICO is a streaming algorithm that leverages BIRCH's data structure to compute coreset solutions fast. `SensitivitySample` constructs coresets by sampling original points in proportion to their cost. `GroupSample` is another importance sampling-based algorithm which associates points to rings, groups the ring points and then performs the sampling inside the groups. `BC` uses k-means++ where k is the number of target samples in the coreset. We implemented `SensitivitySample`, `GroupSample` and `StreamKM++` in C++17. For BICO, we used the authors' reference implementation[2].

4.2 Datasets We use three common real-world datasets for evaluating k-means coresets — *Census*[3], *Covertype*[4], and *Tower*[5] — and several instances of the proposed benchmark. The *Census* dataset is a small subset of the Public Use Microdata Samples from 1990 US census. It consists of demographic information encoded as 68 categorical attributes of 2,458,285 individuals. *Covertype* is comprised of cartographic descriptions and forest cover type of four wilderness areas in the Roosevelt National Forest of Northern Colorado in the US. It consists of 581,012 records, 54 cartographic variables and one class variable. Although *Covertype* was originally made for classification tasks, it is often used for clustering tasks by removing the class variable [1]. *Tower* is a 2,560 by 1,920 picture of a tower on a hill where each pixel is represented by a RGB color value. This dataset consists of 4,915,200 rows and 3 columns. To evaluate how denoising effects the quality of the output, we apply SVD on *Census* and *Covertype* using the

[2] `https://ls2-www.cs.tu-dortmund.de/grav/en/bico`

[3] `https://archive.ics.uci.edu/ml/datasets/US+Census+Data+(1990)`

[4] https://archive.ics.uci.edu/ml/datasets/covertype

[5] `http://homepages.uni-paderborn.de/frahling/coremeans.html`

Table 1: The sizes of the real-world datasets used for the experimental evaluation

	Data points	Dimensions
Census	2,458,285	68
Covertype	581,012	54
Tover	4,915,200	3

Table 2: The parameter values and the sizes of the benchmark instances used for the experimental evaluation.

k	α	Data points	Dimensions
10	6	1,000,000	60
20	5	3,200,000	100
30	4	810,000	120
40	4	2,560,000	160
50	4	6,250,000	200

k largest singular values. Note that we do not reduce the number of dimensions of the original datasets.

As for the benchmark dataset, instances can be generated via three parameters; the size of clusters k, the number of column blocks α and the scaling factor β. Only parameters k and α determine the size of the dataset. The number of data points in the dataset is given by $n = k^\alpha$ and the number of dimensions is $d = \alpha k$. Our aim was to generate instances that roughly match the sizes of the real-world datasets. The chosen parameters values and the corresponding dataset sizes are shown in Table 2. We generated a set of instances with no scaling i.e., $\beta = 1.0$ (referred to as *Benchmark-1.0*) and with maximum scaling; $\beta = 2.0$ (*Benchmark-2.0*).

4.3 Algorithm Parameters We followed the same experimental procedure with respect to the choice of parameter values for the algorithms as prior works [18, 1]. For the target coreset size, we used $200k$ for all our experiments. On *Census* and *Covertype*, we used k values $\{10, 20, 30, 40, 50\}$, while for *Tower* we used larger cluster sizes $k \in \{20, 40, 60, 80, 100\}$ as in. On the benchmark instances, we settled on $k \in \{10, 20, 30, 40, 50\}$ as a reasonable trade-off between running time and dataset size.

4.4 Results

References

[1] Marcel R. Ackermann, Marcus Märtens, Christoph Raupach, Kamil Swierkot, Christiane Lammersen, and Christian Sohler. Streamkm++: A clustering algorithm for data streams. *ACM Journal of Experimental Algorithmics*, 17(1), 2012.

[2] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and computational geometry, MSRI*, pages 1–30. University Press, 2005.

[3] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 1027–1035, 2007.

[4] Luca Becchetti, Marc Bury, Vincent Cohen-Addad, Fabrizio Grandoni, and Chris Schwiegelshohn. Oblivious dimension reduction for k-means: beyond subspaces and the johnson-lindenstrauss lemma. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1039–1050, 2019.

[5] Vladimir Braverman, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in excluded-minor graphs and beyond. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2679–2696. SIAM, 2021.

[6] Timothy M. Chan. Dynamic coresets. *Discret. Comput. Geom.*, 42(3):469–488, 2009.

[7] Ke Chen. On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications. *SIAM J. Comput.*, 39(3):923–947, 2009.

[8] 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 Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 163–172, 2015.

[9] Vincent Cohen-Addad and Karthik C. S. Inapproximability of clustering in lp metrics. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 519–539. IEEE Computer Society, 2019.

[10] Vincent Cohen-Addad, Karthik C. S., and Euiwoong Lee. On approximability of clustering problems without candidate centers. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2635–2648. SIAM, 2021.

[11] Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual*

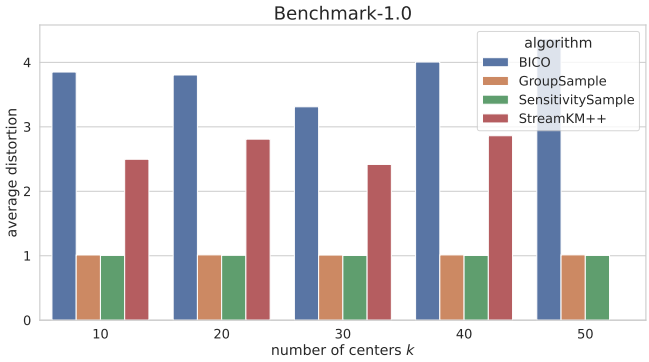

Figure 2: Caption

Event, Italy, June 21-25, 2021, pages 169–182. ACM, 2021.

[12] 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.

[13] Michael Elkin, Arnold Filtser, and Ofer Neiman. Terminal embeddings. *Theor. Comput. Sci.*, 697:1–36, 2017.

[14] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 569–578, 2011.

[15] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 569–578, 2011.

[16] 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 Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1434–1453, 2013.

[17] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM J. Comput.*, 49(3):601–657, 2020.

[18] Hendrik Fichtenberger, Marc Gillé, Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. BICO: BIRCH meets coresets for k-means clustering. In *Algorithms - ESA 2013 - 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings*, pages 481–492, 2013.

[19] Gereon Frahling and Christian Sohler. Coresets in dynamic geometric data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 209–217, 2005.

[20] Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *Discrete & Computational Geometry*, 37(1):3–19, 2007.

[21] 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, Chicago, IL, USA, June 13-16, 2004*, pages 291–300, 2004.

[22] Lingxiao Huang, Shaofeng H.-C. Jiang, and Nisheeth K. Vishnoi. Coresets for clustering with fairness constraints. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7587–7598, 2019.

[23] Lingxiao Huang and Nisheeth K. Vishnoi. Coresets for clustering in euclidean spaces: importance sampling is nearly optimal. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1416–1429. ACM, 2020.

[24] Piotr Indyk, Sepideh Mahabadi, Shayan Oveis Gharan, and Alireza Rezaei. Composable core-sets for determinant maximization problems via spectral spanners. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1675–1694. SIAM, 2020.

[25] Jan-Philipp W. Kappmeier, Daniel R. Schmidt, and Melanie Schmidt. Solving k-means on high-dimensional big data. In Evripidis Bampis, editor, *Experimental Algorithms - 14th International Symposium, SEA 2015, Paris, France, June 29 - July 1, 2015, Proceedings*, volume 9125 of *Lecture Notes in Computer Science*, pages 259–270. Springer, 2015.

[26] Michael Langberg and Leonard J. Schulman. Universal ε-approximators for integrals. In *Proceedings of the*

Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010, pages 598–607, 2010.

[27] Michael Langberg and Leonard J. Schulman. Universal ε-approximators for integrals. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 598–607, 2010.

[28] Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Fast and accurate least-mean-squares solvers. In *Advances in Neural Information Processing Systems*, pages 8307–8318, 2019.

[29] Sepideh Mahabadi, Konstantin Makarychev, Yury Makarychev, and Ilya P. Razenshteyn. Nonlinear dimension reduction via outer bi-lipschitz extensions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1088–1101, 2018.

[30] Tung Mai, Anup B. Rao, and Cameron Musco. Coresets for classification - simplified and strengthened. *CoRR*, abs/2106.04254, 2021.

[31] Marina Meila. Comparing clusterings: an axiomatic view. In Luc De Raedt and Stefan Wrobel, editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 577–584. ACM, 2005.

[32] Marina Meila. The uniqueness of a good optimum for k-means. In William W. Cohen and Andrew W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 625–632. ACM, 2006.

[33] Alexander Munteanu, Chris Schwiegelshohn, Christian Sohler, and David P. Woodruff. On coresets for logistic regression. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6562–6571, 2018.

[34] Shyam Narayanan and Jelani Nelson. Optimal terminal dimensionality reduction in euclidean space. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1064–1069. ACM, 2019.

[35] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In *Approximation and Online Algorithms - 17th International Workshop, WAOA 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers*, pages 232–251, 2019.

[36] Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 802–813, 2018.

[37] Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. *CoRR*, abs/1809.02961, 2018.

[38] Suresh Venkatasubramanian and Qiushi Wang. The johnson-lindenstrauss transform: An empirical study. In Matthias Müller-Hannemann and Renato Fonseca F. Werneck, editors, *Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments, ALENEX 2011, Holiday Inn San Francisco Golden Gateway, San Francisco, California, USA, January 22, 2011*, pages 164–173. SIAM, 2011.

[39] Dennis Wei. A constant-factor bi-criteria approximation guarantee for k-means++. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 604–612, 2016.

[40] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: A new data clustering algorithm and its applications. *Data Min. Knowl. Discov.*, 1(2):141–182, 1997.