

# Clustering on Sliding Windows in Polylogarithmic Space\*

Vladimir Braverman<sup>1</sup>, Harry Lang<sup>2</sup>, Keith Levin<sup>1</sup>, and Morteza Monemizadeh<sup>3</sup>

1 Department of Computer Science  
Johns Hopkins University  
Baltimore, MD, US  
vova@cs.jhu.edu, klevin@jhu.edu

2 Department of Mathematics  
Johns Hopkins University  
Baltimore, MD, US  
hlang8@jhu.edu

3 Computer Science Institute  
Charles University  
Prague, Czech Republic  
monemi@iuuk.mff.cuni.cz

---

## Abstract

In PODS 2003, Babcock, Datar, Motwani and O’Callaghan [4] gave the first streaming solution for the  $k$ -median problem on sliding windows using  $O(\frac{k}{\tau^4} W^{2\tau} \log^2 W)$  space, with a  $O(2^{O(1/\tau)})$  approximation factor, where  $W$  is the window size and  $\tau \in (0, \frac{1}{2})$  is a user-specified parameter. They left as an open question whether it is possible to improve this to polylogarithmic space. Despite much progress on clustering and sliding windows, this question has remained open for more than a decade.

In this paper, we partially answer the main open question posed by Babcock, Datar, Motwani and O’Callaghan. We present an algorithm yielding an exponential improvement in space compared to the previous result given in Babcock, et al. In particular, we give the first polylogarithmic space  $(\alpha, \beta)$ -approximation for metric  $k$ -median clustering in the sliding window model, where  $\alpha$  and  $\beta$  are constants, under the assumption, also made by Babcock et al., that the optimal  $k$ -median cost on any given window is bounded by a polynomial in the window size. We justify this assumption by showing that when the cost is exponential in the window size, no sub-linear space approximation is possible. Our main technical contribution is a simple but elegant extension of *smooth functions* as introduced by Braverman and Ostrovsky [9], which allows us to apply well-known techniques for solving problems in the sliding window model to functions that are not smooth, such as the  $k$ -median cost.

**1998 ACM Subject Classification** I.5.3 Clustering

**Keywords and phrases** Streaming, Clustering, Sliding windows

**Digital Object Identifier** 10.4230/LIPIcs.FSTTCS.2015.350

---

\* The first author was supported in part by the National Science Foundation under Grant No. 1447639, the Google Faculty Award and DARPA grant N660001-1-2-4014. Material presented here is solely the responsibility of the authors and does not represent the official view of DARPA or the Department of Defense.

The second author was supported in part by the Franco-American Fulbright Commission.

The fourth author was supported in part by the center of excellence CE-ITI (project P202/12/G061 of GA ČR).



## 1 Introduction

Throughout the sciences, clustering plays a crucial role in data exploration and analysis. In the typical setting, we are presented with a set of data points that we wish to partition into some number of groups, called clusters, in such a way that some notion of within-cluster similarity is large and, optionally, between-cluster similarity is small. The data in question can then be well-represented by selecting one point from each cluster, typically called cluster centers. Commonly-used objectives for measuring the goodness of a clustering (and a selection of cluster centers) include  $k$ -means,  $k$ -centers and  $k$ -medians [3]. Most such objectives give rise to clustering problems that are known to be **NP**-hard (see, for example, [30] and citations therein). These problems have a rich history in computer science and engineering. Clustering problems date back at least as far as 1857 [35], and a number of clustering algorithms are well-known outside the theory community, most notably Lloyd’s algorithm [32].

In the last 15 years, as data sets have come to outgrow the available memory on most machines, the streaming model of computation has emerged as a popular area of algorithmic research. In this model, computation must be performed using memory of size sublinear in the input, and (typically) using at a single pass over the data [2, 33]. Several clustering problems have been addressed in the streaming model, including  $k$ -median [25, 13, 27, 24],  $k$ -means [14, 23] and facility location [19].

While the streaming model is a sensible one for many applications, it is less suitable for some applications in domains such as network monitoring and social media [15, 17, 16, 34], where observations that have arrived more recently are in some sense more important to the computation being performed than are older observations. For example, when tracking topics in social media, a researcher may wish to have topics decay over time. The sliding window model, a variation on the streaming model, was developed to better capture these situations [20, 8, 11, 7, 10]. In this model, data arrives in a stream, and the goal is to maintain a computation only on the most recent elements. The sliding window model has received renewed attention in recent years [9, 18, 5], but no theoretical results for clustering in the sliding window model have been published since 2003 [4]. This most recent result gives a solution to the  $k$ -median clustering problem, which has been comparatively well-studied by streaming algorithm researchers. Recent years have seen impressive results yielding polylogarithmic space solutions in both the insertion-only streaming model [13, 27] and the insertion-deletion model [28, 24, 29], but to date the question of whether or not analogous results hold in the sliding window model remained open. In particular, the following question by Babcock et al. [4] has remained open for more than a decade:

*“Whether it is possible to maintain approximately optimal medians in polylogarithmic space (as Charikar et al. [13] do in the stream model without sliding windows), rather than polynomial space, is an open problem.”*

### 1.1 Our Contribution

In the current work, we partially answer the question posed by Babcock, et al. [4] in the affirmative. Specifically, we give the first polylogarithmic space  $(\alpha, \beta)$ -approximation algorithm for  $k$ -median clustering in the sliding window model under the assumption that the optimal  $k$ -median cost is at most polynomial in the window size. We note that this boundedness assumption is also made by Babcock, et al. (see Lemma 5 of [4]). We justify this assumption by showing that when the optimal  $k$ -median cost is exponential in the window size, no sublinear space approximation is possible. This is in contrast to the insert-only model, where no such boundedness assumption is necessary [13, 27].

## 1.2 Related Work

$k$ -median clustering of points in arbitrary metric spaces is relatively well-studied in the insertion-only streaming model. Guha, Mishra, Motwani and O’Callaghan [25] were the first to propose an insertion-only streaming algorithm for the  $k$ -median problem. They gave a  $2^{O(1/\epsilon)}$ -approximation streaming algorithm that uses  $O(n^\epsilon)$  space, where  $\epsilon < 1$ . Later, Charikar, O’Callaghan, and Panigrahy [13] exponentially improved this algorithm by developing a constant factor approximation (insertion-only) streaming algorithm using  $O(k \cdot \log^2 n)$  space. Their approach, somewhat similarly to [25], operates in phases, maintaining a set of  $O(k \log n)$  candidate centers with the invariant that at any time during the algorithm, the candidate centers yield a suitably low clustering cost. Once the entire stream has been observed, an offline  $k$ -median clustering of this (weighted) collection of candidate centers is used as the solution for the whole stream.

The  $k$ -median clustering in the geometric setting where points are taken from a  $d$ -dimensional Euclidean space  $\mathbb{R}^d$  is also well-studied in the insertion-only streaming model. In particular, Har-Peled and Mazumdar [27] used (strong) coresets to obtain a  $(1 + \epsilon)$ -approximation algorithm for  $k$ -median and  $k$ -means problems in the insertion-only streaming model. Roughly speaking, a strong  $(k, \epsilon)$ -coreset for  $k$ -median is a weighted subset  $S$  of  $P$ , so that for any set of  $k$  centers in  $\mathbb{R}^d$ , the weighted sum of distances from points in  $S$  to the nearest centers is approximately the same as (differs by a factor of  $(1 \pm \epsilon)$  from) the sum of distances from points in  $P$  to the nearest centers. Their coreset was of size  $O(k\epsilon^{-d} \log n)$ . In the streaming model of computation they implemented their coreset (using the famous Merge-and-Reduce approach [6, 1]) using  $O(k\epsilon^{-d} \log^{2d+2} n)$  space. Later, Har-Peled and Kushal [26] showed that one can construct  $(k, \epsilon)$ -coresets for  $k$ -median with size independent of  $n$ , namely of size  $O(k^2\epsilon^{-d})$ . However, in the streaming model, the implementation of the new coreset (once again, using the Merge-and-Reduce approach) does not give a significant improvement in space usage. Very recently, Feldman, Fiat, and Sharir [22] extended this type of coreset for linear centers or facilities where facilities can be lines or flats.

For high-dimensional spaces, Chen [14] proposed a  $(k, \epsilon)$ -coreset of size  $O(k^2 d \epsilon^{-2} \log^2 n)$  in the insertion-only streaming model using  $O(k^2 d \epsilon^{-2} \log^8 n)$  space. Chen’s coreset works for general metric spaces as well, where he developed a technique that produces a coreset of size  $O(k\epsilon^{-2} \log n (k \log n + \log(1/\delta)))$  with probability of success  $1 - \delta$ . If we plug the very recent 2.661-approximation algorithm for the  $k$ -median problem due to Byrka, Pensyl, Rybicki, Srinivasan, and Trinh [12] into Chen’s  $(k, \epsilon)$ -coreset construction, we obtain an  $O(k^2 \epsilon^{-2} \log(1/\delta) \log^9 n)$ -space 5.322-approximation algorithm in the insertion-only streaming model with probability of success  $1 - \delta$  for  $0 < \delta < 1$ .

To the best of our knowledge, there is no insertion-deletion streaming algorithm for  $k$ -median or  $k$ -means clustering when points are from an arbitrary metric space. However, for geometric  $k$ -median and  $k$ -means clustering, Frahling and Sohler [24] showed that they can maintain a  $(k, \epsilon)$ -coreset of size  $O(k\epsilon^{-d-2} \log n)$  using a different coreset construction (than [27, 26]) for data streams with insertions and deletions. This model of data streams with insertions and deletions for geometric problems was introduced by Indyk [28] and is known as *dynamic geometric data streams*. Frahling and Sohler’s algorithm uses  $O(k^2 \epsilon^{-2d-4} \log^7 n)$  space for the  $k$ -median problem. They further showed that similar coresets exist for the geometric versions of Max-Cut, maximum weighted matching, maximum travelling salesperson, maximum spanning tree, and average distance problems, which in turn give  $O(\epsilon^{-2d-4} \log^7 n)$ -space streaming algorithms for these problems in data streams with insertions and deletions.

In contrast to the insertion-only streaming model and dynamic geometric streaming model, little work has been done on the  $k$ -median problem in the sliding window model,

■ **Table 1** Known results for  $k$ -median problem in data streams. Note that the current work as well as the first algorithm of [4] give bicriteria solutions to the  $k$ -median problem, while the other results in this table return precisely  $k$  centers.

Reference	Metric space	Stream model	Approx.	Centers	Space
[25]	General	Insertion-only	$2^{O(1/\tau)}$	$k$	$O(n^\tau)$
[13]	General	Insertion-only	$O(1)$	$k$	$O(k \log^2 n)$
[14]+[12]	General	Insertion-only	5.322	$k$	$O(k^2 \epsilon^{-2} \log(1/\delta) \log^9 n)$
[27]	Euclidean	Insertion-only	$(1 + \epsilon)$	$k$	$O(k \epsilon^{-d} \log^{2d+2} n)$
[14]	Euclidean	Insertion-only	$(1 + \epsilon)$	$k$	$O(k^2 d \epsilon^{-2} \log^8 n)$
[24]	Euclidean	Insertion-deletion	$(1 + \epsilon)$	$k$	$O(k^2 \epsilon^{-2d-4} \log^7 n)$
[4]	General	Sliding Windows	$2^{O(1/\tau)}$	$2k$	$O(k \tau^{-4} W^{2\tau} \log^2 W)$
[4]	General	Sliding Windows	$2^{O(1/\tau)}$	$k$	$O(k \tau^{-4} W^{2\tau} \log^2 W)$
This work	General	Sliding Windows	35	$2k$	$O(k^2 \epsilon^{-3} \log(1/\delta) \log^{10} W)$

where we wish to produce a solution only on the most recent  $W$  elements in the data stream. The result given in [4] is, to our knowledge, the only previously existing solution in this model. The algorithm given in [4] finds an  $O(2^{O(\frac{1}{\tau})})$ -approximation to the  $k$ -median problem in the sliding window model for  $0 < \tau < \frac{1}{2}$  using  $2k$ -centers and requires memory of size  $O(\frac{k}{\tau^4} W^{2\tau} \log^2 W)$ . With an additional step, they reduce the number of centers from  $2k$  to  $k$  using the same space and with the same approximation ratio. We leave as an open problem whether a similar approach can be applied to our algorithm, which produces a bicriteria solution with between  $k$  and  $2k$  centers.

Table 1 summarizes the known results for clustering problems in various streaming models.

## Outline

Section 2 establishes notation and background for the remainder of the paper. Section 3 presents our main result. Section 4 proves a lower bound on the space required to find an approximate  $k$ -median solution when the optimal cost is exponential in the window size.

## 2 Preliminaries

We will begin by introducing some notation and basic definitions. We first define the metric  $k$ -median clustering problems. Later we will define the sliding window model, smooth functions, and smooth histograms. Finally, we will illustrate with an example that the  $k$ -median clustering is not smooth (and in fact, neither are many other clustering problems), but fortunately, we show we can compute  $k$ -median approximately in the sliding windows model, if we relax the constraint of returning exactly  $k$  centers.

### 2.1 Metric and Geometric $k$ -Median Problems

Let  $(X, \text{dist})$  be a metric space where  $X$  is a set of points and  $\text{dist} : X \times X \rightarrow \mathbb{R}$  is a distance function defined over the points of  $X$ . Let  $\text{dist}(p, Q) = \min_{q \in Q} \text{dist}(p, q)$  denote the distance between a point  $p \in X$  and a set  $Q \subseteq X$ .

Let  $P \subseteq X$  be a subset of points. We define  $\rho_P = \min_{p, q \in P, p \neq q} \text{dist}(p, q)$  as the minimum distance between two distinct points in a set  $P$ .

► **Definition 2.1** (Metric  $k$ -median). Let  $P \subseteq X$  be a set of  $n$  points in a metric space  $(X, d)$  and let  $k \in \mathbb{N}$  be a natural number. Suppose  $C = \{c_1, \dots, c_k\}$  is a set of  $k$  centers. The clustering of point set  $P$  using  $C$  is the partitioning of  $P$  such that a point  $p \in P$  is in cluster  $C_i$  if  $c_i \in C$  is the nearest center to  $p$  in  $C$ , that is point  $p$  is assigned to its nearest center  $c_i \in C$ . The cost of  $k$ -median clustering by  $C$  is  $\text{COST}(P, C) = \sum_{p \in P} \text{dist}(p, C)$ . The metric  $k$ -median problem is to find a set  $C \subseteq P$  of  $k$  centers that minimizes the cost  $\text{COST}(P, C)$ , that is

$$\text{COST}(P, C) = \sum_{p \in P} \text{dist}(p, C) = \min_{C' \subseteq P: |C'|=k} \text{COST}(P, C') = \min_{C' \subseteq P: |C'|=k} \sum_{p \in P} \text{dist}(p, C'),$$

where  $\text{dist}(p, C) = \min_{c \in C} \text{dist}(p, c)$  and  $\text{dist}(p, C') = \min_{c \in C'} \text{dist}(p, c)$

We define  $\text{OPT}(P, k) = \min_{C' \subseteq P: |C'|=k} \text{COST}(P, C')$  to be the minimum  $k$ -median cost of  $P$ . Since the metric  $k$ -median problem is known to be **NP**-hard [30], we will focus on *approximation* algorithms.

► **Definition 2.2** ( $(\alpha, \beta)$ -approximation algorithm). We say an algorithm  $\mathcal{A}$  is an  $(\alpha, \beta)$ -approximation algorithm for the  $k$ -median problem on point set  $P \subseteq X$  if  $\mathcal{A}(P, k)$  returns a set  $C \subseteq P$  of at most  $\beta \cdot k$  centers whose cost is  $\alpha$ -approximation of  $\text{OPT}(P, k)$ , that is,  $\text{COST}(P, C) \leq \alpha \cdot \text{OPT}(P, k)$ .

## 2.2 Sliding Windows Model

Let  $(X, d)$  be a metric space. Let  $P = \{p_1, p_2, \dots, p_n\} \subseteq X$  be a point set of size  $|P| = n$ . In the insertion-only streaming model [2, 27, 13], we think of a (mostly adversarial) permutation  $\{p'_1, p'_2, \dots, p'_n\}$  of point set  $P$  given in a streaming fashion and the goal is to compute a function  $f$  exactly or approximately at the end of the stream using sublinear space in  $n$ , i.e.,  $o(n)$ . Here we say point  $p'_t$  is revealed at time  $t$ .

The *sliding windows model* [20] is a generalization of the insertion-only streaming model in which we seek to compute a function  $f$  over only the  $W$  most recent elements of the stream. Given a current time  $t$  of the stream, we consider a window  $\mathcal{W}$  of size  $W$  consisting of points that are inserted in the interval  $[\max(t - W, 1), \dots, t]$ . Here we still assume  $W$  is large enough that we cannot store all of window  $\mathcal{W}$  in memory, for example  $W = \Omega(n)$ ; otherwise computing function  $f$  over window  $\mathcal{W}$  will be trivial. A point  $p$  in the current window  $\mathcal{W}$  is called *active*, and *expired*, otherwise.

## 2.3 Smooth Function and Smooth Histogram

Braverman and Ostrovsky [9] introduced *smooth histograms* as an effective method to compute *smooth functions* in the sliding windows model. A smooth function is defined as follows.

► **Definition 2.3** ( $(\epsilon, \epsilon')$ -smooth function [9]). Let  $f$  be a function,  $0 < \epsilon, \epsilon' < 1$ , and  $c$  be a constant number. We say  $f$  is a  $(\epsilon, \epsilon')$ -smooth function if function  $f$  is non-negative (i.e.,  $f(A) \geq 0$ ), non-decreasing (i.e., for  $A \subseteq B$ ,  $f(A) \leq f(B)$ ), and polynomially bounded  $f(A) \leq O(|A|^c)$  such that

$$f(B) \geq (1 - \epsilon) \cdot f(A \cup B) \quad \text{implies} \quad f(B \cup C) \geq (1 - \epsilon') \cdot f(A \cup B \cup C) .$$

Interestingly, many functions are smooth. For instance, sum, count, min, diameter,  $L_p$ -norms, frequency moments and the length of the longest subsequence are all smooth functions.

We define  $[a] = \{1, 2, 3, \dots, a\}$  and  $[a, b] = \{a, a + 1, a + 2, \dots, b\}$  for  $a \leq b$  and  $a, b \in \mathbb{N}$ . When there is no danger of confusion, we denote the set of points  $\{p_a, p_{a+1}, \dots, p_b\}$  as simply  $[a, b]$ . For example, we denote  $f(\{p_a, p_{a+1}, \dots, p_b\})$  by  $f([a, b])$ .

To maintain a smooth function  $f$  on sliding windows, Braverman and Ostrovsky [9] proposed a data structure that they called *smooth histograms* which is defined as follows.

► **Definition 2.4** (Smooth histogram [9]). Let  $0 < \epsilon, \epsilon' < 1$  and  $\alpha > 0$ . Let  $f$  be an  $(\epsilon, \epsilon')$ -smooth function. Suppose there exists an *insertion-only* streaming algorithm  $\mathcal{A}$  that calculates an  $\alpha$ -approximation  $f'$  of  $f$ . The *smooth histogram* is a data structure consisting of an increasing set of indices  $X_N = [x_1, x_2, \dots, x_t = N]$  and  $t$  instances  $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_t$  of  $\mathcal{A}$  such that

1.  $p_{x_1}$  is expired and  $p_{x_2}$  is active or  $x_1 = 0$ .
2. For  $1 < i < t - 1$  one of the following holds
  - a.  $x_{i+1} = x_i + 1$  and  $f'([x_{i+1}, N]) \leq (1 - \epsilon')f'([x_i, N])$ ,
  - b.  $f'([x_{i+1}, N]) \geq (1 - \epsilon)f'([x_i, N])$  and if  $i \in [t - 2]$ ,  $f'([x_{i+2}, N]) \leq (1 - \epsilon')f'([x_i, N])$ .
3.  $\mathcal{A}_i = \mathcal{A}([x_i, N])$  maintains  $f'([x_i, N])$ .

Observe that the first two elements of sequence  $X_N$  always sandwiches the current window  $\mathcal{W}$  of size  $W$ , that is,  $x_1 \leq N - W \leq x_2$ . Braverman and Ostrovsky [9] used this observation to show that either  $f'([x_1, N])$  or  $f'([x_2, N])$  is a reasonably good approximation of  $f'([N - W, N])$ . In particular, using smooth histograms, they proved the following theorem.

► **Theorem 2.5.** [9] Let  $0 < \epsilon, \epsilon' < 1$  and  $\alpha, \beta > 0$ . Let  $f$  be an  $(\epsilon, \epsilon')$ -smooth function. Suppose there exists an insertion-only streaming algorithm  $\mathcal{A}$  that calculates an  $\alpha$ -approximation  $f'$  of  $f$  using  $g(\alpha)$  space and  $h(\alpha)$  update time. Then, there exists a sliding window algorithm  $\mathcal{B}$  that maintains  $(1 \pm (\alpha + \epsilon))$ -approximation  $f''$  of  $f$  using  $O(\beta^{-1} \cdot \log n \cdot (g(\alpha) + \log n))$  space and  $O(\beta^{-1} \cdot \log n \cdot h(\alpha))$  update time.

### 2.4 $k$ -median is not a smooth function

Unfortunately, it is simple to see that many clustering functions are not smooth. Here we give a simple example showing that the  $k$ -median cost is not a smooth function.

► **Lemma 2.6.**  *$k$ -median clustering is not a smooth function.*

**Proof.** We give a counterexample showing that  $k$ -median clustering is not a smooth function. Assume that we have points  $p, q, r \in X$  and  $k = 2$ . Then  $\text{OPT}(\{p, q\}, k) = 0$  and  $\text{OPT}(\{q\}, k) = 0$ , but  $\text{OPT}(\{q, r\}, k) = 0$  and  $\text{OPT}(\{p, q, r\}, k) = \min(\text{dist}(p, q), \text{dist}(p, r), \text{dist}(q, r))$  which can be arbitrarily large. ◀

However, the following lemma shows that we can compute  $k$ -median approximately in the sliding windows model if we relax the constraint of returning exactly  $k$  centers.

► **Lemma 2.7.** *Let  $A, B, C \subset X$  be three point sets. Let  $\lambda > 1$  be a parameter. Then,*

$$\text{OPT}(B, k) \geq \frac{1}{\lambda} \cdot \text{OPT}(A \cup B, k)$$

*implies*

$$\text{OPT}(B \cup C, k) \geq \frac{1}{(\lambda + 1)} \cdot \text{OPT}(A \cup B \cup C, 2k) .$$

**Proof.** We bound the optimal  $2k$ -median cost of the set  $A \cup B \cup C$  as follows. First of all, the optimal  $2k$ -median cost of  $A \cup B \cup C$  is upper-bounded by the optimal  $k$ -median cost of  $A$  plus the optimal  $k$ -median cost of  $B \cup C$

$$\text{OPT}(A \cup B \cup C, 2k) \leq \text{OPT}(A, k) + \text{OPT}(B \cup C, k) ,$$

as otherwise we can always replace the  $2k$  optimal centers of  $A \cup B \cup C$  by the  $k$  optimal centers of  $A$  and the  $k$  optimal centers of  $B \cup C$  contradicting the minimum cost of the  $2k$  optimal centers  $A \cup B \cup C$ . Now we have,

$$\begin{aligned} \text{OPT}(A \cup B \cup C, 2k) &\leq \text{OPT}(A, k) + \text{OPT}(B \cup C, k) \leq \text{OPT}(A \cup B, k) + \text{OPT}(B \cup C, k) \\ &\leq \lambda \cdot \text{OPT}(B, k) + \text{OPT}(B \cup C, k) \\ &\leq \lambda \cdot \text{OPT}(B \cup C, k) + \text{OPT}(B \cup C, k) = (\lambda + 1)\text{OPT}(B \cup C, k), \end{aligned}$$

which completes the proof. ◀

### 3 $(\alpha, \beta)$ -approximation for $k$ -median problem on sliding windows

Here we state our main result.

► **Theorem 3.1.** *Let  $\alpha > 1$  be a constant,  $\lambda = (1 + \epsilon)$  and  $k \in \mathbb{N}$  be a parameter. Let  $W$  be the size of the sliding window. Suppose the optimal  $k$ -median cost of a point set  $P \subset X$  is polynomially bounded, that is  $\text{OPT}(P, k) = |P|^{O(c)} \cdot \rho_P$  for sufficiently large constant  $c$ , where  $\rho_P$  is the minimum distance between two distinct points in  $P$ . Suppose there exists an insertion-only streaming algorithm  $\mathcal{A}(P, k)$  that maintains an  $\alpha$ -approximation set of  $k$  centers for  $P$  using  $g(\alpha)$  space and  $h(\alpha)$  update time. Then there is an  $[\alpha(\alpha(1 + \epsilon) + 1), 2]$ -approximation sliding windows algorithm that uses  $O(g(\alpha) \cdot \epsilon^{-1} \cdot \log(W \cdot \rho_P))$  space and has  $O(h(\alpha) \cdot \epsilon^{-1} \cdot \log(W \cdot \rho_P))$  update time.*

#### Overview of Algorithm 1 ( $k$ -MEDIAN)

Suppose our stream is  $S = [p_1, p_2, p_3, \dots, p_N, \dots, p_n]$  and we are interested to maintain a  $k$ -median clustering of a window  $\mathcal{W}$  of  $W$  most recent points in stream  $S$ . We maintain an ordered list  $x_i \in X = [x_1, x_2, \dots, x_t]$  of  $t = O(\epsilon^{-1} \cdot \log(n \cdot \rho_P))$  indices for  $x_i \in \{1, \dots, N\}$ . Denote by  $\mathcal{A}([i, j], k)$  an instance of algorithm  $\mathcal{A}$  run on points  $\{p_i, p_{i+1}, \dots, p_j\}$ . For every index  $x_i$  we run two instances of insertion-only streaming algorithm  $\mathcal{A}(P, k)$ . One instance is  $\mathcal{A}([x_i, N], k)$  that maintains a set  $C_{i,k}$  of  $k$  centers for the point set  $\{p_{x_i}, p_{x_{i+1}}, \dots, p_N\}$ . The other instance is  $\mathcal{A}([x_i, N], 2k)$ , which maintains a set  $C_{i,2k}$  of  $2k$  centers for the point set  $\{p_{x_i}, p_{x_{i+1}}, \dots, p_N\}$ .

Upon arrival of a new point  $p_N$ , we feed  $p_N$  to instances  $\mathcal{A}([x_i, N], k)$  and  $\mathcal{A}([x_i, N], 2k)$  for every  $x_i \in X$ . We also instantiate two instances  $\mathcal{A}([N, N], k)$  and  $\mathcal{A}([N, N], 2k)$ . We then go through indices  $1 \leq i \leq t - 2$  and find the greatest  $j > i$  for which  $\text{COST}([x_j, N], C_{j,k}) \geq \frac{1}{\lambda} \cdot \text{COST}([x_i, N], C_{i,k})$ , and eliminate all indices  $x_r$  and their instances  $\mathcal{A}$  for  $i < r < j$ . We then update the indices in sequence  $X$  accordingly. Finally, we find the smallest index  $i$  whose  $p_{x_i}$  is expired and  $p_{x_{i+1}}$  is active. For all  $r < i$ , we delete  $x_r$  and its instances and update the indices in sequence  $X$ . At the end, we return the center set  $C_{i,2k}$  of  $2k$  centers maintained by  $\mathcal{A}([x_1, N], 2k)$  as our solution.

**Algorithm 1**  $k$ -MEDIAN in Sliding Windows.

**Input:** A stream  $S = [p_1, p_2, p_3, \dots, p_N, \dots, p_n]$  of points from a metric space  $(X, \text{dist})$  and the sliding window size  $W$ .

**Output:** A center set  $C_{1,2k}$  of  $2k$  centers that is an  $[\alpha(\alpha(1 + \epsilon) + 1), 2]$ -approximation of  $\text{OPT}(P, k)$ .

**Update Process, upon the arrival of new point  $p_N$ :**

- 1: **for**  $x_i \in X = [x_1, x_2, \dots, x_t]$  **▶** (where  $x_i \in \{1, \dots, N\}$ ) **do**
- 2:   Let  $C_{i,k} = \mathcal{A}([x_i, N], k)$  be the set of  $k$  centers maintained by  $\mathcal{A}([x_i, N], k)$ , where  $[x_i, N] = \{p_{x_i}, p_{x_{i+1}}, \dots, p_N\}$ .
- 3:   Let  $C_{i,2k} = \mathcal{A}([x_i, N], 2k)$  be the set of  $2k$  centers maintained by  $\mathcal{A}([x_i, N], 2k)$ .
- 4: Let  $t = t + 1$ ,  $x_t = N$ .
- 5: Let  $C_{t,k} = \mathcal{A}([N, N], k)$  and  $C_{t,2k} = \mathcal{A}([N, N], 2k)$ , where  $[N, N]$  contains only point  $p_N$ .
  
- 6: **for**  $i = 1$  to  $t - 2$  **do**
- 7:   Find the greatest  $j > i$  such that  $\text{COST}([x_j, N], C_{j,k}) \geq \frac{1}{\lambda} \cdot \text{COST}([x_i, N], C_{i,k})$ .
- 8:   For  $i < r < j$ , delete  $x_r$  and center sets  $C_{r,k}$  and  $C_{r,2k}$ .
- 9:   Update the indices in sequence  $X$  accordingly.
- 10: Let  $i$  be the smallest index such that  $p_{x_i}$  is expired and  $p_{x_{i+1}}$  is active.
- 11: **for**  $r < i$  **do**
- 12:   Delete  $x_r$  and center sets  $C_{r,k}$  and  $C_{r,2k}$ .
- 13:   Update the indices in sequence  $X$ .

**Output Process:**

- 1: Return  $C_{1,2k}$  maintained by  $\mathcal{A}([x_1, N], 2k)$ .

**Analysis**

Next we prove Theorem 3.1. First we prove the approximation factor that we claim in this theorem.

**▶ Lemma 3.2.** *For assumptions of Theorem 3.1, Algorithm 1 maintains an  $[\alpha(\alpha\lambda + 1), 2]$ -approximation set of  $k$  centers for  $P$  in the sliding windows model, i.e.,*

$$\text{COST}([x_1, N'], C_{1,2k}) \leq \alpha(\alpha \cdot \lambda + 1) \cdot \text{OPT}(\mathcal{W}, k) ,$$

where  $\mathcal{W}$  is the current window of size  $W$ , that is the points in the interval  $[\max(N' - W, 1), N']$ .

**Proof.** Let us fix the arrival of a new point  $p_N$  from stream  $S = [p_1, p_2, p_3, \dots, p_N, \dots, p_n]$ . From Lines (10) to (13) of Algorithm 1 we always have  $x_1 \leq N - W \leq x_2$  where  $W$  is the window size. Moreover, based on Lines (6) to (9), we have

$$\text{COST}([x_2, N], C_{2,k}) \geq \frac{1}{\lambda} \cdot \text{COST}([x_1, N], C_{1,k}) .$$

Since Algorithm  $\mathcal{A}(P, k)$  is an insertion-only streaming algorithm that maintains a set of  $k$  centers with an  $\alpha$ -approximation guarantee of  $\text{OPT}(P, k)$ , we have

$$\text{OPT}([x_2, N], k) \leq \text{COST}([x_2, N], C_{2,k}) \leq \alpha \cdot \text{OPT}([x_2, N], k)$$

and

$$\text{OPT}([x_1, N], k) \leq \text{COST}([x_1, N], C_{1,k}) \leq \alpha \cdot \text{OPT}([x_1, N], k) .$$



Therefore,

$$\alpha \text{OPT}([x_2, N], k) \geq \text{COST}([x_2, N], C_{2,k}) \geq \frac{1}{\lambda} \cdot \text{COST}([x_1, N], C_{1,k}) \geq \frac{1}{\lambda} \cdot \text{OPT}([x_1, N], k),$$

which means

$$\text{OPT}([x_2, N], k) \geq \frac{1}{\alpha \cdot \lambda} \cdot \text{OPT}([x_1, N], k).$$

Imagine the arrival of a new point  $p_{N'}$  from the stream

$$S = [p_1, p_2, p_3, \dots, p_N, \dots, p_{N'}, \dots, p_n]$$

for which  $x_1 \leq N' - W \leq x_2$  where  $N' > N$ . Denote our window by  $\mathcal{W} = \{p_{N'-W}, \dots, p_{N'}\}$  and let  $A = [x_1, x_2) = \{p_{x_1}, p_{x_1+1}, \dots, p_{x_2-1}\}$ ,  $B = [x_2, N] = \{p_{x_2}, p_{x_2+1}, \dots, p_N\}$ , and  $C = \{p_{N+1}, p_{N+2}, \dots, p_{N'}\}$ . Observe that  $A \cup B = [x_1, N] = \{p_{x_1}, p_{x_1+1}, \dots, p_N\}$  and

$$B \cup C \subseteq \mathcal{W} = \{p_{N'-W}, \dots, p_{N'}\} \subseteq A \cup B \cup C = \{p_{x_1}, \dots, p_{N'}\}.$$

Now we use Lemma 2.7 which says if  $\text{OPT}(B, k) \geq \frac{1}{\lambda} \cdot \text{OPT}(A \cup B, k)$ , we then have  $\text{OPT}(B \cup C, k) \geq \frac{1}{(\lambda+1)} \cdot \text{OPT}(A \cup B \cup C, 2k)$ . We replace  $\lambda$  by  $\alpha\lambda$  to obtain

$$\begin{aligned} \text{OPT}(B, k) &= \text{OPT}([x_2, N], k) \geq \frac{1}{\alpha \cdot \lambda} \cdot \text{OPT}(A \cup B, k) \\ &= \frac{1}{\alpha\lambda} \text{OPT}([x_1, N], k) = \frac{1}{\alpha\lambda} \text{OPT}(\{p_{x_1}, p_{x_1+1}, \dots, p_N\}, k). \end{aligned}$$

Therefore, we have

$$\begin{aligned} \text{OPT}(\mathcal{W}, k) &\geq \text{OPT}(B \cup C, k) = \text{OPT}([x_2, N'], k) = \text{OPT}(\{p_{x_2}, p_{x_2+1}, \dots, p_{N'}\}, k) \\ &\geq \frac{1}{(\alpha \cdot \lambda + 1)} \cdot \text{OPT}(A \cup B \cup C, 2k) = \frac{1}{(\alpha \cdot \lambda + 1)} \cdot \text{OPT}([x_1, N'], 2k) \\ &= \frac{1}{\alpha(\alpha \cdot \lambda + 1)} \cdot \text{COST}([x_1, N'], C_{1,2k}), \end{aligned}$$

since  $\mathcal{A}([x_1, N'], 2k)$  returns an  $\alpha$ -approximation  $2k$ -median center set  $C_{1,2k}$  of point set  $[x_1, N']$ , i.e.,

$$\text{COST}([x_1, N'], C_{1,2k}) \leq \alpha \cdot \text{OPT}([x_1, N'], 2k).$$

Therefore, we have

$$\text{COST}([x_1, N'], C_{1,2k}) \leq (\alpha(\alpha \cdot \lambda + 1)) \cdot \text{OPT}(\mathcal{W}, k),$$

which proves the lemma.  $\blacktriangleleft$

Next, we prove the space usage of Algorithm  $k$ -MEDIAN.

► **Lemma 3.3.** *For assumptions of Theorem 3.1, Algorithm 1 uses  $O(g(\alpha) \cdot \epsilon^{-1} \cdot \log(W \cdot \rho_P))$  space and has  $O(h(\alpha) \cdot \epsilon^{-1} \cdot \log(W \cdot \rho_P))$  update time.*

**Proof.** It is clear that the space is  $O(g(\alpha) \cdot t)$  and that time is  $O(h(\alpha) \cdot t)$ , so it suffices to prove that  $t = O(\epsilon^{-1} \log(W \cdot \rho_P))$ .

As a loop invariant for Line 1 upon arrival of a new point, we will prove the bound  $t < t^*$ . Since Line 4 is the only place where  $t$  is incremented, we will have that  $t \leq t^*$  throughout the algorithm.

In the previous iteration, the following invariant for all  $i \in [1, t - 2]$  is guaranteed after execution of the loop beginning on Line 6:

$$\text{COST}([x_{i+2}, N], C_{i+2,k}) < \frac{1}{\lambda} \cdot \text{COST}([x_i, N], C_{i,k})$$

The previous inequality is guaranteed because if it were not true, then in Line 8,  $x_{i+1}$  would have been deleted.

Note that  $\text{COST}([x_{t-2}, N], C_{t-2,k}) > 0$ , for otherwise we conclude that  $\text{OPT}([x_{t-2}, N]) = \text{OPT}([x_t, N]) = 0$  so  $x_{t-1}$  would have been deleted. The value  $\rho_P$  is the minimum distance between points in the metric space, so  $\text{COST}([x_{t-2}, N], C_{t-2,k}) \geq \rho_P$ . By induction, we see that:

$$\text{COST}([x_{t-2}, N], C_{t-2,k}) < \left(\frac{1}{\lambda}\right)^{\frac{t-i}{2}-1} \cdot \text{COST}([x_i, N], C_{i,k})$$

where  $i = 2, 3$  depends on the parity of  $t$  (i.e. such that  $t - i$  is an even number). The bound  $\frac{t-i}{2} - 1 \geq t/2 - 3$ , and therefore:

$$\lambda^{(t/2)-3} \rho_P < \text{COST}([x_i, N], C_{i,k})$$

Next, we note by polynomial-boundedness that  $\text{OPT}([x_i, N]) \leq W^{O(c)} \cdot \rho_P$  because Line 10 guarantees that  $x_2$  (and thus  $x_i$  since  $i \geq 2$ ) is not expired and thus  $|[x_i, N]| \leq W$ . This is the only place where polynomial-boundedness is used. Next, by the approximation-ratio of the blackbox  $\alpha$ -approximation, we have that  $\text{COST}([x_i, N], C_{i,k}) \leq \alpha \cdot \text{OPT}([x_i, N])$ . Putting these together, we have:

$$\lambda^{(t/2)-3} \rho_P < \text{COST}([x_i, N], C_{i,k}) \leq \alpha \cdot \text{OPT}([x_i, N]) \leq \alpha \cdot W^{O(c)} \cdot \rho_P$$

Algebraic manipulation yields that  $t < 6 + 2 \log_\lambda(W^{O(c)} \cdot \rho_P)$ . Setting  $\lambda = 1 + \epsilon$ , we get that  $t = O(\epsilon^{-1} \log(W \cdot \rho_P))$ . ◀

Now we finish the proof of Theorem 3.1.

**Proof.** Proof of Theorem 3.1 We set  $\lambda = (1 + \epsilon)$  and let  $t = O(\epsilon^{-1} \cdot \log(W \cdot \rho_P))$ . Using Lemma 3.2, Algorithm 1 ( $k$ -MEDIAN) maintains an  $[\alpha(\alpha\lambda + 1), 2]$ -approximation set of  $k$  centers for  $P$  in the sliding windows model. For  $\lambda = (1 + \epsilon)$ , the approximation factor would be  $\alpha(\alpha(1 + \epsilon) + 1)$  which proves the theorem. ◀

Finally, we use the approximation algorithm of Theorem 3.1 to obtain the following result.

► **Corollary 3.4.** *Let  $\epsilon < 1/\alpha^2$  and  $0 < \delta < 1$ . Assume for Algorithm  $\mathcal{A}(P, k)$  in Theorem 3.1, we use the combination of algorithms [12] and [14] that guarantees 5.322-approximation to the  $\text{OPT}(P, k)$  with probability  $1 - \delta$  and uses space  $O(k^2 \epsilon^{-2} \log^9(n) \cdot \log(1/\delta))$ . Then, the sliding windows algorithm of Theorem 3.1 is an  $[35, 2]$ -approximation algorithm for the  $k$ -median problem with probability  $1 - \delta$  and uses  $O(k^2 \epsilon^{-3} \log^{10}(n) \cdot \log(1/\delta))$  space.*

**Proof.** As we mentioned in Section 1.2, we can plug the very recent 2.661-approximation algorithm for the  $k$ -median problem due to Byrka, Pensyl, Rybicki, Srinivasan, and Trinh [12] into  $(k, \epsilon)$ -coreset construction of Chen [14] to obtain  $O(k^2 \epsilon^{-2} \log^9(n) \cdot \log(1/\delta))$ -space 5.322-approximation algorithm in the insertion-only streaming model with probability  $1 - \delta$  for  $0 < \delta < 1$ . We use this algorithm as an  $\alpha$ -approximation algorithm  $\mathcal{A}$  in Theorem 3.1 and set  $\epsilon < 1/\alpha^2$  to have the approximation factor of

$$\alpha(\alpha(1 + \epsilon) + 1) = 5.322(5.322(1 + \frac{1}{(5.322)^2}) + 1) \leq 35 \ . \quad \blacktriangleleft$$

#### 4 Lower bound for $k$ -Median Clustering on Sliding Windows

In this section, we show that without the assumption of polynomial-boundedness, no randomized algorithm can approximate clustering in sub-linear space.

► **Theorem 4.1.** *Every randomized  $\alpha$ -approximation algorithm for 2-median on sliding windows requires  $\Omega(W)$ -storage.*

This result will be proved by reduction from the COUNT problem, which we now define:

► **Definition 4.2 (COUNT Problem).** Given a Boolean stream  $p_1p_2p_3\dots$  (i.e for  $p_i \in \{0, 1\}$ ) and a positive integer  $W$ , the COUNT problem on sliding windows maintains the number of elements equal to 1 in the most recent  $W$  elements.

The COUNT problem was explored by Datar, Gionis, Indyk, and Motwani [20], who developed a  $(1 + \epsilon)$ -estimator using  $O(\epsilon^{-1} \log^2 n)$  space. They also gave a matching lower bound of  $\Omega(\epsilon^{-1} \log^2 n)$  memory bits for any deterministic or randomized algorithm. In Theorem 4.3, we give a simple  $\Omega(W)$ -space lower bound for randomized algorithms that compute COUNT exactly. The proof is included here for the sake of completeness.

► **Theorem 4.3.** *Any randomized algorithm that exactly computes COUNT with probability  $2/3$  requires  $\Omega(W)$ -space.*

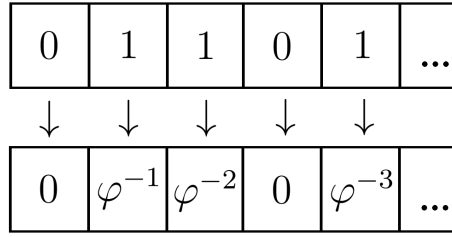
**Proof.** The INDEX problem in communication complexity [31] is the following problem. Let Alice and Bob be two players. Alice has a vector  $A \in \{0, 1\}^n$  with entries  $\{a_i\}_{i=1}^n$ . Bob has an index  $I \in [n]$ . Together they have to identify the value of  $a_I$  using minimum communication. It is well known [31] that the INDEX problem has a  $\Omega(n)$  lower bound in the one-way communication model, when Bob cannot send messages to Alice.

We provide the following simple reduction of the INDEX problem to the COUNT problem. Suppose there is a randomized streaming algorithm  $X$  that solves the COUNT problem w.p.  $2/3$  and uses  $w$  bits of memory. Alice computes  $X$  on windows of length  $n$  on stream  $a_1, a_2, \dots, a_n$  and sends the memory of  $X$  to Bob. Bob continues the computation on the stream of  $n$  zeros. Thus, the input that Alice and Bob collectively create is the stream of length  $2n$  with entries  $p_i = a_i$  for  $i \leq n$  and  $p_i = 0$  for  $n < i \leq 2n$ . Bob outputs  $(c_{n+I-1} - c_{n+I})$  where  $c_j = \sum_{l=j-n+1}^j p_l$  is the number of ones in the  $j$ -th window. Indeed

$$c_{n+I-1} - c_{n+I} = \sum_{l=I}^{n+I-1} p_l - \sum_{l=I+1}^{n+I} p_l = a_I - a_{I+n} = a_I.$$

Thus, the INDEX problem can be solved using  $w$  bits, and thus we must have  $w = \Omega(n)$ . ◀

The reduction from COUNT to 2-median will work by using Algorithm 2 to transform a stream of Boolean values into a stream of points in one-dimensional Euclidean space. The transformation will depend on the approximation guarantee  $\alpha$  of the 2-median algorithm. Algorithm 2 will maintain a counter  $j_N$ . It is clear that  $j_N - j_{N-W+1}$  is the exact solution of COUNT in the  $N^{\text{th}}$  window. The current counter  $j_N$  is known, but storing the entire history back to  $j_{N-W+1}$  would require  $W$  bits (since the window slides, we must keep the past  $W$  values). Instead, we use a 2-median approximation as a blackbox. The 2-median algorithm will output a set of centers, and we will prove that the location of the right-most center determines  $j_{N-W+1}$ . Thus we reduce the COUNT problem to computing 2-median approximately (to any degree of approximation, possibly dependent on  $W$ ).



■ **Figure 1** Example of Algorithm 2.

---

**Algorithm 2** Streaming Transformation for an  $\alpha$ -approximation.

---

```

 $\varphi \leftarrow 4\alpha$ 
 $j_1 \leftarrow 1$ 
for  $N \in \{1, \dots, n\}$  do
  if  $p_N = 0$  then
    Write  $x_N = 0$ 
     $j_{N+1} \leftarrow j_N$ 
  if  $p_N = 1$  then
    Write  $x_N = \varphi^{-j_N}$ 
     $j_{N+1} \leftarrow j_N + 1$ 

```

---

For each  $N \geq W$ , define  $i_N := j_{N-W+1}$ . Note that the  $N^{th}$  window is completely described by  $i_N$  and  $j_N$  (henceforth denoted  $i$  and  $j$ ). This set is  $\{\varphi^{-i}, \dots, \varphi^{-j+1}, 0, \dots, 0\}$ , and we assume that  $k \leq j - i < W$ . This assumption is valid since in the alternative cases (i.e., when  $j - i \in \{0, \dots, k - 1, W\}$ ), both COUNT and 2-median can be solved in  $O(k \log W)$ -space by keeping track of the indices of non-zero terms modulo  $W$ .

Before proving the desired reduction from COUNT to 2-median in Theorem 4.5, we begin with a lemma that is essential to the argument.

► **Lemma 4.4.** *The optimal cost for 2-median is strictly less than  $2\varphi^{-i-1}$ .*

**Proof.** Consider the center set  $\{0, \varphi^{-i}\}$ . All points  $x < \varphi^{-i}/2$  will be assigned to the center  $c = 0$ , and all points  $x > \varphi^{-i}/2$  will be assigned to the center  $c = \varphi^{-i}$ . Since  $\alpha \geq 1$ , we have that  $\varphi = 4\alpha \geq 4 > 2$  and thus all points in the window except  $x = \varphi^{-i}$  have the inequality  $x \leq \varphi^{-(i+1)} < \varphi^{-i}/2$ . This clustering assigns the point  $x = \varphi^{-i}$  to the center  $c = \varphi^{-i}$  and assigns all other points to the center  $c = 0$ . The cost of this clustering is:

$$\sum_{a=i+1}^j \varphi^{-a} = \frac{\varphi^{-i} - \varphi^{-j}}{\varphi - 1} < \frac{\varphi^{-i}}{\varphi - 1}$$

$\varphi > 2$  implies  $\varphi - 1 > \varphi/2$ , and thus this cost is strictly less than  $2\varphi^{-i-1}$ . The optimum cost is bounded above by the cost of this clustering, i.e.,  $OPT(P, 2) < 2\varphi^{-i-1}$ . ◀

► **Theorem 4.5.** *On sliding windows, any  $\alpha$ -approximation of 2-median can be used to compute the exact solution of COUNT.*

**Proof.** We will run the  $\alpha$ -approximation on the output of Algorithm 2 (with parameter  $\varphi = 4\alpha$ ) and obtain a set of 2 centers for each window. We will show that the right-most center is in the interval  $(\frac{1}{2}\varphi^{-i}, \frac{3}{2}\varphi^{-i})$ . Because  $\varphi > 3$ , these intervals are disjoint for distinct  $i$ , so the right-most center will identify a unique value of  $i$ . Since we have the counter  $j$  from Algorithm 2, we may output  $j - i$  as the exact solution to COUNT in that window.

All that remains to be shown is that the right-most center is in the desired interval. By Lemma 4.4, we have that  $OPT(P, 2) < 2\varphi^{-i-1}$ . Suppose that the right-most center  $c_2$  is not assigned any points in the clustering. Then the cost (using only the left-most center  $c_1$ ) is bounded below by the optimal cost of clustering the subset  $\{0, \varphi^{-i}\}$ , so  $COST(P, \{c_1\}) \geq OPT(\{0, \varphi^{-i}\}, 1) = \varphi^{-i}$ . The approximation guarantee implies that:

$$\alpha \geq \frac{COST(P, \{c_1\})}{OPT(P, 2)} > \frac{\varphi^{-i}}{2\varphi^{-i-1}} = \frac{\varphi}{2} = 2\alpha$$

By this contradiction, we conclude that both centers are used in the clustering, and in particular we conclude that the right-most point  $x = \varphi^{-i}$  is assigned to the right-most center.

Suppose that the right-most center  $c_2 \notin (\frac{1}{2}\varphi^{-i}, \frac{3}{2}\varphi^{-i})$ . Then the cost of clustering the right-most point  $x = \varphi^{-i}$  is at least  $\frac{1}{2}\varphi^{-i}$ . This implies that  $COST(P, \{c_1, c_2\}) \geq COST(\{\varphi^{-i}\}, \{c_2\}) \geq \frac{1}{2}\varphi^{-i}$  which leads to the following contradiction:

$$\alpha \geq \frac{COST(P, \{c_1, c_2\})}{OPT(P, 2)} > \frac{\varphi^{-i}/2}{2\varphi^{-i-1}} = \frac{\varphi}{4} = \alpha$$

Therefore the rightmost-center  $c_2$  is in the desired interval, and this completes the proof. ◀

The reduction of Theorem 4.5 together with the lower-bound of Theorem 4.3 implies the linear-space lower bound for 2-median stated in Theorem 4.1. We now give Theorem 4.6 which generalizes the reduction in two ways: (1) the result will hold for  $k \geq 2$ , and (2) the result will hold for clustering  $f(x, c) = d(x, c)^p$  for any  $p > 0$ . Theorem 4.5, pertaining to 2-median, is the special case of  $k = 2$  and  $p = 1$ . Notable special cases are  $k$ -median (when  $p = 1$ ) and  $k$ -mean (when  $p = 2$ ). The proof is a straight-forward generalization of the reduction for 2-median, which we briefly outline.

► **Theorem 4.6.** *For  $k \geq 2$ , every randomized  $\alpha$ -approximation algorithm that clusters  $f(x, c) = d(x, c)^p$  for  $p > 0$  on sliding windows requires  $\Omega(W)$ -storage.*

**Proof.** Run Algorithm 2 with  $\varphi = 2(2\alpha)^{1/p}$ . A straightforward modification of Lemma 4.4 when using the center-set  $\{0, \varphi^{-i-(k-2)}, \dots, \varphi^{-i}\}$  now reads  $OPT(P, k) < 2\varphi^{-(i-k+1)p}$ . Supposing that only  $k - 1$  centers were used in the clustering, the subset  $\{0, \varphi^{-i-(k-2)}, \dots, \varphi^{-i}\}$  shows that the cost is at least  $2(\varphi^{-i-(k-2)}/2)^p$ . These establish a contradiction showing that the right-most point  $\varphi^i$  must be assigned to the right-most center. We conclude by proving that the right-most center lies in the interval  $(\frac{1}{2}\varphi^{-i}, \frac{3}{2}\varphi^{-i})$  since otherwise the cost would be at least  $(\varphi^{-i}/2)^p$ , which again leads to the desired contradiction. ◀

Note that the transformed data was constructed in one-dimensional Euclidean space. This shows that without polynomially-boundedness, it is impossible to perform sublinear-space clustering on any Riemannian manifold, as shown in the next theorem.

► **Theorem 4.7.** *Let  $d$  be the metric of a Riemannian manifold  $M$ . For  $k \geq 2$ , every randomized  $\alpha$ -approximation algorithm that clusters  $f(x, c) = d(x, c)^p$  for  $p > 0$  on sliding windows requires  $\Omega(W)$ -storage.*

**Proof.** For  $\delta > 0$ , let  $\gamma : [0, \delta] \rightarrow M$  be a geodesic parameterized by arc-length [21]. The entire construction of Theorem 4.6 lies in the interval  $[0, \varphi^{-1}] \subset [0, 1]$ , so we modify Algorithm 2 to output the points  $\gamma(\delta\varphi^{-j})$ . The proof then carries through without modification, since  $d(\gamma(\delta\varphi^{-i}), \gamma(\delta\varphi^{-j})) = \delta|\varphi^{-i} - \varphi^{-j}|$  for all  $i, j \geq 1$ . ◀

**Acknowledgements.** The authors wish to thank the conference organizers for their hard work and the anonymous reviewers for their helpful feedback.

---

## References

---

- 1 P. K. Agarwal, S. Har-Peled, and K.R. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51(4):606–635, 2004.
- 2 N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.
- 3 P. Awasthi and N.F. Balcan. Center based clustering: A foundational perspective. In *Handbook of Cluster Analysis*. CRC Press, 2014.
- 4 B. Babcock, M. Datar, R. Motwani, and L. O’Callaghan. Maintaining variance and  $k$ -medians over data stream windows. In *Proceedings of the 9th ACM SIGMOD Symposium on Principles of Database Systems (PODS)*, pages 234–243, 2003.
- 5 P. Beame, R. Clifford, and W. Machmouchi. Element distinctness, frequency moments, and sliding windows. In *Proceedings of the 54th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 290–299, 2013.
- 6 J.L. Bentley and J.B. Saxe. Decomposable searching problems I: Static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.
- 7 V. Braverman, R. Gelles, and R. Ostrovsky. How to catch  $L_2$ -heavy-hitters on sliding windows. In *Proceedings of the 19th Computing and Combinatorics Conference*, pages 638–650, 2013.
- 8 V. Braverman and R. Ostrovsky. Smooth histograms for sliding windows. In *Proceedings of the 48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 283–293, 2007.
- 9 V. Braverman and R. Ostrovsky. Effective computations on sliding windows. *SIAM Journal on Computing*, 39(6):2113–2131, 2010.
- 10 V. Braverman, R. Ostrovsky, and A. Roytman. Zero-one laws for sliding windows and universal sketches. In *Proceedings of APPROX-RANDOM*, pages 573–590, 2015.
- 11 V. Braverman, R. Ostrovsky, and C. Zaniolo. Optimal sampling from sliding windows. *Journal of Computer and System Sciences*, 78(1):260–272, 2012.
- 12 J. Byrka, T. Pensyl, B. Rybicki, A. Srinivasan, and K. Trinh. An improved approximation for  $k$ -median, and positive correlation in budgeted optimization. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2015.
- 13 M. Charikar, L. O’Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 30–39, 2003.
- 14 K. Chen. On coresets for  $k$ -median and  $k$ -means clustering in metric and Euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009.
- 15 G. Cormode. The continuous distributed monitoring model. *SIGMOD Record*, 42(1):5–14, 2013.
- 16 G. Cormode and S. Muthukrishnan. What’s new: finding significant differences in network data streams. *IEEE/ACM Transactions on Networking*, 13(6):1219–1232, 2005.
- 17 Graham Cormode and Minos N. Garofalakis. Streaming in a connected world: querying and tracking distributed data streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1178–1181, 2007.
- 18 M. S. Crouch, A. McGregor, and D. Stubbs. Dynamic graphs in the sliding-window model. In *Proceedings of the 21st Annual European Symposium on Algorithms (ESA)*, pages 337–348, 2013.

- 19 A. Czumaj, C. Lammersen, M. Monemizadeh, and C. Sohler.  $(1 + \varepsilon)$ -approximation for facility location in data streams. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1710–1728, 2013.
- 20 M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. *SIAM Journal on Computing*, 31(6):1794–1813, 2002.
- 21 M. P. do Carmo. *Riemannian Geometry*. Birkhäuser, Boston, MA, 1992.
- 22 D. Feldman, A. Fiat, and M. Sharir. Coresets for weighted facilities and their applications. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 315–324, 2006.
- 23 D. Feldman, M. Monemizadeh, and C. Sohler. A PTAS for  $k$ -means clustering based on weak coresets. In *Proceedings of the 23rd Annual Symposium on Computational Geometry (SoCG)*, pages 11–18, 2007.
- 24 G. Frahling and C. Sohler. Coresets in dynamic geometric data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 209–217, 2005.
- 25 S. Guha, N. Mishra, R. Motwani, and L. O’Callaghan. Clustering data streams. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 359–366, 2000.
- 26 S. Har-Peled and A. Kushal. Smaller coresets for  $k$ -median and  $k$ -means clustering. In *Proceedings of the 21st Annual Symposium on Computational Geometry (SoCG)*, pages 126–134, 2005.
- 27 S. Har-Peled and S. Mazumdar. Coresets for  $k$ -means and  $k$ -median clustering and their applications. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 291–300, 2004.
- 28 P. Indyk. Algorithms for dynamic geometric problems over data streams. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 373–380, 2004.
- 29 P. Indyk and E. Price.  $k$ -median clustering, model-based compressive sensing, and sparse recovery for earth mover distance. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 627–636, 2011.
- 30 K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM*, 50(6):795–824, 2003.
- 31 E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.
- 32 S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.
- 33 S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.
- 34 M. Osborne, S. Moran, R. McCreadie, A. Von Lunen, M. Sykora, E. Cano, N. Ireson, C. MacDonald, I. Ounis, Y. He, T. Jackson, F. Ciravegna, and A. O’Brien. Real-time detection, tracking and monitoring of automatically discovered events in social media. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, USA, 2014.
- 35 J. J. Sylvester. A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*, 1:79, 1857.