Online k-means Clustering on Arbitrary Data Streams

Robi Bhattacharjee RCBHATTA@ENG.UCSD.EDU
UCSD

Jacob John Imola JIMOLA@ENG.UCSD.EDU
UCSD

Michal Moshkovitz MICHAL.MOSHKOVITZ@MAIL.HUJI.AC.IL
Tel-Aviv University

Sanjoy Dasgupta DASGUPTA@ENG.UCSD.EDU
UCSD

Editors: Shipra Agrawal and Francesco Orabona

Abstract

We consider k-means clustering in an online setting where each new data point is assigned to its closest cluster center and incurs a loss equal to the squared distance to that center, after which the algorithm is allowed to update its centers. The goal over a data stream X is to achieve a total loss that is not too much larger than $L(X, OPT_k)$, the best possible loss using k fixed centers in hindsight.

We start by introducing a data parameter, $\Lambda(X)$, such that for any online algorithm that maintains $O(k \operatorname{poly}(\log n))$ centers after seeing n points, there exists a data stream X for which a loss of $\Omega(\Lambda(X))$ is inevitable. Next, we give a randomized algorithm that achieves online loss $O(\Lambda(X) + L(X, OPT_k))$, while taking $O(k \operatorname{poly}(\log n))$ centers and additional memory. It has an update time of $O(k \operatorname{poly}(\log n))$ and is the first algorithm to achieve polynomial space and time complexity in the online setting.

We note that our results have implications to the related streaming setting, where one final clustering is outputted, and the no-substitution setting, where center selections are permanent. We show a general reduction between the no-substitution cost of a blackbox algorithm and its online cost. Finally, we translate our algorithm to the no-substitution setting and streaming settings, and it competes with and can outperform existing work in the areas.

1. Introduction

The *online learning* framework (Littlestone, 1987), first introduced in the context of classification, is a model that does away with the benign (i.i.d.) statistical assumptions that underlie much of learning theory, and instead deals with data that is arbitrary and possibly adversarial, and that arrives one point at a time, indefinitely.

Here we consider *clustering* in an online setting. At every time t, the Learner announces a clustering; then, Nature provides the next data point x_t; finally, the Learner incurs a loss depending on how well its clustering captures x_t. There are no assumptions on the data.

Specifically, we look at online realizations of *k-means clustering*. For any data stream $X = \{x_1, \ldots, \} \subset \mathbb{R}^d$, the k-means cost of a set of *centers* $S \subset \mathbb{R}^d$ is $\mathcal{L}(X, S) = \sum_{x \in X} d(x, S)^2$ where we define $d(x, S)$ to be the Euclidean distance from x to its nearest neighbor in S.

In the (batch) k-means problem, the input is a full data stream X, and the goal is to find a set of centers whose cost is close to that of the optimal k centers, denoted by $\mathcal{L}_k(X) = \inf_{|S|=k} \mathcal{L}(X, S)$.

© 2023 R. Bhattacharjee, J.J. Imola, M. Moshkovitz & S. Dasgupta.

Finding the optimal clustering is NP-hard (Aloise et al., 2009; Dasgupta, 2008), but a variety of constant-factor approximation algorithms are known (Kanungo et al., 2004; Ahmadian et al., 2020).

Batch k-means is the canonical method for *vector quantization*, in which training data X is clustered to obtain a set of k *codewords* S, and any subsequent data point x is quantized by replacing it by its nearest codeword $s \in S$, at a quantization cost of $d(x, s)^2$. It is well-known that if all points (past and future) come from some fixed distribution, and if $|X|$ is large enough, then good codewords for X are also good codewords for this underlying distribution Pollard (1981, 1982). We are interested in the more challenging setting of *lifelong learning*, where the data distribution can change and thus codewords need to be updated from time to time.

We study the *online k-means setting*, in which the Learner maintains a set of centers S_t that it updates as it sees more data. At each time t, three things occur.

- Nature provides a data point x_t.
- Learner incurs loss $d(x_t, S_{t-1})^2$.
- Learner announces an updated set of centers S_t.

The total loss incurred by the learner up to time t is then compared to the loss of the best k-means solution in hindsight; that is, $\mathcal{L}_k(\{x_1, \ldots, x_t\})$. A crucial difficulty is that the data x_t are arbitrary: Nature can choose a stream X with full knowledge of the Learner's algorithm.

There has been a large body of work on the different, but related, *streaming* k-means problem. In this setting, there is a finite data stream whose size n is typically known in advance. These data are revealed one point at a time and the learner updates its model after seeing each successive new data item. The key requirements are that the learner not use too much memory and that the individual updates be efficient. Once the entire data stream has been processed in this way, the cost of the Learner's final model is compared to that of the optimal k-means clustering.

In contrast, for the *online* k-means problem, losses accumulate along the way, and crucially, the loss of x_t depends on a clustering that was produced *before* x_t had been seen. The only work we are aware of in this framework is that of Cohen-Addad et al. (2021). They show how the classical multiplicative weights strategy (Littlestone and Warmuth, 1994) can be used in this setting, with each candidate clustering being an *expert*. This space of experts is continuous, but it can be discretized, and the authors show how to do this in a way that leads to a strong performance guarantee: the learner outputs exactly k centers at each time step and the total loss it accumulates at each time n is at most $(1 + \epsilon)\mathcal{L}_k(\{x_1, \ldots, x_n\})$. The downside, however, is that the algorithm requires resources (space and time) that are exponential in k and d, making it impractical in many settings.

In this paper, we are interested in developing *efficient* algorithms for online k-means clustering. Our solution strategy does not use multiplicative weights. Instead we rely upon three key ideas.

First, the adversarial nature of the data means that the *scale* of the clustering problem can increase dramatically from time to time, for instance if the latest point x_t is much farther away from the rest of the data than the typical previous interpoint distance. Between such scale-changes, however, it turns out that algorithmic ideas from the streaming k-means literature are applicable.

Second, we make use of the availability of good algorithms for the *streaming k-center* problem. For a data stream X and a set of centers S, the k-center cost of S is the maximum distance from a point in X to its closest center in S. The algorithm of Charikar et al. (1997) for streaming k-center takes one data point x_t at a time and updates its set of k centers in $O(kd)$ time. Its total space requirement is just $O(kd)$. And at any time n, this set of k centers has cost at most eight times that

of the best k-center solution for $X_n = \{x_1, \ldots, x_n\}$. This does not give us a solution for the k-means problem, since the k-means and k-center cost functions can differ by a multiplicative factor of $\Omega(n)$ for n data points. However, the k-center cost is useful for gauging when there has been a dramatic *scale-change*. We run the streaming k-center algorithm in the background, and whenever the k-center cost increases sharply, we think of a new *scale* as having begun.

Third, it is necessary to periodically throw away centers when we have accumulated too many of them. This is tricky because we must always ensure that the data points x_t close to those centers are still adequately covered. We introduce a novel way of handling this: we throw away all centers from before the previous scale began, and replace them by the k-center solution so far. The nature of scale-change means that quantization error can still be controlled. Our algorithm is shown in Algorithm 1, and its performance is governed by our main result.

Theorem 1 (Upper Bound) *Let X be an arbitrary data sequence, k be a positive integer, and δ satisfy $0 < \delta < 1$. Suppose we run $Online\_Cluster(X, k, \delta)$ (Algorithm 1). Let S_t denote the centers outputted at time t and $\mathcal{M}_t$ denote the total amount of memory used at the end of time t. Then with probability at least $1-\delta$ over the randomness of $Online\_Cluster$, for all integers $n \geq 2$, the following hold:*

1. *(Approximation Factor) $\sum_{t=2}^{n} d(x_t, S_{t-1})^2 = O(\mathcal{L}_k(X_n) + \Lambda(X_n))$.*
2. *(Center Complexity) $|S_n| = O(k \log^6 \frac{n}{\delta})$.*
3. *(Memory and Time Complexity) Each step uses $O(kd \log^6 \frac{n}{\delta})$ time and memory.*

Here, X_n is a shorthand for the sequence $x_1, \ldots, x_n$, $\mathcal{L}_k(X_n)$ is the optimal k-means cost in hindsight, and the final term is

$$\Lambda(X_n) = \sum_i d(x_i, X_{i-1})^2.$$

The last term, $\Lambda(X_n)$, can be seen as the loss we would incur, in the online setting, if we were allowed to store *all* points seen so far. We complement this with a lower bound demonstrating a broad class of data sequences for which at least $\Omega(\Lambda(X_n))$ loss must be paid over the first n points.

Theorem 2 (Lower Bound) *Let X be any data sequence that contains infinitely many distinct points. Let A be an online clustering algorithm such that its output satisfies $|S_n| \leq n$ for all n and for all input sequences. Then there exists a sequence $\tilde{X} = \tilde{x}_1, \tilde{x}_2, \ldots$ such that the following conditions hold.*

1. *$\tilde{X}$ is drawn from the closure of X, (i.e. X and its limit points). Thus all points in $\tilde{X}$ are arbitrarily close to points in X.*
2. *For all $n \geq 2$, the expected loss over A satisfies $\mathbb{E}_A \left[\sum_{s=2}^{n} d(\tilde{x}_s, S_{s-1})^2\right] \geq \Omega(\Lambda(\tilde{X}_n))$.*

Our lower bound does not construct a fixed sequence for which *all* algorithms incur a large loss, for the simple reason that an algorithm may memorize an arbitrary number of the points in the sequence. Thus the sequences achieving the lower bound do depend on the algorithm, but they are not pathological in the sense that they can be constructed from any sequence X with infinitely many distinct points.

1.1. Connections to Other Clustering Settings

A key step in analyzing our algorithm is centered around reducing the online loss, $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$, to the closely related *no-substitution loss*, $\sum_{t=1}^{n} d(x_t, S_t)^2$. This loss function comes from the no-substitution setting (Liberty et al. (2016); Moshkovitz (2021)), which is a variant of the online setting in which the centers are allowed to be updated *after* observing the latest point x_t. For this reason, the loss paid is with respect to S_t rather than S_{t-1}. In exchange for this advantage, the no-substitution setting severely limits the types of updates that can be made to S_t; at each time we can either include the newest point as a center, or reject it forever, with all decisions being final. More precisely, for all times t, either $S_t = S_{t-1}$ or $S_t = S_{t-1} \cup \{x_t\}$. No-substitution algorithms balance both of these criteria by minimizing their loss function without selecting too many centers.

It turns out that for *any* no-substitution algorithm, its loss in the online setting, $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$, can be bounded by its no-substitution loss, $\sum_{t=1}^{n} d(x_t, S_t)^2$ and the lower bound parameter, $\Lambda(X_n)$. We express this in the following reduction result, which, in addition to being a key step of proving Theorem 1, is also of independent interest.

Theorem 3 (Reduction to No-substitution) *Let S_t denote the selected centers at the end of time t of any no-substitution algorithm. Then at all times n, the online clustering loss, $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$ can be bounded by a constant factor of the no-substitution loss, $\sum_{t=1}^{n} d(x_t, S_{t-1})^2$ and a corrective term $\Lambda(X_n)$. That is,*

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2 \leq 6 \sum_{t=1}^{n} d(x_t, S_t)^2 + 6\Lambda(X_n)$$

It is important to note that although this result doesn't directly apply to our algorithm (as it is *not* a no-substitution algorithm as it periodically removes centers), a close variant of this result does. A simple intuition for this is that our algorithm mimics no-substitution algorithms sufficiently well to have a similar relation between its online loss and its no-substitution loss.

Performance in other settings: Finally, although our paper is focused on the online clustering setting, it is natural to consider our algorithm's performance in the other clustering settings, such as the streaming setting and the no-substitution setting. To this end, we show that our algorithm gives a nearly identical performance in the streaming setting, with the streaming loss being reduced to $O(\mathcal{L}_k(X_n))$ and the center, memory, and time, complexity remaining the same.

Corollary 4 *With probability $1 - \delta$ over its randomness, $Online\_Cluster(X, k, \delta)$ has streaming loss, $\sum_{t=1}^{n} d(x_t, S_n)^2 \leq 33\mathcal{L}_k(X_n)$, and has center, memory, and time complexity as bounded in Theorem 1.*

Although our algorithm cannot be directly applied to the no-substitution setting as it deletes centers (thus violating the no-deletion policy), it can easily be transformed into a no-substitution setting by simply removing all deletions, and forcing the algorithm to adhere to the rule that $S_t = S_{t-1}$ or $S_t = S_{t-1} \cup \{x_t\}$. After doing this, the resulting algorithm, denoted $No\_Sub\_Cluster(X, k, \delta)$, has performance as follows.

Corollary 5 *Let X be an arbitrary data sequence, k be a positive integer, and δ satisfy $0 < \delta < 1$. Suppose we run $No\_Sub\_Cluster(X, k, \delta)$. Let S_t denote the centers outputted at time t and $\mathcal{M}_t$ denote the total amount of memory used at the end of time t. Then with probability at least $1 - \delta$ over the randomness of $No\_Sub\_Cluster$, for all integers $n \geq 1$, the following hold:*

1. *(Approximation Factor)* $\sum_{t=2}^{n} d(x_t, S_t)^2 = O\left(\mathcal{L}_k(X_n)\right)$.
2. *(Center Complexity)* $|S_n| = O(kOC_{k+1}(X_n)\log^6 \frac{n}{\delta})$, *where* $Online\_Cluster_k(X_n)$ *is the lower bound parameter introduced in Bhattacharjee and Moshkovitz (2021).*
3. *(Memory and Time Complexity) Each step uses* $O(kdOC_{k+1}\log^6 \frac{n}{\delta})$ *time and memory.*

The term, OC_{k+1}, is a lower bound term introduced in Bhattacharjee and Moshkovitz (2021) that characterizes the necessary center complexity for no substitution clustering. The resulting center complexity of our algorithm is comparable to that of other algorithms in this setting (such as Liberty et al. (2016) or Bhattacharjee and Moshkovitz (2021)), and the no-substitution loss guarantees is an improvement over all known algorithms. We include a more detailed comparison between our algorithm and existing work in Appendix C.2.

2. Related Work

In the offline (batch) k-means setting, all points are given simultaneously, and the goal is to find a small subset of centers with a small approximation compared to the optimal k-means clustering. Efficient algorithms returning $poly(k)$ centers with a constant approximation are known (Kanungo et al., 2004; Aggarwal et al., 2009; Ahmadian et al., 2020). On the negative side, it is NP-hard to return the optimal k centers or even approximate it up to a small constant (Aloise et al., 2009; Dasgupta, 2008).

In the online setting, points arrive one after another and not simultaneously. After each point, the algorithm decides whether to take this point as a center. In Cohen-Addad et al. (2021) algorithms for a similar setting as ours were proposed. The algorithms are inefficient and run in exponential time in k and the dimension of the data. On the other hand, our algorithm runs in polynomial time.

Recently a popular variant of the online setting was explored, the no-substitution setting. In this setting, decisions are irrevocable, and the loss is paid *after* the centers are updated (Liberty et al. (2016)). Additionally, many other papers also consider the case where the cost is measured only at the end (thus mimicking the streaming setting) (Hess and Sabato, 2020; Moshkovitz, 2021; Bhattacharjee and Moshkovitz, 2021; Hess et al., 2021). In Liberty et al. (2016) an algorithm utilizing ideas from Meyerson (2001) was introduced. Unfortunately, the number of centers inherently depends on the aspect ratio, which can be enormous. In this paper, the cost is incurred immediately after each point arrives. The number of centers our algorithm uses is only $k\log^{O(1)}(n)$. No assumptions on the input data are needed.

A closely related setting is the streaming model (Aggarwal, 2007; Guha et al., 2003; Braverman et al., 2011; Shindler et al., 2011; Ailon et al., 2009). As in the online setting, points arrive one after another, and the goal is the utilize a small memory while ensuring that the cost at the end is small. Several passes over the data are allowed. In this paper, only one pass on the data is allowed, and more importantly, the cost is incurred immediately and not at the end of the stream.

One other related setting is the k-servers problem (Koutsoupias, 2009). Here, at all times k points (i.e. servers) are maintained in memory, and the loss incurred is the total distance that the points are moved to include the newest point. Crucially, the learner is charged a penalty for moving *any* of the k points, and is not allowed to add or delete points (unlike our setting).

3. Preliminaries

Notation For the rest of this paper, we prefer referring to data streams as data sequences to emphasize their possibly infinite size. For a data sequence $X \subseteq \mathbb{R}^d$, the k-means cost of a set of centers $S \subseteq \mathbb{R}^d$ is given by $\mathcal{L}(X, S) = \sum_{x \in X} \min_{s \in S} d(x, s)^2$, where d denotes the Euclidean distance. Additionally, we let $\mathcal{L}_k(X) = \min_{|S|=k} \mathcal{L}(X, S)$ denote the optimal k-means cost of clustering data sequence X.

For a positive integer t, we write $X_t = \{x_1, \ldots, x_t\}$ to denote the first t elements of X. We also let $\mathcal{L}(X)$ and $\mathcal{L}(X, x)$ denote $\mathcal{L}_1(X)$ and $\mathcal{L}(X, \{x\})$ respectively.

Setting Our input is an infinite sequence $X = \{x_1, \ldots\} \subset \mathbb{R}^d$ which is given to the algorithm one point at a time. At each time t, the algorithm observes a new point x_t and then outputs a set of cluster centers, S_t. The algorithm incurs loss at x_t based on how well the *previous* clustering, S_{t-1}, captures x_t. Thus, the total loss of the algorithm is

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2.$$

Here, the index begins at $t = 2$ because the algorithm is allowed to see the initial point, x_1 without incurring a loss.

There are no restrictions on the sequence X. It can even be chosen by an adversary that has knowledge of our algorithm ahead of time. The only restriction is that X cannot be changed *after* observing the output of our algorithm, S_t – the adversary (or nature) must decide on X before the algorithm starts running.

4. The Lower Bound Parameter, Λ

Typically, the goal of most online or streaming clustering problems is to achieve a loss at time n on the order of $O(\mathcal{L}_k(X_n))$, where k is a pre-specified parameter denoting the optimal number of centers. However, in this setting, this is *not* always possible.

Consider the data sequence $\{1, \alpha, \alpha^2, \alpha^3, \ldots\} \subset \mathbb{R}$ for some constant $\alpha > 1$. For α sufficiently large, the point α^n will be extremely far away from all the points preceding it, and consequently will be very likely to incur a large loss, $d(x_n, S_{n-1})^2$. The only way $d(x_n, S_{n-1})^2$ will be small is if the algorithm is somehow able to "guess" its location. By contrast, the optimal k-means clustering of $\{1, \ldots, \alpha^n\}$ will include some cluster center that is reasonably close to α^n, and as a result will incur a significantly smaller loss.

To generalize this example to arbitrary data sequences, the key insight is that the distance from a given point to the set of points preceding it serves as a baseline for its incurred loss unless the algorithm makes a lucky guess. This leads us to our lower bound parameter.

Definition 6 (Lower Bound Parameter) *For an ordered sequence of points $X_n = \{x_1, x_2, \ldots, x_n\}$, define $\Lambda(X_n) = \sum_{t=2}^{n} d(x_t, X_{t-1})^2$.*

The quantity $\Lambda(X_n)$ can be interpreted as the loss incurred by an online algorithm whose cluster centers at any time t consist of *all* points seen so far. One can view this as the best possible algorithm that makes no guesses about locations of future points.

We now formalize the intuition that $\Lambda(X_n)$ is a lower bound on the online loss at time n. With no assumptions, the algorithm may make an arbitrary number of guesses about the data sequence, or even memorize X_n, and defeat the lower bound of $\Lambda(X_n)$. To control this behavior, we assume that at time n, the number of centers outputted by the algorithm is bounded by an integer, b_n.

The most standard way to prove a lower bound would be to show that for any algorithm A, there exists a data sequence X for which A pays loss bounded by $\Omega(\Lambda(X_n))$ at time n. However, the lower bound would then only be tight for pathological choices of X_n (such as $\{1, \alpha, \alpha^2, \dots\}$). Instead, we show a stronger result— that for any algorithm A and data sequence X, there exists a data sequence $\tilde{X}$ that can be constructed from X for which A pays loss at least $\Omega(\Lambda(\tilde{X}_n))$. The strength of this stricter approach is that even for extremely limited sets of data sequences (say sets where "pathological" examples are excluded), our lower bound $\Lambda(X_n)$ maintains relevance. Our lower bound appears in Theorem 2, and is proved in section A.

5. A No-Substitution Approach to Online Clustering

We now turn our attention towards finding efficient algorithms for the online clustering setting. A natural starting point is to consider the vast literature of streaming k-means algorithms. Recall that streaming algorithms maintain an output S_t for which the loss $\mathcal{L}_k(X_t, S_t)$ is small (typically $O(\mathcal{L}_k(X_t))$. We might conjecture that such algorithms have an online loss bounded by $O(\Lambda(X_n) + \mathcal{L}_k(X_n))$ for the following reason. When x_t is far from previously encountered points, the incurred loss $d(x_t, S_{t-1})^2$ can be absorbed by $\Lambda(X_n)$, and when x_t is near previous data, the maintained set of centers S_{t-1} serves as a good representation of all data including x_t. Unfortunately, the following example illustrates that that this conjecture is not true for all streaming algorithms, highlighting the need for a more sophisticated approach.

Let X be a sequence that cycles through a set of $k+1$ points with pairwise distances all equal to 1 embedded in $\mathbb{R}^k$. Let A be the algorithm that always outputs the last k points that it has encountered, that is $S_t = \{x_t, x_{t-1}, \dots, x_{t-k+1}\}$. A achieves a good streaming loss over X — at the end of any time n, A clearly outputs a 2-approximation to the optimal k-means loss, and consequently achieves a low streaming loss.

Conversely, A does poorly on X in the online setting. On each subsequent point, A pays a loss of exactly 1, meaning that our online loss at time n is $\Omega(n)$. By contrast, $\Lambda(X_n) = k$, as we only have $k+1$ distinct points, and $\mathcal{L}_k(X_n) \leq \frac{n}{2(k+1)}$. It follows that A pays online loss that is a factor of $\Omega(k)$ larger than the combination of $\Lambda(X_n)$ and $\mathcal{L}_k(X_n)$.

This example highlights that more structure is needed on the set of centers S_t beyond simply having a low streaming loss. In the example above, at every single time, A removes from S precisely the new point x_t, and thus incurs loss. To circumvent this issue, a natural idea is to consider clustering algorithms that are unlikely to remove points from S_t. This idea is the defining characteristic of the *no-substitution* setting.

5.1. A Reduction to the No-Substitution Setting

Recall that no-substitution clustering can be thought of as a variant of the online clustering setting in which two things are changed. First, rather than incur loss before updating our centers, this setting reverses the order, and the algorithm incurs loss *after* it has a chance to update its centers. We can thus write the cumulative loss at time n as $\sum_{t=1}^{n} d(x_t, S_t)^2$.

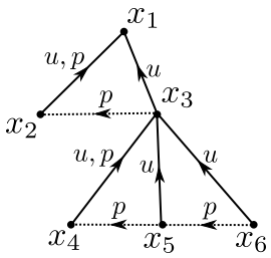

Figure 1: An illustration depicting $u(t), p(t)$ on a sample dataset.

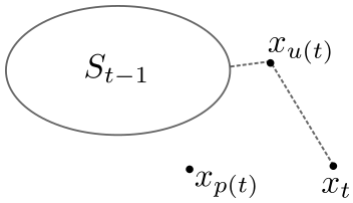

Figure 2: A naive application of the triangle inequality, which results in a $d(x_{u(t)}, S_{t-1})^2$ term (see (1)). This produces too many terms involving $x_{u(t)}$.

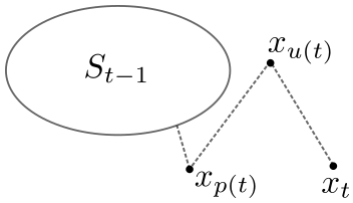

Figure 3: A double application of the triangle inequality produces $d(x_{p(t)}, S_{t-1})^2$ and $d(x_{p(t)}, x_{u(t)})^2$ terms, which involves each point a constant number of times (see (2)).

This setting is clearly trivial if no restrictions are placed on S_t, as otherwise can simply set $S_t = \{x_t\}$ thus having perfect loss and using only 1 center at all times. This leads to the second difference from the online setting; once a center is chosen, it can *never* be removed. That is, $S_1 \subseteq S_2 \subseteq \ldots$. This restriction prevents the trivial solution given above, as although it still obtains 0 loss, it now has unacceptable center complexity as it would require $|S_t| = t$ for all times t.

This restriction of no deletions precisely circumvents the counterexample of the previous section. Using this idea, we are able to prove Theorem 3, which reduces the online clustering loss to the loss of any no-substitution algorithm.

Before going to the proof, we introduce a special way of indexing that will make our inequalities more intuitive. Consider the following functions:

Definition 7 *Let X_n be a data sequence. For $t \geq 2$, define the previous nearest neighbor of the point with index t by $u(t) = \arg\min_{i=1,\ldots,t-1} d(x_i, x_t)$; i.e. the index i such that $x_i \in X_{t-1}$ is the closest to x_t. Consider the tree induced by $u(t)$ where the parent of t is given by $u(t)$. Denote the previous sibling in the tree of the point with index t as $p(t) : \mathbb{N} \to \mathbb{N}$. In other words, $p(t)$ is the greatest index s such that $x_s \in X_{t-1}$ satisfies $u(s) = u(t)$—i.e. s is the greatest sibling less than t. If t has no previous siblings, then set $p(t) = u(t)$.*

For an illustration of these functions, see Figure 1. Notice that $\Lambda(X) = \sum_{t=2}^{n} d(x_t, x_{u(t)})^2$. It is easy to see that both $p(t) < t$ and $u(t) < t$ for all t. Furthermore, for any index s, there can be at most two distinct indices t, t' such that $p(t) = p(t') = s$, namely, the very next sibling of s and the smallest child of s. We call this property 2-injectivity of $p(t)$. We are now prove Theorem 3.

Proof (Of Theorem 3): Our goal is to upper bound $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$. Let l be such that $\tau_{l+1} \leq n < \tau_{l+2}$. We begin with a straightforward application of the triangle inequality,

$$\begin{aligned} d(x_t, S_{t-1}) &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, S_{t-1}) \\ &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, S_{u(t)}) + d(x_{p(t)}, S_{u(t)}) \end{aligned} \tag{1}$$

with the last inequality holding because $S_{p(t)} \subset S_{t-1}$ since the algorithm is in the no-substitution setting. Each $d(x_t, x_{u(t)})$ term is how far x_t is from all other points. Each $d(x_{u(t)}, S_{u(t)})$ term represents how well $S_{u(t)}$ represents the points $x_{u(t)}$, which is simply the part of X_{t-1} that is closest

to x_t. Squaring and summing (1) over all times t, we could obtain a bound for the online loss in terms of how far new points are from previous points, plus how well S_t represents X_t over time.

However, examining (1), a problematic term is $\sum_{t=2}^{n} d(x_{u(t)}, S_{u(t)})^2$, which results in a sum of $d(x_u, S_u)^2$ for all t such that $u(t) = u$. Since $u(t)$ need not be injective, this could produce n copies of $d(x_u, S_u)^2$. To circumvent this problem, we apply the triangle inequality twice and involve $p(t)$. This will let us leverage the fact that for any s, there are at most two distinct indices t, t' for which $p(t) = p(t') = s$. We have,

$$\begin{aligned} d(x_t, S_{t-1}) &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, x_{p(t)}) + d(x_{p(t)}, S_{t-1}) \\ &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, x_{p(t)}) + d(x_{p(t)}, S_{p(t)}). \end{aligned} \tag{2}$$

Our first, naive application of the triangle inequality appears in Figure 2 and our double application appears in Figure 3. This double application results in terms of the form $d(x_{p(t)}, S_{p(t)})^2$, which when summed over all points, can be upper bounded by the new loss function $\sum_{t=1}^{n} d(x_t, S_t)^2$ because p is 2-injective.

By squaring (2) and applying Cauchy Schwartz, we have

$$d(x_t, S_{t-1})^2 \leq 3d(x_t, x_{u(t)})^2 + 3d(x_{u(t)}, x_{p(t)})^2 + 3d(x_{p(t)}, S_{p(t)})^2.$$

Substituting this, we form an upper bound on the loss as follows:

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2 \leq \underbrace{3\sum_{t=2}^{n} d(x_t, x_{u(t)})^2}_{=3\Lambda(X_n)} + \underbrace{3\sum_{t=2}^{n} d(x_{u(t)}, x_{p(t)})^2}_{\leq 3\Lambda(X_n)} + \underbrace{3\sum_{t=2}^{n} d(x_{p(t)}, S_{p(t)})^2}_{\leq 6\sum_{t=1}^{n} d(x_t, S_t)^2}.$$

Here, the first equality holds by definition, the second because $u(t)$ is the nearest neighbor of $p(t)$ in $X_{p(t)-1}$ (otherwise $p(t) = u(t)$), and the third holds from the 2-injectivity of $p(t)$. Substituting these gives the desired bound, $\sum_{t=2}^{n} d(x_s, S_{t-1})^2 \leq 6\Lambda(X_n) + 6\sum_{t=1}^{n} d(x_t, S_t)^2$. ■

6. An Online Clustering Algorithm

Unfortunately, directly applying a no-substitution algorithm to the online clustering setting is not sufficient. First, although Theorem 3 implies that they would achieve a cost of $O(\Lambda(X_n)) + O(\sum_{t=1}^{n} d(x_t, S_t)^2)$, it turns out that most existing no-substitution algorithms give a slightly weaker guarantee where they instead show that for all times n, $\sum_{t=1}^{n} d(x_t, S_n)^2 = O(\mathcal{L}_k(X_n))$. Here, note that the centers at S_n are used in retrospect for all $1 \leq t \leq n$ rather than directly using S_t. The only algorithm that directly bounds $\sum_{t=1}^{n} d(x_t, S_t)^2$ is that of Liberty, Sriharsha, and Sviridenko Liberty et al. (2016), but their bound includes a log factor in the approximation factor as they show that $\sum_{t=1}^{n} d(x_t, S_t)^2 \leq O(\log n \mathcal{L}_k(X_n))$.

Second, and perhaps more importantly, all no-substitution algorithms potentially incur significantly more than $\text{poly}(\log n, k)$ centers at time n. As a simple example, observe that for the sequence considered earlier (Section 4) $\{1, \alpha, \alpha, \alpha^2, \dots\}$, any no-substitution algorithm will be forced to select every point, as failing to select a point would incur a very large clustering cost at that time. This idea is formalized in Bhattacharjee and Moshkovitz (2021), which introduces a lower bound

parameter, $\mathrm{OC}(X_n)$, that gives a lower bound on the number of centers any no-substitution algorithm must select to achieve loss $O(\mathcal{L}_k(X_n))$. Although selecting $\mathrm{OC}(X_n)$ centers is acceptable (and unavoidable) in the no-substitution setting, this is far from desirable in the online setting as $\mathrm{OC}(X_n)$ can be potentially as large as $\Omega(n)$. To address these issues, we propose a new algorithm.

6.1. Our Algorithm

Algorithm 1: The main algorithm, $Online\_Cluster(X, k, \delta)$.

$S_k \leftarrow \{x_1, \ldots, x_k\}$ `Initial set of centers`
$R_k, F \leftarrow 0$
$\tau_1 \leftarrow k+1, i \leftarrow 2$
$(Z_k, w_k) \leftarrow online\_k\_centers\_update(X_k)$
for $t = k+1, k+2, \ldots$ **do**
  $(Z_t, w_t) \leftarrow online\_k\_centers\_update(x_t)$ `Scale approximation`
  $time(z) = t$ for $z \in Z_t$
  if $w_t > 16 w_{\tau_{i-1}} \sqrt{t}$ **then** `Scale change detected`
    $\tau_i \leftarrow t$
    $S_t \leftarrow S_{t-1} \setminus \{x \in S_{t-1} : time(x) \leq \tau_{i-1}\}$ `Remove old centers from` S
    $S_t \leftarrow S_t \cup Z_{\tau_{i-1}}$ `Replace with` k`-centers`
    $R_t \leftarrow \frac{w_t^2}{128k}, F \leftarrow 0$
    $i \leftarrow i+1$
  else
    $S_t \leftarrow S_{t-1}, R_t \leftarrow R_{t-1}$
  With probability $\min\left\{1, \frac{d(x_t, S_t)^2 \log^4 \frac{2t}{\delta}}{R_t}\right\}$, $S_t = S_t \cup \{x_t\}$ `Center selection`
  $F \leftarrow F + \mathbf{1}_{x_t \text{ is selected}}$
  if $F > 25k \log^5 \frac{2t}{\delta}$ **then**
    $R_t \leftarrow 2R_t, F \leftarrow 0$
end

Online_Cluster, with probability $1-\delta$, achieves online loss $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$ bounded by a combination of $\Lambda(X_n)$ and $\mathcal{L}_k(X_n)$ for all times $n \geq 2$. The pseudocode for the algorithm appears in Algorithm 1. On a high level, the algorithm adds centers to S probabilistically and removes older centers when it detects that the scale of the data changes. Specifically, this process involves three ideas: scale approximation, center deletion, and center selection. We now outline these ideas.

Scale Approximation: Online_Cluster approximates the scale of the data, or how spread out the data is, by running a k-centers algorithm in the background at all times. Recall that for a data sequence X and a set of centers S, the k-center cost of S is $\max_{x \in X} d(x, S)$. The algorithm of Charikar et al. (1997) is an online k-center algorithm that at any time n outputs a set of k centers $Z_n = \{z_n^1, z_n^2, \ldots, z_n^k\}$ with k-centers loss at most 8 times the optimal loss. Furthermore, this algorithm enjoys space and time complexity $O(k)$. While it may seem odd to use a streaming k-centers algorithm as opposed to k-means, we use k-centers because unconditional approximation

algorithms exist. We will let w_n denote loss outputted by Charikar et al. (1997)'s algorithm when applied to $x_1, \ldots, x_n$.

Scale approximation using k-centers is useful for two reasons. First, it enables us to approximate the k-means cluster cost $\mathcal{L}_k(X_n)$ at time n up to a factor of $O(n)$. In particular, the k-centers clustering outputted by Charikar et al. (1997)'s algorithm can be simply applied as a k-means clustering, with the k-means loss being bounded by noting that each point incurs loss at most w_n^2. Formally, we have the following (proved in Appendix B):

Proposition 8 *For all data sequences X and all n, $\frac{w_n^2}{128} \leq \mathcal{L}_k(X_n) \leq n w_n^2$.*

We will see in the later that despite w_n^2 being a loose approximation for $\mathcal{L}_k(X_n)$ (with a gap of up to $O(n)$), it can nevertheless be used to set the center selection rate and prevent too many centers from being selected. Additionally, scale approximation enables the algorithm to remove selected centers from smaller scales, which brings us to our next key idea.

Removing centers: As we observed earlier, removing centers is essential to avoid the lower bounds that the no-substitution setting faces. To this end, Online_Cluster uses scale *increases* to decide when to remove centers it has previously selected. The key insight is that when the scale, tracked by w_n, drastically increases, we have that all previous points can be clustered in "relatively small" clusters. Because of this, clustering the previous points using their *k-centers approximation* provides a sufficient summary. Although the k-centers clustering can incur k-means loss up to nw_{n-1}, the nature of the scale increase implies that even this total cost is still small compared to the k-means cost at time n. We will refer to times during which these large scale increases occur as *scale changes*, and denote them as $\tau_1, \tau_2, \ldots$. They have the following formal definition.

Definition 9 (scale changes) *Let $\tau_1 = k + 1$, and let $\tau_i = \min\{t : t > \tau_{i-1}, w_t > 16\sqrt{t} w_{\tau_{i-1}}\}$. If no such τ_i exists, then we set $\tau_i = \infty$ and terminate the sequence. Each τ_i is referred to as a* ***scale change****.*

When a scale change τ_i is detected, the algorithm is able to replace all selected centers x that were streamed as inputs before τ_{i-1} with the set of k centers from that time, denoted by $Z_{\tau_{i-1}}$. To this end, we implicitly assume that the algorithm timestamps each point it selects as a center; this costs a trivial amount of additional time and memory. After removing the desired centers, the only points that will remain in memory are those centers taken after time τ_{i-1} and $Z_{\tau_{i-1}}$. This prevents the number of selected centers from accumulating over increasing scales.

Center Selection: The algorithm selects centers using existing ideas from the streaming setting with one important change. Our method resembles that of Liberty et al. (2016) which is the following: for each subsequent point, select it with probability $O(\frac{d(x_t, S_{t-1})^2}{R_t})$, where R_t is a dynamically adjusted parameter governing the rate of selections. Use a counter F to keep track of the number of centers selected since R_t was previously changed. When $F \geq O(k \log t)$, this indicates that the value of R_t is too small, and consequently double R_t so as to discourage further center selection. When R_t reaches a value of $O(\frac{\mathcal{L}_k(X_t)}{k})$, we can prove the desired center complexity and approximation ratio bounds.

Unfortunately, the previous method is incapable of dealing with data that exhibits many scale changes. Consider again the data sequence $\{1, \alpha, \alpha^2, \ldots\}$. In this sequence, naively applying the

above center selection criteria would result in every point being selected. This is because each point x_t is far from *all* previous points, so necessarily $d(x_t, S_{t-1})^2 > R_t$ is true, resulting in the point being taken. Thus, algorithms using this method (e.g., (Liberty et al., 2016; Bhaskara and Rwanpathirana, 2020)) have center complexities depending on the *aspect ratio*, the ratio between the distances between the furthest two and closest two points in the input sequence. Although subsequent work (Bhattacharjee and Moshkovitz, 2021) has managed to reduce the dependence on the aspect ratio to a dependence on a tighter lower bound parameter, OC, it still remains the case that data sequences with many scale changes are provably difficult in the no-substitution setting, and are unable to be effectively clustered by a direct application of no-substitution clustering.

Our important change to overcome the above issue is to use w_t to track the scale. Specifically, during a scale change, we set $R_t = \frac{w_t^2}{128k}$. By Proposition 8, this guarantees that the current value of R_t is at most a factor of $O(n)$ from the optimal value of $\frac{\mathcal{L}_k(X_t)}{k}$. While $O(n)$ seems like a relatively poor approximation, only $O(\log n)$ doublings of R_t are required to increase it to the optimal value, and just poly $\log n$ centers are taken to do this.

Putting it all together: Combining our three main ideas, our algorithm consists of the following:

At the start, in lines 1- 4, initialize S_k, R_k, w_k, τ_1 by selecting the first k points and considering the $k+1$th point as the first scale change. Also, initialize the k-centers algorithm, which we assume can be given a set of points with the method $online\_k\_centers\_update(x)$ and will return (Z, w): the centers and the cost of the k-centers clustering it has computed on all points it has seen so far.

Each time a new point is encountered, update the *scale approximation* (line 6) and decide if the new point produces a *scale change* (line 8). If a scale change is detected, then *remove centers* that were streamed before the previous scale change and replace them with their corresponding k-centers summary (lines 10,11). In the algorithm, we assume that every point streamed has a timestamp that can be accessed with a function denoted as $time$. We emphasize that when we add $Z_{\tau_{i-1}}$ to S_t, the timestamps of the points in $Z_{\tau_{i-1}}$ are from before τ_{i-1}, and thus they will be removed when the next scale change happens. Furthermore, during a scale change we reset the values of R_t and F (line 12) to keep these parameters updated for the new scale.

Finally, perform *center selection* using the parameters R_t, F in lines 16-19.

While the algorithm appears to require us to remember the parameters R_t, Z_t, w_t, S_t for all t and τ_i for all i, we may implement it by simply remembering R_t, w_t, S_t at the most recent time, as well as τ_{i-1}, τ_i, and $Z_{\tau_{i-1}}$ where i is the index of the most recent scale change. This allows us to achieve the desired memory bounds.

7. Analysis of Algorithm 1

The performance of Algorithm 1 is given in Theorem 1, with its performance in the streaming and no-substitution settings given in Corollaries 4 and 5. We defer a discussion of these alternate settings to Appendix C, and focus on Theorem 1.

Observe that the approximation factor is bounded both in terms of $\mathcal{L}_k(X_n)$, the optimal k-centers loss and $\Lambda(X_n)$, our lower bound parameter. The center and memory complexity are both bounded by $O(k \operatorname{poly}(\log n))$, and consequently the time complexity of the algorithm is the same. This makes our algorithm the first $O(1)$-approximation in this setting with efficient time and memory complexity. We devote the remainder of our paper to proving Theorem 1.

Our proof is based on three key steps. First, we use a refinement of Theorem 3 to bound the online loss $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$ with three terms: $\mathcal{L}_k(X_n)$, the desired k-means loss, $\Lambda(X_n)$, our unavoidable lower bound term, and $\sum_{t=1}^{n} d(x_t, S_t)^2$, the no-substitution loss function. Because our algorithm no longer adheres to the no-substitution rule, that centers are *never* removed, we need to refine our analysis to handle cases in which centers disappear. It turns out that because we only remove centers after the scale *increases*, the deletions only have a small effect. Formally, we have the following proposition (proved in Section B.2).

Proposition 10 *Suppose we run $Online\_Cluster(X, k, \delta)$. Then at all times n we have*

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2 \leq 8\Lambda(X_n) + 8\sum_{t=1}^{n} d(x_s, S_t)^2 + 4\mathcal{L}_k(X_n).$$

Our second step is to bound the loss $\sum_{t=1}^{n} d(x_t, S_t)^2$ under the assumption that R_t is sufficiently small for all t. This assumption allows us to circumvent the complex way in which the value of R is intertwined with whether or not selections have been made. As a result, we can cleanly divide our analysis into handling the loss, $d(x_t, S_t)^2$ and handling R_t separately. We do so with the following proposition (proved in Appendix B.3).

Proposition 11 *With probability at least $1 - \frac{\delta}{2}$ over the randomness of Online_Cluster, the following holds simultaneously for all $n \geq 1$:*

$$\sum_{t=1}^{n} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) \leq 33\mathcal{L}_k(X_n).$$

Our third step, is to show that R_t is bounded as indicated in the previous step. We have the following proposition (proved in Appendix B.4).

Proposition 12 *With probability at least $1 - \frac{\delta}{100}$ over the randomness of Online_Cluster, the following holds simultaneously for all $n \geq k$:*

$$R_n \leq \frac{\mathcal{L}_k(X_n)}{k}.$$

Armed with these three propositions, we have all the ingredients necessary to prove Theorem 1. First, a straightforward combination of the three propositions gives us the desired approximation factor of Theorem 1. Propositions 12 and 11 imply that $\sum_{t=1}^{n} d(x_t, S_t)^2$ is highly likely to be at most $O(\mathcal{L}_k(X_n))$, and Proposition 10 implies that our online loss consequently satisfies the desired bound.

For the center complexity, memory, and time complexity guarantees of Theorem 1, we directly derive them from our bound on R, Proposition 12. The argument is simple: selecting too many points (or equivalently, holding too many points in memory) necessarily increases the value of R, which will eventually force the bound in 12 to be violated. Given that our total number of point selections is small, it also follows that our memory and computation time must be small as well. We now give our proof of Theorem 1.

Proof We will show that part 1 of Theorem 1 holds with probability at least $1 - \frac{3\delta}{4}$, and parts 2 and 3 of Theorem 1 hold with probability at least $1 - \frac{\delta}{4}$. Theorem 1 will then follow from a union bound.

Approximation Factor: By a union bound, with probability at least $1 - \frac{3\delta}{4}$, the bounds in Propositions 12 and Proposition 11 both hold. It thus suffices to show that these conditions are sufficient for bounding $\sum_{t=2}^{n} d(x_t, S_{t-1})^2$. We have,

$$\begin{aligned}
\sum_{t=2}^{n} d(x_t, S_{t-1})^2 &\leq 8\Lambda(X_n) + 8\sum_{t=1}^{n} d(x_t, S_t)^2 + 4\mathcal{L}_k(X_n) \\
&= 8\Lambda(X_n) + 8\sum_{t=1}^{n} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) + 4\mathcal{L}_k(X_n) \\
&\leq 8\Lambda(X_n) + 264\mathcal{L}_k(X_n) + 4\mathcal{L}_k(X_n) \\
&= O(\Lambda(X_n) + \mathcal{L}_k(X_n)),
\end{aligned}$$

where the first inequality holds by Proposition 10, the second by Proposition 12, and the third by Proposition 11.

Center Complexity and Memory: For any time n, let U_n be those points added to S after time τ_i and up to time n, where $\tau_i \leq n < \tau_{i+1}$. Namely, $U_n = S_n \cap \{x_{\tau_i}, \ldots, x_n\}$. Because Algorithm 1 deletes no points from S between τ_i and τ_{i+1}, we have $S_n = (S_{\tau_i} \setminus \{x_{\tau_i}\}) \cup U_n$. At the time of the last scale change τ_i, just before Line 16 is executed, observe that $(S_{\tau_i} \setminus \{x_{\tau_i}\}) = U_{(\tau_i)-1} \cup \{z^1_{\tau_{(i-1)}}, \ldots, z^k_{\tau_{(i-1)}}\}$. Thus, $S_n = U_{(\tau_i)-1} \cup U_n \cup \{z^1_{\tau_{(i-1)}}, \ldots, z^k_{\tau_{(i-1)}}\}$.

Now, we will focus on bounding $\max_{n>k} |U_n|$. Suppose there were a time $n \geq k$ such that $|U_n| \geq 375k \log^6(\frac{2n}{\delta})$. Let τ_i satisfy $\tau_i \leq n \leq \tau_{i+1}$. By the definition of scale changes and by using Proposition 8, we know $256nw^2_{\tau_i} \geq w^2_n \geq \frac{\mathcal{L}_k(X_n)}{n}$. Therefore, R_{τ_i}, which is set to be $\frac{w^2_{\tau_i}}{128k}$ during the last scale change of Algorithm 1, satisfies $R_{\tau_i} \geq \frac{\mathcal{L}_k(X_n)}{2^{15}n^2k}$.

Let f be the number of times that R is doubled from times τ_i to n—because no scale changes occur in this time interval, we have that $R_n = R_{\tau_i} 2^f$. Every point in U_n comes from points chosen between times τ_i and n, and R is doubled at least every $25k \log^5(\frac{2n}{\delta})$ points that are chosen. Thus, $f \geq \frac{|U_n|}{25k \log^5(\frac{2n}{\delta})} \geq 15 \log(\frac{2n}{\delta})$. However, this implies that $R_n = R_{\tau_i} 2^{15 \log(\frac{2n}{\delta})} \geq \frac{n^{13}}{\delta^{15}} \frac{\mathcal{L}_k(X_n)}{k} \geq \frac{\mathcal{L}_k(X_n)}{k}$. By Proposition 12, this event occurs with probability at most $\frac{\delta}{4}$. Thus, the probability that there exists n such that $|U_n| \geq 375k \log^6 \frac{2n}{\delta}$ is at at most $\frac{\delta}{4}$.

Thus, $|U_n| \leq O(k \log^6(\frac{n}{\delta}))$ for all n with probability at least $1 - \frac{\delta}{4}$, and this implies $|S_n| \leq O(k \log^6(\frac{n}{\delta}))$ with the same probability. The memory of Algorithm 1 involves storing S_n and just $O(k)$ additional points for the k centers and $O(k)$ additional natural numbers bounded by n. Thus, the memory requirement is dominated by $|S_n| = O(k \log^6(\frac{n}{\delta}))$. Finally, it is clear from our algorithm that time time complexity is directly proportional to our memory, which completes the proof.

■

Acknowledgments

Robi Bhattacharjee thanks NSF under CNS 1804829 for research support. Jacob Imola would like to thank ONR under N00014-20-1-2334 and UC Lab Fees under LFR 18-548554 for research support.

References

Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 15–28. Springer, 2009.

Charu C Aggarwal. *Data streams: models and algorithms*, volume 31. Springer Science & Business Media, 2007.

S. Ahmadian, A. Norouzi-Fard, O. Svensson, and J. Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM Journal on Computing*, 49(4), 2020.

Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In *Advances in neural information processing systems*, pages 10–18, 2009.

Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

Aditya Bhaskara and Aravinda Kanchana Rwanpathirana. Robust algorithms for online k-means clustering. In Aryeh Kontorovich and Gergely Neu, editors, *Algorithmic Learning Theory, ALT 2020, 8-11 February 2020, San Diego, CA, USA*, volume 117 of *Proceedings of Machine Learning Research*, pages 148–173. PMLR, 2020.

Robi Bhattacharjee and Michal Moshkovitz. No-substitution k-means clustering with adversarial order. In *Algorithmic Learning Theory*, pages 345–366. PMLR, 2021.

Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 26–40. Society for Industrial and Applied Mathematics, 2011.

Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 626–635. ACM, 1997.

V. Cohen-Addad, B. Guedj, V. Kanade, and G. Rom. Online k-means clustering. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130, pages 1126–1134, 2021.

Sanjoy Dasgupta. *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California, 2008.

Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: Theory and practice. *IEEE transactions on knowledge and data engineering*, 15(3):515–528, 2003.

M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors. *Probabilistic methods for algorithmic discrete mathematics*, volume 16 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1998.

Tom Hess and Sivan Sabato. Sequential no-substitution k-median-clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 962–972, 2020.

Tom Hess, Michal Moshkovitz, and Sivan Sabato. A constant approximation algorithm for sequential no-substitution k-median clustering under a random arrival order. *arXiv preprint arXiv:2102.04050*, 2021.

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.

Elias Koutsoupias. The k-server problem. *Comput. Sci. Rev.*, 3(2):105–118, 2009.

Edo Liberty, Ram Sriharsha, and Maxim Sviridenko. An algorithm for online k-means clustering. In *2016 Proceedings of the eighteenth workshop on algorithm engineering and experiments (ALENEX)*, pages 81–89. SIAM, 2016.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1987.

N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

Adam Meyerson. Online facility location. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 426–431. IEEE, 2001.

Michal Moshkovitz. Unexpected effects of online no-substitution k-means clustering. In *Algorithmic Learning Theory*, pages 892–930. PMLR, 2021.

D. Pollard. Strong consistency of k-means clustering. *Annals of Statistics*, 9(1):135–140, 1981.

D. Pollard. Quantization and the method of k-means. *IEEE Transactions on Information Theory*, 28:199–205, 1982.

Michael Shindler, Alex Wong, and Adam W Meyerson. Fast and accurate k-means for large datasets. In *Advances in neural information processing systems*, pages 2375–2383, 2011.

Appendix A. Proof of Theorem 2

For convenience, we begin by restating Theorem 2.

Theorem 2 ***[Lower Bound]*** *Let X be any data sequence that contains infinitely many distinct points. Let A be an online clustering algorithm such that its output satisfies $|S_n| \leq b_n$ for all n and for all input sequences, where $\{b_n\}$ is a sequence of positive integers. Then there exists a sequence $\tilde{X} = \tilde{x}_1, \tilde{x}_2, \ldots$ such that the following conditions hold.*

1. *$\tilde{X}$ is drawn from the closure of X, (i.e. X and its limit points). Thus all points in $\tilde{X}$ are arbitrarily close to points in X.*

2. *For all $n \geq 2$, the expected loss over A satisfies $\mathbb{E}_A\left[\sum_{s=2}^{n} d(\tilde{x}_s, S_{s-1})^2\right] \geq \Omega(\Lambda(\tilde{X}_n))$.*

Here, we have a slight generalization of the statement in the introduction, as we now only assume that $|S_n| \leq b_n$ for some arbitrary sequence of integers $b_1, b_2, \ldots$. To obtain the earlier result, we simply substitute $b_n = n$.

Proof idea of Theorem 2 We first summarize the main ideas of the proof of Theorem 2.

The key idea is to first consider the case in which the input sequence X satisfies some additional structure that allows us to cleanly construct sub-sequences, $\tilde{X}$, with the desired property. We call such inputs *nice* sequences.

Definition 13 *A sequence X is **nice** if it consists of distinct points such that for all $1 < i < j$, $d(x_i, x_j) > \frac{1}{2}d(x_i, x_1)$ and $d(x_i, x_j) > \frac{1}{2}d(x_j, x_1)$.*

Nice sequences have the property that all of their points are relatively well "spread out" in comparison to their distance to the first point. Thus, in order for an algorithm to achieve a *better* loss than $\Lambda(X_n)$, the loss incurred by an algorithm that makes no guesses about future points, it must have guessed the value of x_n. This is the main idea behind how we construct $\tilde{X}$ from a nice sequence: the adversary randomly chooses its next points from a large set (larger than b_n) of points which means that on average, any algorithm is going to fail at guessing. We give a full proof of this case in Lemma 14 in Appendix A.

Next, to prove the general version of Theorem 2, it suffices to reduce a general sequence X to a nice sequence. In particular, we must show that any sequence X has a nice subsequence (with possible rearrangements to the order of the subsequence). To do this, we appeal to the fact that X is a subset of $\mathbb{R}^d$ by using casework on whether X is bounded, and thus contained in a compact set, or unbounded. Both cases follow from a similar type of argument, the only difference is that for bounded sequences, our nice sub-sequence $\tilde{X}$ converges while in the unbounded case it diverges.

We now prove a specific version of Theorem 2 for nice sequences (Definition 13).

Lemma 14 (Theorem 2 for nice sequences) *Let X be a nice sequence, and let A be an online clustering algorithm such that its output satisfies $|S_n| \leq b_n$ for all n and for all input sequences, where $\{b_n\}$ is a sequence of positive integers. There exists a sequence $\tilde{X}$ such that*

1. *All elements of $\tilde{X}$ are taken from X.*
2. *For all $n \geq 2$, the expected loss over A satisfies $\mathbb{E}_A\left[\sum_{s=2}^{n} d(\tilde{x}_s, S_{s-1})^2\right] \geq \frac{\Lambda(\tilde{X}_n)}{24}$.*

Proof We generate $\tilde{X}$ recursively with $\tilde{x}_1 = x_1$. Suppose we have generated $\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n$; we will explain how $\tilde{x}_{n+1}$ is obtained.

Let $r_i = \frac{d(x_i, x_1)}{4}$ for $i \geq 2$. For $i \neq j$, since X is nice,

$$\begin{aligned} d(x_i, x_j) &> \frac{d(x_i, x_1)/2 + d(x_j, x_1)/2}{2} \\ &= \frac{d(x_i, x_1)}{4} + \frac{d(x_j, x_1)}{4} \\ &= r_i + r_j, \end{aligned}$$

which implies $B(x_i, r_i)$ and $B(x_j, r_j)$ are disjoint where $B(x, r)$ denotes the closed ball of radius r centered at x. This implies that for *any* set S_n of size $\leq b_n$, there are at most b_n indices i for which $|S_n \cap B(x_i, r_i)| \geq 1$. Thus, for a fixed choice of S_n and a randomly chosen $2 \leq i \leq 3b_n + 1$, with probability at least $\frac{2}{3}$, $d(x_i, S_n) > r_i$. Applying this over all S_n generated by A (after seeing $\tilde{x}_1, \ldots, \tilde{x}_n$) and switching orders, we have

$$\Pr_i \Pr_A[d(x_i, S_n) > r_i] = \Pr_A \Pr_i[d(x_i, S_n) > r_i] \geq \frac{2}{3}.$$

Thus, there exists i for which $\Pr_A[d(x_i, S_n) > r_i] \geq \frac{2}{3}$. It follows by Markov's inequality that

$$\mathbb{E}_A[d(x_i, S_n)^2] \geq \frac{2}{3} r_i^2 + \frac{1}{3} 0 = \frac{d(x_1, x_i)^2}{24}.$$

We now set $\tilde{x}_{n+1} = x_i$, which concludes the definition of $\tilde{X}$. Since $x_1 \in \tilde{X}$, the above equation implies that

$$\mathbb{E}_A[d(\tilde{x}_{n+1}, S_n)^2] \geq \frac{d(\tilde{x}_1, \tilde{x}_{n+1})^2}{24} \geq \frac{d(\tilde{x}_{n+1}, X_n)^2}{24}.$$

Summing this and applying linearity of expectation, we have that

$$\begin{aligned} \mathbb{E}_A \left[\sum_{s=2}^{n} d(\tilde{x}_s, S_{s-1})^2 \right] &\geq \sum_{s=2}^{n} \frac{d(\tilde{x}_s, \tilde{X}_{s-1})^2}{24} \\ &\geq \frac{\Lambda(\tilde{X}_n)}{24}. \end{aligned}$$

■

We now prove Theorem 2.

Proof (Theorem 2)

We claim there exists a nice sequence of points $Z \subseteq \overline{X}$. By Lemma 14, this suffices. To show this, we have two cases.

Case 1: X is bounded: Since X contains infinitely many distinct points, there is an infinitely-long subsequence that has all distinct points. Taking this subsequence if necessary, we assume without loss of generality that all points in X are distinct. Since X is a sequence in a bounded region in $\mathbb{R}^d$, it follows that X is contained within a compact subset of $\mathbb{R}^d$. Thus, by the definition of sequential compactness, X must contain a subsequence that converges. Let $y_1, y_2, \ldots$ denote this sequence and let y denote its limit.

In the case that $y = y_i$ for some i, simply delete this entry. So we may now assume that $y, y_1, \ldots$ are distinct points such that $\lim_{i \to \infty} y_i = y$. We now construct Z recursively. Let $z_1 = y$ and $z_2 = y_1$.

Suppose we have constructed $z_1, \ldots, z_n$ thus far. Since y_i converges to $z_1 = y$, we can simply pick a point y_j such that $d(z_1, y_j) < \frac{1}{2} d(z_1, z_n)$. We let z_{n+1} equal such a point, and this concludes the construction.

To verify the condition holds, let $i \neq j > 1$ and without loss of generality suppose $i > j$. This already implies $d(z_j, z_1) > d(z_i, z_1)$, and all that remains is to show that $d(z_i, z_j) > \frac{1}{2} d(z_j, z_1)$.

This follows from the triangle inequality:

$$\begin{aligned} d(z_i, z_j) &\geq d(z_j, z_1) - d(z_i, z_1) > d(z_j, z_1) - \frac{1}{2} d(z_j, z_1) \\ &= \frac{d(z_j, z_1)}{2}. \end{aligned}$$

Case 2: X is unbounded: Let $z_1 = x_1$ and $z_2 = x_2$ with $x_2 \neq x_1$. We then construct Z recursively.

Suppose we have constructed $z_1, \ldots, z_n$ thus far. Then select z_{n+1} to be any point in X such that $2d(z_1, z_n) < d(z_1, z_{n+1})$. This completes the construction.

To verify the condition holds, let $i \neq j > 1$ and without loss of generality suppose $i > j$. This implies $d(z_i, z_1) > d(z_j, z_1)$, and all that remains is to show that $d(z_i, z_j) > \frac{1}{2} d(z_i, z_1)$. Using the triangle inequality in the same way as in the bounded case, we can establish this. This completes the proof for both the bounded and unbounded cases. ■

Appendix B. Proof of Theorem 1

We begin by proving Proposition 8, which will be frequently used to relate scale changes to the k-means clustering loss.

B.1. Proof of Proposition 8

Proof The subroutine $online\_k\_centers$ maintains a k-centers clustering with cost w_n (at time n) that is an 8-approximation to the optimal k-centers cost.

By directly using the given k-centers clustering with cost w_n as a k-means clustering, we get an upper bound of nw_n^2. Because the lower bound of the k-centers cost is $\frac{w_n}{8}$, there must exist two points in any set of k clusters that are clustered together with distance at least $\frac{w_n}{8}$. By the triangle inequality, under any cluster center these two points will incur cost at least $2(\frac{w_n}{16})^2 = \frac{w_n^2}{128}$, which finishes the proof. ■

B.2. Proof of Proposition 10: Bounding the loss with Λ

The proof follows an almost identical argument to the proof of Theorem 3 given in Section 5.1. The only difference is that the set S_t of centers maintained by Online_Cluster changes over time as Online_Cluster is not a no-substitution clustering algorithm. However, Online_Cluster only deletes points at scale changes, and these are easy events to reason about.

In Theorem 3, we used the fact that for a no-substitution algorithm, $d(x_t, S_{t'}) < d(x_t, S_t)$ for any $t' < t$ because $S_{t'}$ can only get bigger. Analyzing the behavior of Online_Cluster at scale changes, we can prove a similar result.

Lemma 15 *Let $l \geq 1$, and let times t, t' that satisfy $1 \leq t \leq t' < \tau_{l+2}$. Then $d(x_t, S_{t'}) \leq d(x_t, S_t) + w_{\tau_l} \mathbb{1}(t \leq \tau_{l+1})$.*

Proof Suppose $t > \tau_{l+1}$. Then since $t' < \tau_{l+2}$, the latest scale change on or before t' occurs at time τ_{l+1} during which all points on or before τ_l are deleted. It follows that $S_t \subseteq S_{t'}$ since no points are deleted between times t and t' inclusive. This implies the inequality.

Otherwise, if $t \leq \tau_{l+1}$, let $x \in S_t$ be the nearest neighbor of x_t in S_t. If x arrived after τ_l, then $x \in S_{t'}$. Thus $d(x_t, S_{t'}) \leq d(x_t, S_t)$, and we are done. Otherwise, there is a point z such that $d(z, x) \leq w_{\tau_l}$ is in $S_{t'}$: it is one of the k-center points added in Line 11 at time τ_l. We can use the triangle inequality to conclude the result. ■

Now, similar to Theorem 3, it follows that

$$\begin{aligned} d(x_t, S_{t-1}) &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, x_{p(t)}) + d(x_{p(t)}, S_{t-1}) \\ &\leq d(x_t, x_{u(t)}) + d(x_{u(t)}, x_{p(t)}) + d(x_{p(t)}, S_{p(t)}) + w_{\tau_l}^2 \mathbb{1}(p(t) \leq \tau_{l+1})), \end{aligned}$$

where the first step is the double application of the triangle inequality and the second follows from applying Lemma 15. By squaring the above and applying Cauchy Schwartz, we have

$$d(x_t, S_{t-1})^2 \leq 4d(x_t, x_{u(t)})^2 + 4d(x_{u(t)}, x_{p(t)})^2 + 4d(x_{p(t)}, S_{p(t)})^2 + 4w_{\tau_l}^2 \mathbb{1}(p(t) \leq \tau_{l+1}).$$

Note the similarity between this inequality and the one used in the proof of Theorem 3: the only difference is the additional $4w_{\tau_l}^2 \mathbb{1}(p(t) \leq \tau_{l+1})$ term. We form an upper bound on the loss as follows:

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2 \leq \underbrace{4\sum_{t=2}^{n} d(x_t, x_{u(t)})^2}_{=4\Lambda(X_n)} + \underbrace{4\sum_{t=2}^{n} d(x_{u(t)}, x_{p(t)})^2}_{\leq 4\Lambda(X_n)} + \underbrace{4\sum_{t=2}^{n} d(x_{p(t)}, S_{p(t)})^2}_{\leq 8\sum_{t=1}^{n} d(x_t, S_t)^2} + \underbrace{4\sum_{t=1}^{n} w_{\tau_l}^2 \mathbb{1}(p(t) \leq \tau_{l+1})}_{\leq 8\tau_{l+1} w_{\tau_l}^2}$$

Here, the first equality holds by definition; the second inequality holds because $\sum_{t=2}^{n} d(x_{p(t)}, x_{u(t)})^2 \leq \sum_{t=2}^{n} d(x_t, x_{u(t)})^2$ since $p(t)$ maps a child of $u(t)$ to its previous child (and to $u(t)$ if there is no previous child); the third inequality holds from 2-injectivity of $p(t)$; and the fourth inequality holds because $p(t)$ is 2-injective and maps an index to an index strictly less, and thus at most $2\tau_{l+1}$ values of t may satisfy $p(t) \leq \tau_{l+1}$.

Finally, by Definition 9, $\tau_{l+1} w_{\tau_l}^2 < \frac{w_{\tau_{l+1}}^2}{256}$, and by Proposition 8, $\frac{w_{\tau_{l+1}}^2}{256} \leq \frac{\mathcal{L}_k(X_n)}{2}$. Substituting, we obtain

$$\sum_{t=2}^{n} d(x_t, S_{t-1})^2 \leq 8\Lambda(X_n) + 8\sum_{t=1}^{n} d(x_t, S_t)^2 + 4\mathcal{L}_k(X_n).$$

B.3. Proof of Proposition 11: Bounding $\sum_{t=1}^{n} d(x_t, S_t)^2$

Proposition 11 *With probability at least $1 - \frac{\delta}{2}$ over the randomness of Online_Cluster, the following holds simultaneously for all $n \geq 1$:*

$$\sum_{t=1}^{n} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) \leq 33\mathcal{L}_k(X_n).$$

Our main idea will be to fix n (we will later use a union bound to obtain simultaneity over all n), and bound the desired sum over subsets of X_n that are between scale changes. This allows us to circumvent issues posed by deleting points. Then, we will sum our bounds over all intervals of scale changes and conclude by appealing to the exponentially growing nature of data over successive scale changes.

We begin with a general lemma that assists in bounding our loss for an arbitrary set of times that occur between of pair of scale changes. Think of X as a fixed, possibly infinite, sequence. Since the online k-center subroutine is deterministic, the scale-change times τ_i are also predetermined. The only randomness arises from line 16 of the algorithm where we probabilistically add a point to S.

Lemma 16 *Let $T \subseteq \{1, \ldots, n\}$ be set of times such that $T \subset [\tau_i, \tau_{i+1})$ for some i. Let $\min T$ denote the smallest time in T. Let $X_{(T)}$ denote $\{x_t : t \in T\}$. Then for any $\Gamma > 0$ and $\beta \geq 0$, with probability at least $1 - \exp\left(-\frac{\beta \log^4 \frac{2 \min T}{\delta}}{\Gamma}\right)$,*

$$\sum_{t \in T} d(x_t, S_t)^2 \mathbb{1}(R_t \leq \Gamma) \leq \beta + \max_{t \in T} \mathcal{L}(X_{(T)}, x_t).$$

Proof Let $A(T, \Gamma, \beta)$ denote the event that we don't want; i.e. that $\sum_{t \in T} d(x_t, S_t)^2 \mathbb{1}(R_t \leq \Gamma) > \beta + \max_{t \in T} \mathcal{L}(X_{(T)}, x_t)$.

Let $\{E_t\}_{t=1}^{\infty}$ denote the sequence of Boolean variables indicating whether x_t was selected as a center on line 16. Writing $s = \min T$, we will show the slightly stronger statement that $\Pr[A(T, \Gamma, \beta) | E_1, \ldots, E_{s-1}] \leq \exp(-\frac{\beta \log^4 \frac{2s}{\delta}}{\Gamma})$ for any choice of $E_1, \ldots, E_{s-1}$. The result will then follow by conditional probability.

We use induction on $|T|$ to show the result holds. For $T = \{\}$, the claim trivially holds. For the inductive step, take $T' = T \setminus \{s\}$; we will suppose that the claim holds for T' and for any choice of β', Γ', and $E_1, \ldots, E_{s'}$, where $s' = \min T'$. By marginalizing over the times $s+1$ through s', our inductive hypothesis implies that $\Pr[A(T', \Gamma', \beta') | E_1, \ldots, E_s] \leq \exp(-\frac{\beta' \log^4 \frac{2s}{\delta}}{\Gamma'})$.

Given $E_1, \ldots, E_{s-1}$, we have

$$\Pr[E_s = 0 | E_1, \ldots, E_{s-1}] \leq \max\left\{0, \left(1 - \frac{\gamma^2 \log^4 \frac{2s}{\delta}}{R_s}\right)\right\} \leq \exp\left(-\frac{\gamma^2 \log^4 \frac{2s}{\delta}}{R_s}\right), \quad (3)$$

where $\gamma = d(x_s, S'_s)$, and $S'_s = S_s \setminus \{x_s\}$ (S'_s is the value of S before line 16 when deciding to take x_s).

We will now bound $\Pr[A(T, \Gamma, \beta) | E_1, \ldots, E_{s-1}]$ as follows:

$$\begin{aligned}\Pr[A(T, \Gamma, \beta) | E_1, \ldots, E_{s-1}] \leq \Pr[E_s = 0 | E_1, \ldots, E_{s-1}] \Pr[A(T, \Gamma, \beta) | E_1, \ldots, E_{s-1}, E_s = 0] \\ + \Pr[E_s = 1 | E_1, \ldots, E_{s-1}] \Pr[A(T, \Gamma, \beta) | E_1, \ldots, E_{s-1}, E_s = 1].\end{aligned}$$

We analyze the two parts separately, beginning with the easier one.

Case 1: $E_s = 1$**.** The key observation in this case is that because $T \subseteq [\tau_i, \tau_{i+1})$, no deletions occur after x_s is selected. Therefore, $x_s \in S_t$ for all $t \in T$ with $t > s$. Furthermore, x_s itself incurs

0 loss as $x_s \in S_s$. This implies that

$$\begin{aligned}\sum_{t\in T} d(x_t, S_t)^2 \mathbb{1}(R_t \le \Gamma) &\le \sum_{t\in T} d(x_t, x_s)^2 \\ &= \mathcal{L}(X_{(T)}, x_s) \\ &\le \beta + \max_{t\in T} \mathcal{L}(X_{(T)}, x_t).\end{aligned}$$

Thus, $\Pr[A(T,\Gamma,\beta)|E_1,\dots,E_{s-1},E_s=1]=0$.

Case 2: $E_s = 0$**.** In this case, we have two subcases, first, when $\gamma^2 > \beta$ and second when $\gamma^2 \le \beta$.

First, suppose $\gamma^2 > \beta$. If $R_s > \Gamma$, then $\Pr[A(T,\Gamma,\beta)|E_1,\dots,E_{s-1},E_s = 0] = 0$ as $A(T,\Gamma,\beta)$ cannot occur. Otherwise, if $R_s \le \Gamma$, then using (3), we have

$$\begin{aligned}&\Pr[E_s = 0|E_1,\dots,E_{s-1}]\Pr[A(T,\Gamma,\beta)|E_1,\dots,E_{s-1},E_s=0] \\ &\le \Pr[E_s=0|E_1,\dots,E_{s-1}] \le \exp\left(-\frac{\beta \log^4 \frac{2s}{\delta}}{\Gamma}\right).\end{aligned}$$

Second, suppose $\gamma^2 \le \beta$. Recall $T' = T \setminus \{s\}$. Observe that

$$\begin{aligned}\sum_{t\in T} d(x_t,S_t)^2\mathbb{1}(R_t\le\Gamma) &\le d(x_s,S_s)^2 + \sum_{t\in T'} d(x_t,S_t)^2\mathbb{1}(R_t\le\Gamma) \\ &\le \gamma^2 + \sum_{t\in T'} d(x_t,S_t)^2\mathbb{1}(R_t\le\Gamma).\end{aligned}$$

Since $T' \subset T$, it follows that $\mathcal{L}(X_{(T)},x_t) \ge \mathcal{L}(X_{(T')},x_t)$ which implies that $\max_{t\in T}\mathcal{L}(X_{(T)},x_t) \ge \max_{t\in T'}\mathcal{L}(X_{(T')},x_t)$. Combining these observations, we have

$$\begin{aligned}&\sum_{t\in T} d(x_t,S_t)^2\mathbb{1}(R_t\le\Gamma) \ge \beta + \max_{t\in T}\mathcal{L}(X_{(T)},x_t) \\ \implies\ &\gamma^2 + \sum_{t\in T'} d(x_t,S_t)^2\mathbb{1}(R_t\le\Gamma) \ge \beta + \max_{t\in T'}\mathcal{L}(X_{(T')},x_t),\end{aligned}$$

so when $\gamma^2 \le \beta$, $A(T,\Gamma,\beta)$ implies $A(T',\Gamma,\beta-\gamma^2)$. Thus

$$\begin{aligned}&\Pr[E_s=0|E_1,\dots,E_{s-1}]\Pr[A(T,\Gamma,\beta)|E_1,\dots,E_{s-1},E_s=0] \\ &\le \Pr[E_s=0|E_1,\dots,E_{s-1}]\Pr[A(T',\Gamma,\beta-\gamma^2)|E_1,\dots,E_{s-1},E_s=0] \\ &\le \exp\left(-\frac{\gamma^2\log^4\frac{2s}{\delta}}{R_s}\right)\exp\left(-\frac{(\beta-\gamma^2)\log^4\frac{2s'}{\delta}}{\Gamma}\right) \\ &\le \exp\left(-\frac{\beta\log^4\frac{2s}{\delta}}{\Gamma}\right).\end{aligned}$$

where we have used equation (3) and the inductive hypothesis for the second inequality. Thus, regardless of whether or not $\gamma^2 \le \beta$, the bound of $\exp\left(-\frac{\beta\log^4\frac{2s}{\delta}}{\Gamma}\right)$ holds, and we have

$$\Pr[A(T,\Gamma,\beta)|E_1,\dots,E_{s-1}] \le \exp\left(-\frac{\beta\log^4\frac{2s}{\delta}}{\Gamma}\right),$$

as desired. ■

Our next step is to apply Lemma 16 to get a bound on the loss function over well behaved time intervals. Our key construction for doing this is the notion of a *cluster ring*, which was introduced in Braverman et al. (2011).

Definition 17 *Let C be a set of points. Let μ denote the mean of C, and $\gamma = \frac{\mathcal{L}(C)}{|C|}$ be the average cost of clustering each point. Then the jth* ***cluster ring*** *of C, denoted C_j, is defined as*

- $C_0 = \{x \in C : d(x, \mu)^2 < \gamma\}$
- $C_j = \{x \in C : 2^{j-1}\gamma \leq d(x, \mu)^2 < 2^j\gamma\}$ *for* $j \geq 1$.

The intuition behind cluster rings is that any point in C_j serves as a reasonable cluster center. Thus, cluster rings are particularly amenable to Lemma 16: when $X_{(T)}$ is a cluster ring, the term $\max_{t\in T} \mathcal{L}(X_{(T)}, x_t)$ can be controlled. We apply this in our next step where we consider time intervals T that are both bounded between scale changes and change by at most a factor of two.

Lemma 18 *Let a, m be times satisfying $\tau_i \leq a \leq m < \tau_{i+1}$ for some i, and $m < 2a$. Let $X_{a:m}$ denote $\{x_a, \dots x_m\}$. Then with probability at least $1 - \frac{\delta}{4m^2}$ over the randomness of Online_Cluster,*

$$\sum_{t=a}^{m} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) \leq 8\mathcal{L}_k(X_{a:m}) + 4\frac{\mathcal{L}_k(X_m)}{\log^2 a}.$$

Proof Let $C^1, C^2, \dots, C^k$ denote the optimal k-means clustering of $X_{a:m}$, and let $c^1, c^2 \dots c^k$ denote their respective centers. Using Definition 17, let C^i_j denote the jth cluster ring of C^i. Let $\Gamma = \frac{\mathcal{L}_k(X_m)}{k}$ and $\beta = \frac{3\Gamma}{\log^3 a}$. Then by applying Lemma 16 to all non-empty C^i_j and applying a union bound, we have that with probability at least $1 - \sum_{i,j:|C^i_j|\geq 1} \exp\left(-\frac{\beta \log^4 \frac{2a}{\delta}}{\Gamma}\right)$

$$\begin{aligned}\sum_{t=a}^{m} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) &\leq \sum_{i,j:|C^i_j|\geq 1} \sum_{x_t \in C^i_j} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \Gamma\right) \\ &\leq \sum_{i,j:|C^i_j|\geq 1} \frac{3\Gamma}{\log^3 a} + \max_{c\in C^i_j} \mathcal{L}(C^i_j, c).\end{aligned} \tag{4}$$

Note that we can safely replace the indicator variables bounding R_t with a uniform $\mathbb{1}(R_t \leq \Gamma)$ since $\mathcal{L}_k(X_t)$ is monotonically non-decreasing.

It consequently suffices to show that $\sum_{i,j:|C^i_j|\geq 1} \exp\left(-\frac{\beta \log^4 \frac{2a}{\delta}}{\Gamma}\right) \leq \frac{\delta}{4m^2}$, and that (4) implies the desired bound. To do so, we will leverage a few simple properties of cluster rings.

First, observe that there are at most m non-empty cluster rings as there are at most m points in $X_{a:m}$. It follows by substituting this along with $\beta = \frac{3\Gamma}{\log^3 a}$ and $m < 2a$ that

$$\begin{aligned}\sum_{i,j:|C_j^i|\geq 1} \exp\left(-\frac{\beta \log^4 \frac{2a}{\delta}}{\Gamma}\right) &\leq m \exp\left(-\frac{\beta \log^4 \frac{2a}{\delta}}{\Gamma}\right) \\ &\leq m \exp\left(-\frac{3\Gamma \log^4 \frac{2a}{\delta}}{\Gamma \log^3 a}\right) \\ &\leq m \exp\left(-3 \log \frac{m}{\delta}\right) \\ &\leq \frac{\delta}{4m^2}.\end{aligned}$$

Thus, (4) holds with the desired probability of at least $1 - \frac{\delta}{4m^2}$. Next, for any C_j^i we upper bound $\max_{c \in C_j^i} \mathcal{L}(C_j^i, c)$. Let c^i denote the optimal cluster center (mean) of C^i and $\gamma_i = \frac{\mathcal{L}(C^i)}{|C^i|}$. Then by Definition 17, we have

$$\begin{aligned}\max_{c \in C_j^i} \mathcal{L}(C_j^i, c) &= \max_{c \in C_j^i} \sum_{c' \in C_j^i} d(c, c')^2 \\ &\leq \max_{c \in C_j^i} \sum_{c' \in C_j^i} 2d(c, c^i)^2 + 2d(c', c^i)^2 \\ &\leq 4|C_j^i|(2^j \gamma_i) \leq 8|C_j^i|(2^{j-1}\gamma_i) \\ &\leq 8 \sum_{c \in C_j^i} d(c^i, c)^2 = 8\mathcal{L}(C_j^i, c^i).\end{aligned} \tag{5}$$

Additionally, there are at most $m - a + 1 \leq a$ points in $X_{a:m}$, which means there are at most a points in C^i for any i. It follows that there are at most $\lfloor \log a \rfloor + 1$ non-empty cluster rings, C_j^i as $2^{\log a}\gamma_i$ is too large to be the cost incurred by any point in C^i. Applying this along with (5), we have that

$$\begin{aligned}\sum_{i,j:|C_j^i|\geq 1} \frac{3\Gamma}{\log^3 a} + \max_{c \in C_j^i} \mathcal{L}(C_j^i, c) &\leq k(\lfloor \log a \rfloor + 1)\frac{3\Gamma}{\log^3 a} + 8\sum_{i=1}^{k} \sum_{|C_j^i|\geq 1} \mathcal{L}(C_j^i, c^i) \\ &\leq \frac{4\mathcal{L}_k(X_m)}{\log^2 a} + 8\mathcal{L}_k(X_{a:m}).\end{aligned}$$

■

We are now ready to prove Proposition 11. The main idea is to apply the previous lemma to a series of intervals $[a : m]$ which effectively partition the entire input sequence. While a natural starting point is to simply use the scale changes, $\tau_1, \tau_2, \ldots$ (thus considering intervals $[\tau_i : \tau_{i+1} - 1]$), this faces a problem; τ_{i+1} can be potentially much larger than τ_i, and the loss term from Lemma 18 would have a dependence on τ_i (as we are implicitly setting $a = \tau_i$). To deal with this, we need to further subdivide the intervals $[\tau_i, \tau_{i+1})$ using a sequence of times

$$\tau_i = \tau_{i,1} < \tau_{i,2} < \cdots < \tau_{i,s_i+1} = \tau_{i+1}, \tag{6}$$

that are chosen so that $\tau_{i,j+1} \leq 2\tau_{i,j}$. In the context of Lemma 18, this means that $a \simeq m$ up to a constant factor.

Proof (Proposition 11). Let $X = \{x_1, \dots\}$ be an input sequence, and let $\tau_1, \tau_2, \dots$ be the scale changes in X (Definition 9). As per the discussion above, we begin by defining $\tau_{i,j}$ as follows.

Let $\tau_{i,1}, \tau_{i,2}, \dots, \tau_{i,s_i}$ be defined as $\tau_{i,1} = \tau_i$, and $\tau_{i,j+1} = \min(2\tau_{i,j}, \tau_{i+1})$ with $\tau_{i,s_i+1} = \tau_{i+1}$ by convention. Note that we define $\tau_l = \infty$ if τ_{l-1} is the last scale change in X, and we correspondingly have that $s_{l-1} = \infty$.

For any $m \geq k+1$, define $\sigma(m)$ as the largest $\tau_{i,j}$ with $m \geq \tau_{i,j}$. For all m, it follows that $\sigma(m) \leq m \leq 2\sigma(m)$ (as otherwise maximality of $\sigma(m)$ would be contradicted) and that no scale changes occur in $[\sigma(m)+1, m]$. Finally, we let E_m denote the event that

$$\sum_{t=\sigma(m)}^{m} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) \leq 8\mathcal{L}_k(X_{\sigma(m):m}) + 4\frac{\mathcal{L}_k(X_m)}{\log^2 \sigma(m)}.$$

By Lemma 18, E_m holds with probability at least $1 - \frac{\delta}{4m^2}$ which implies (through a union bound) that $\bigcap_{m \geq k+1} E_m$ holds with probability at least $1 - \frac{\delta}{2}$. Thus, it suffices to show that $\bigcap_{m \geq k+1} E_m$ implies that $\sum_{t=1}^{n} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) \leq 33\mathcal{L}_k(X_n)$ holds for all n.

To this end, suppose $\bigcap_{m \geq k+1} E_m$ holds. Fix any $n \geq k+1$ (the case $n \leq k$ is trivial as we pick the first k points by default). Let $\sigma(n) = \tau_{l,r}$. For brevity, we also write $d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right)$ as α_t. It follows that

$$\begin{aligned}
\sum_{t=1}^{n} \alpha_t &= \sum_{t=1}^{\tau_l - 1} \alpha_t + \sum_{t=\tau_l}^{n} \alpha_t \\
&= \sum_{t=1}^{k} d(x_t, S_t)^2 + \sum_{t=\tau_1}^{\tau_l - 1} \alpha_t + \sum_{t=\tau_l}^{n} \alpha_t \\
&= \sum_{i=1}^{l-1} \sum_{j=1}^{s_i} \sum_{t=\tau_{i,j}}^{\tau_{i,j+1}-1} \alpha_t + \sum_{j=1}^{r-1} \sum_{t=\tau_{l,j}}^{\tau_{l,j+1}-1} \alpha_t + \sum_{t=\tau_{l,r}}^{n} \alpha_t. \\
&= \sum_{i=1}^{l-1} \sum_{j=1}^{s_i} \sum_{t=\sigma(\tau_{i,j+1}-1)}^{\tau_{i,j+1}-1} \alpha_t + \sum_{j=1}^{r-1} \sum_{t=\sigma(\tau_{l,j+1}-1)}^{\tau_{l,j+1}-1} \alpha_t + \sum_{t=\sigma(n)}^{n} \alpha_t,
\end{aligned} \tag{7}$$

with the last step following from $\sigma(\tau_{i,j+1} - 1) = \tau_{i,j}$. We can now bound the inner summands by noting that each of them corresponds to an event E_m. In particular, by applying E_m for $m = \tau_{i,j+1} - 1$ for $1 \leq i \leq l-1$ and $1 \leq j \leq s_i$, we have

$$\begin{aligned}
\sum_{i=1}^{l-1} \sum_{j=1}^{s_i} \sum_{t=\sigma(\tau_{i,j+1}-1)}^{\tau_{i,j+1}-1} \alpha_t &\leq \sum_{i=1}^{l-1} \sum_{j=1}^{s_i} 8\mathcal{L}_k\left(X_{\sigma(\tau_{i,j+1}-1):\tau_{i,j+1}-1}\right) + 4\frac{\mathcal{L}_k\left(X_{\tau_{i,j+1}-1}\right)}{\log^2 \sigma(\tau_{i,j+1}-1)} \\
&\leq 8\mathcal{L}_k(X_{\tau_l - 1}) + \sum_{i=1}^{l-1} \sum_{j=1}^{s_i} 4\frac{\mathcal{L}_k\left(X_{\tau_{i,j+1}-1}\right)}{\log^2 \sigma(\tau_{i,j+1}-1)} \\
&\leq 8\mathcal{L}_k(X_{\tau_l - 1}) + \sum_{i=1}^{l-1} 4\mathcal{L}_k\left(X_{\tau_{i+1}}\right) \sum_{j=1}^{s_i} \frac{1}{\log^2 \tau_{i,j}},
\end{aligned}$$

with the manipulations coming from combining the k-means losses for different intervals of X (i.e. $\mathcal{L}_k(A \cup B) \geq \mathcal{L}_k(A) + \mathcal{L}_k(B)$) and by observing that $\mathcal{L}_k(X_t)$ is monotonic in t.

To further bound this quantity, we simply note that $\tau_{i,j} = 2\tau_{i,j-1}$ for all but the last term, which implies that $\sum_{j=1}^{s_i} \frac{1}{\log^2 \tau_{i,j}}$ is at most $\frac{7}{4}$ (by crudely bounding the maximal infinite series $\sum \frac{1}{n^2}$). Substituting this gives

$$\sum_{i=1}^{l-1} \sum_{j=1}^{s_i} \sum_{t=\sigma(\tau_{i,j+1}-1)}^{\tau_{i,j+1}-1} \alpha_t \leq 8\mathcal{L}_k(X_{\tau_l-1}) + 7\sum_{i=1}^{l-1} \mathcal{L}_k\left(X_{\tau_{i+1}}\right)$$

However, the latter sum can be further bounded by observing that by applying Proposition 8 and Definition 9, we have

$$\mathcal{L}_k(X_{\tau_i}) \geq \frac{w_{\tau_i}^2}{128} > \frac{256\tau_{i-1}w_{\tau_{i-1}}^2}{128} = 2\tau_{i-1}w_{\tau_{i-1}}^2 \geq 2\mathcal{L}_k(X_{\tau_{i-1}}).$$

Thus, by summing a geometric sequence, we have

$$\sum_{i=1}^{l-1} \sum_{j=1}^{s_i} \sum_{t=\sigma(\tau_{i,j+1}-1)}^{\tau_{i,j+1}-1} \alpha_t \leq 8\mathcal{L}_k(X_{\tau_l-1}) + 14\mathcal{L}_k(X_{\tau_l}) \tag{8}$$

By applying essentially the same argument to the other two sums in Equation 7, we have

$$\begin{aligned} \sum_{j=1}^{r-1} \sum_{t=\sigma(\tau_{l,j+1}-1)}^{\tau_{l,j+1}-1} \alpha_t &\leq 8\mathcal{L}_k(X_{\tau_l:(\tau_{l,r}-1)}) + 7\mathcal{L}_k\left(X_{\tau_{l,r}}\right) \\ \sum_{t=\sigma(n)}^{n} \alpha_t &\leq 8\mathcal{L}_k\left(X_{\tau_{l,r}:n}\right) + 4\mathcal{L}_k(X_n). \end{aligned} \tag{9}$$

Finally, summing Equations 8 and 9 and combining with Equation 7 gives that

$$\begin{aligned} \sum_{t=1}^{n} d(x_t, S_t)^2 \mathbb{1}\left(R_t \leq \frac{\mathcal{L}_k(X_t)}{k}\right) &\leq 8\mathcal{L}_k(X_n) + 14\mathcal{L}_k(X_{\tau_l}) + 7\mathcal{L}_k(X_{\tau_{l,r}}) + 4\mathcal{L}_k(X_n) \\ &\leq 33\mathcal{L}_k(X_n), \end{aligned}$$

completing the proof. ■

B.4. Proof of Proposition 12

Proposition 12 *Running Online_Cluster(X, k, δ) satisfies* $\Pr[R_n \leq \frac{\mathcal{L}_k(X_n)}{k}$ *for all* $n \geq k] \geq 1 - \frac{\delta}{100}$.

The proof boils down to showing that once R_n becomes large, Online_Cluster is unlikely to select many centers after the last scale change. This claim will be useful later because we will see that selecting lots of centers is the main factor increasing the value of R_n. Intuitively, this claim is true because points are selected with probability inversely proportional to R_n; however, proving the claim is complicated by the fact that point selections are not independent. Thus, we make use of martingale concentration results to prove the formal lemma below, though the intuition is still straightforward.

Lemma 19 *Let n be a positive integer such that $n > k$. Suppose we run $Online\_Cluster(X, k, \delta)$, and we are given that there is a time $q < n$ where the following hold*

1. *No scale changes occur in the interval $[q, n]$*
2. *$\frac{\mathcal{L}_k(X_n)}{2k} \leq R_t$ for all $t \in [q, n]$.*

Let $count(q, n)$ be the number of centers selected during the interval $[q, n]$. For any $0 < \delta \leq 1$, we have that $\Pr[count(q, n) \geq 25k \log^5 \frac{2n}{\delta}] \leq \frac{\delta}{165n^2}$.

Proof

Let $C^1, C^2, \ldots, C^k$ denote the optimal k-clustering of X_n, and let $c^1, c^2 \ldots c^k$ denote their respective centers. Let $\gamma_i = \frac{\mathcal{L}(C^i)}{|C^i|}$ be the average cost of cluster C^i. For $j \geq 0$, let C^i_j be the jth cluster ring of C^i as in Definition 17. Recall that C^i_j is empty for all $j > \log |C^i|$. Since $|C^i| \leq n$, at most $k(\log n + 2)$ of the sets C^i_j are non-empty. For ease of notation, for $x \in X$, let $C(x)$ denote C^i_j, where C^i_j is the unique ring with $x \in C^i_j$.

For $q \leq t \leq n$, let E_t be the random variable defined by

$$E_t = \begin{cases} 1 & x_t \text{ is selected, } |C(x_t) \cap S_t| \geq 1 \\ 0 & \text{otherwise} \end{cases},$$

where we take the value of S_t right before Line 16 of the algorithm is executed.

In other words, if point x_t is taken, then $E_t = 1$, except if x_t is the first to be selected in $C(x_t)$. For each ring C^i_j, there can be at most one such point. Since the number of rings is at most $k(\log n + 2)$, it follows that $count(q, n) \leq k(\log n + 2) + \sum_{t=q}^n E_t$. We will complete the proof by showing that $\Pr[\sum_{t=q}^n E_t \geq 24k \log^5 \frac{2n}{\delta}] \leq \frac{\delta}{165n^2}$.

We do this by computing an absolute bound on $\Pr[E_t = 1]$ for all $q \leq t \leq n$ and then by using a martingale concentration result—even though the E_t are not independent, the absolute bound still allows us to prove tight concentration.

Fix $x_t \in C^i_j$. If x_t is the first point in C^i_j, then $\Pr[E_t = 1] = 0$. Otherwise, suppose there is a distinct $x_s \in C^i_j$ that was selected before x_t. Because no scale changes occur in the time interval, $x_s \in S_t$ and $d(x_t, S_t) \leq d(x_t, x_s)$. By the triangle inequality, $d(x_t, x_s)^2 \leq 2d(x_t, c^i)^2 + 2d(c^i, x_s)^2 \leq 2^{j+2}\gamma_i$. This implies that

$$Pr[E_t = 1] \leq \frac{d(x_t, S_t)^2 (\log \frac{2t}{\delta})^4}{R_t} \leq \frac{2^{j+2}\gamma_i (\log \frac{2n}{\delta})^4}{R_t} \leq \frac{8k(2^j \gamma_i)(\log \frac{2n}{\delta})^4}{\mathcal{L}_k(X_n)}, \tag{10}$$

with the last inequality holding since $\frac{\mathcal{L}_k(X_n)}{2k} \leq R_t$.

We will apply a standard martingale generalization of Bernstein's theorem (e.g. Habib et al. (1998), Theorem 3.8), which states that if the random variables E_s are zero-one valued, and the maximum possible variance of $\sum_{t=q}^n E_t$ is $\hat{v}$, then for all $\lambda > 0$, $\Pr[\sum_{t=q}^n E_t \geq \lambda + \mathbb{E}[\sum_{t=q}^n E_t]] \leq \exp(\frac{-\lambda^2/2}{\hat{v} + \lambda/3})$. First, we have the following bound on the expectation:

$$\begin{aligned}
\mathbb{E}[\sum_{t=q}^{n} E_t] &\leq \sum_{i=1}^{k} \sum_{j=0}^{\log n+1} \sum_{x_t \in C_i^j} \mathbb{E}[E_t] \\
&\leq \frac{8k(\log \frac{2n}{\delta})^4}{\mathcal{L}_k(X_n)} \sum_{i=1}^{k} \sum_{j=0}^{\log n+1} |C_j^i| 2^j \gamma_i \\
&\leq \frac{8k(\log \frac{2n}{\delta})^4}{\mathcal{L}_k(X_n)} \sum_{i=1}^{k} \left[\sum_{x \in C_0^i} \gamma_i + \sum_{j=1}^{\log n+1} \sum_{x \in C_j^i} 2d(x, c^i)^2 \right] \\
&\leq \frac{8k(\log \frac{2n}{\delta})^4}{\mathcal{L}_k(X_n)} \sum_{i=1}^{k} \left[\mathcal{L}(C^i) + 2\mathcal{L}(C^i) \right] \\
&= 24k(\log \frac{2n}{\delta})^4.
\end{aligned}$$

The main manipulations we make come from the fact that $2^{j-1}\gamma_i < d(x, c^i)^2 \leq 2^j \gamma_i$ for all $j \geq 1$, $x \in C_j^i$, and from the definition of $\mathcal{L}(C^i)$.

Next, since each E_t is zero-one valued, the variance of any E_t is at most $\sup \Pr[E_t = 1 | E_q, \ldots, E_{t-1}]$ which can be upper bounded by (10). Thus, $\hat{v} \leq \sum_{t=q}^{n} \sup \Pr[E_t = 1 | E_q, \ldots, E_{t-1}] \leq 24k \log(\frac{2n}{\delta})^4$ (using the same steps as the expectation bound). Applying the martingale form of Bernstein's theorem with $\lambda = 24k \log(\frac{2n}{\delta})^4 (\log \frac{2n}{\delta} - 1)$, we have

$$\begin{aligned}
\Pr\left[\sum_{t=q}^{n} E_t > t + \mathbb{E}\left[\textstyle\sum_{t=q}^{n} E_t \right] \right] &\leq \exp\left(\frac{-\lambda^2/2}{\hat{v} + \lambda/3} \right) \\
&\leq \exp\left(\frac{-\lambda^2/2}{8k \log^4 \frac{2n}{\delta} (\log \frac{2n}{\delta} + 2)} \right) \\
&\leq \exp\left(-\frac{24^2 k (\log \frac{2n}{\delta} - 1)^2}{16(\log \frac{2n}{\delta} + 2)} \right) \\
&\leq \exp\left(-9 \log \frac{2n}{\delta} \right) \\
&\leq \left(\frac{\delta}{2n} \right)^9 \leq \frac{\delta}{165n^2}.
\end{aligned}$$

This completes the proof. ■

To complete the proof of Proposition 12, we observe that if R_n becomes larger than $\frac{\mathcal{L}_k(X_n)}{k}$, it could not have been set this high by Line 12 (as $\frac{w_n^2}{128k}$ is controlled by Proposition 8). Thus, it must be the case that R_n was doubled at time n, and that many points were selected since the last scale change, since the counter F resets in between scale changes. This gives the exact conditions for Lemma 19, and we are able to bound the probability that R_n becomes large.

Proof (Proposition 12)

Let A_n be the event that $R_n > \frac{\mathcal{L}_k(X_n)}{k}$ and $R_t \leq \frac{\mathcal{L}_k(X_t)}{k}$ for all $t < n$—i.e. n is the minimal time for which the property $R_n \geq \frac{\mathcal{L}_k(X_n)}{k}$ holds. The events A_k are pairwise disjoint, so we have

$$\Pr[\exists n : R_n > \tfrac{\mathcal{L}_k(X_n)}{k}] = \sum_{n=k}^{\infty} \Pr[A_n]$$

Observe that A_n holds only if R_n increased via lines 12 or 19. In line 12, it could have been set to $\frac{w_n^2}{128k}$, but this quantity is at most $\frac{\mathcal{L}_k(X_n)}{k}$ by Proposition 8.

Thus, A_n holds only if line 19 is executed at time n, so $F_{n-1} + 1 > 25k \log^5 \frac{2n}{\delta}$. Let $q(n)$ be the largest time less than n for which $F_{q(n)} = 0$. The above conditions imply that if A_n holds, then:

1. During the interval $[q(n), n]$, at least $25k \log^5 \frac{2n}{\delta}$ centers are taken, because the counter increases every time a center is taken. Denote this event by A_n^1.
2. No scale changes occur in $[q(n), n]$, because scale changes reset the counter F to 0, and thus $q(n)$ is at least the time of the last scale change. Denote this event by A_n^2.
3. For all $t \in [q(n), n-1]$, we have $R_t = \frac{R_n}{2}$ because no scale changes occur, and the only time Line 19 can execute in this interval is at time n. This implies $\frac{\mathcal{L}_k(X_n)}{2k} \leq R_t$ for all $t \in [q(n), n-1]$. Denote this event by A_n^3.

Conditioning on the value of $q(n)$ above, we have for any n,

$$\begin{aligned}
\Pr[A_n] &= \sum_{q=k}^{n} \Pr[A_n | q(n) = q] \Pr[q(n) = q] \\
&\leq \max_{k \leq q \leq n} \Pr[A_n | q(n) = q] \quad \text{(because } \Pr[q(n) = \cdot] \text{ sum to 1)} \\
&\leq \max_{k \leq q \leq n} \Pr[A_n^1 \wedge A_n^2 \wedge A_n^3 | q(n) = q] \\
&\leq \max_{k \leq q \leq n} \Pr[A_n^1 | q(n) = q, A_n^2, A_n^3]
\end{aligned}$$

However, the inner term $\Pr[A_n^1 | q(n) = q, A_n^2, A_n^3]$ is controlled by Lemma 19 to be at most $\frac{1}{165n^2}$. Thus,

$$\sum_{n=k}^{\infty} \Pr[A_n] \leq \sum_{n=k}^{\infty} \frac{\delta}{165n^2} \leq \frac{\delta}{165} \frac{\pi^2}{6} \leq \frac{\delta}{100}.$$

■

Appendix C. Analysis of Other Clustering Settings

Based on the significant influence of other clustering settings to our algorithm (especially the no-substitution setting), it is natural to ask how well our algorithm performs in other settings. To this end, we consider the streaming and no-substitution settings.

Streaming: Observe that Online_Cluster enjoys a small streaming cost, as $\sum_{t=1}^{n} d(x_t, S_n)^2 \leq \sum_{t=1}^{n} d(x_t, S_t)^2$, and the latter cost is bounded by Propositions 11 and 12. Since the analysis for the center complexity remains unchanged, we immediately have Corollary 4 (stated in the introduction).

Thus, Online_Cluster serves as a reasonable choice for online streaming – its only drawback is that it outputs more centers (by a poly$(\log n)$ factor), and has a higher approximation factor than other algorithms in this setting.

C.1. The No-Substitution Setting

Online_Cluster cannot be directly applied to the no-substitution setting because it violates the defining rule – it deletes centers. To remedy this, we simply amend Online_Cluster to proceed exactly as it does *without* performing any of its deletions – that is, the only change necessary is to delete lines 10 and 11 of Algorithm 1. The resulting algorithm adheres to the no-substitution paradigm, since it only updates its centers at time t by either adding or not adding x_t. We refer to the resulting algorithm as No_Sub_Cluster, which is presented in Algorithm 2

Algorithm 2: The main algorithm, $No\_Sub\_Cluster(X, k, \delta)$.

1 $S_k \leftarrow \{x_1, \ldots, x_k\}$ `Initial set of centers`
2 $R_k, F \leftarrow 0$
3 $\tau_1 \leftarrow k+1, i \leftarrow 2$
4 $(Z_k, w_k) \leftarrow online\_k\_centers\_update(X_k)$
5 **for** $t = k+1, k+2, \ldots$ **do**
6 $\quad (Z_t, w_t) \leftarrow online\_k\_centers\_update(x_t)$ `Scale approximation`
7 $\quad$ **if** $w_t > 16 w_{\tau_{i-1}} \sqrt{t}$ **then** `Scale change detected`
8 $\quad\quad \tau_i \leftarrow t$
9 $\quad\quad R_t \leftarrow \frac{w_t^2}{128k}, F \leftarrow 0$
10 $\quad\quad i \leftarrow i+1$
11 $\quad$ **else**
12 $\quad\quad S_t \leftarrow S_{t-1}, R_t \leftarrow R_{t-1}$
13 $\quad$ With probability $\min\left\{1, \frac{d(x_t,S_t)^2 \log^4 \frac{2t}{\delta}}{R_t}\right\}$, $S_t = S_t \cup \{x_t\}$ `Center selection`
14 $\quad F \leftarrow F + \mathbf{1}_{x_t \text{ is selected}}$
15 $\quad$ **if** $F > 25k \log^5 \frac{2t}{\delta}$ **then**
16 $\quad\quad R_t \leftarrow 2R_t, F \leftarrow 0$
17 **end**

Here, the set of centers S_t is only ever changed in line 13, in which the point x_t is potentially added. The k-centers clustering step is now only used to estimate the scale, w_t, which in turn helps tune the parameter R_t.

We now restate and prove Corollary 5.

Corollary 5 *Let X be an arbitrary data sequence, k be a positive integer, and δ satisfy $0 < \delta < 1$. Suppose we run $No\_Sub\_Cluster(X, k, \delta)$. Let S_t denote the centers outputted at time t and $\mathcal{M}_t$ denote the total amount of memory used at the end of time t. Then with probability at least $1 - \delta$ over the randomness of $No\_Sub\_Cluster$, for all integers $n \geq 1$, the following hold:*

1. *(Approximation Factor)* $\sum_{t=2}^{n} d(x_t, S_t)^2 = O\left(\mathcal{L}_k(X_n)\right)$.

2. *(Center Complexity)* $|S_n| = O(kOC_{k+1}(X_n)\log^6 \frac{n}{\delta})$*, where* $OC_{k+1}(X_n)$ *is the lower bound parameter introduced in Bhattacharjee and Moshkovitz (2021).*

3. *(Memory and Time Complexity) Each step uses* $O(kdOC_k \log^6 \frac{n}{\delta})$ *time and memory.*

Proof Part 1. is a direct implication of the proof of Theorem 1. To adapt the proof to the no-substitution setting, we can use exact analogs of Propositions 11 and 12 that hold for No_Sub_Cluster. The fact that no deletions occur does not effect these propositions, and it is not difficult to see that all of the lemmas employed in the proof carry over.

Part 3. is an implication of part 2. as the bulk of the memory is the center complexity (we only require an additional $O(k)$ points to be stored for the k centers clustering, and $O(1)$ values to be stored for R_t, F_t, w_t). We now turn towards proving part 2.

First, observe by the proof of Theorem 1 (specifically the discussion of center complexity and memory) that with high probability, for all n, only $O\left(k\log^6 \frac{n}{\delta}\right)$ are selected between scale changes occurring before n. It follows that if ℓ_n denotes the number of scale changes before n, then with probability at least $1-\frac{\delta}{4}$ over the randomness of No_Sub_Cluster, $|S_n| = O\left(\ell_n k \log^6 \frac{n}{\delta}\right)$. It consequently suffices to bound ℓ_n in terms of $\mathrm{OC}_k(X_n)$. We now review the definition of OC_k.

Definition 20 *Bhattacharjee and Moshkovitz (2021) Let* $S = \{x_1, x_2, \ldots, x_n\}$ *be any set of points. Then for any* $p > 0$*,* $OC_p(S)$ *is the length of the longest sequence of points,* $y_1, y_2, \ldots, y_m$ *that are elements of* S *such that each point has distance to the previous points that is more than double the maximum diameter of a cluster when the previous points are optimally (with respect to their diameters) partitioned into* $p-1$ *clusters. More precisely, for all* $p \le i \le m$*,*

$$d(y_i, \{y_1, \ldots, y_{i-1}\}) > 2 \min_{C^1 \cup C^2 \cdots \cup C^{p-1} = S} \max_{1 \le j \le k} diam(C^j).$$

Bhattacharjee and Moshkovitz (2021) both gave upper and lower bounds on the center complexities of no-substitution algorithms based upon this parameter. Our goal is to relate the number of scale changes before time n, ℓ_n, to this complexity measure, $\mathrm{OC}_{k+1}(X_n)$. More precisely, it suffices to show that there exists a constant C such that,

$$\ell_n \le C \cdot \mathrm{OC}_{k+1}(X_n), \tag{11}$$

as substituting this into the bound above will prove part 2.

Our strategy for proving Equation 11 will be to use the scale changes in X_n to construct a sequence of points of length $\Omega(\ell_n)$ that satisfies the conditions given in Definition 20. To this end, let w_t^* denote the optimal k-centers clustering loss at time t. Since Charikar et al. (1997) provides an 8-approximation to the optimal k-center cost, it follows that $w_t^* \le w_t \le 8w_t^*$. As a result, it follows that $1 \le i \le \ell_n - 1$ satisfy that

$$w^*_{\tau_{i+1}} \ge \frac{w_{\tau_{i+1}}}{8} \ge 2\sqrt{\tau_{i+1}} w_{\tau_i} \ge 2\sqrt{t} w^*_{\tau_{i+1}}. \tag{12}$$

Next, for any time $t \ge 7$, suppose $t \le \tau_i < \tau_{i+1}$. We claim that there exists t' with $t \le t' < \tau_{i+2}$ such that

$$d(x_{t'}, \{x_1, \ldots, x_t\}) > 4w_t^*.$$

To show this, assume towards a contradiction that this isn't the case. It follows by the triangle inequality that any k-clustering of $X_t = \{x_1, \ldots, x_t\}$ with a k-centers loss of w_t^* can be used as a k-clustering of $X_{\tau_{i+1}}$ with loss at most $5w_t^*$. However, this contradicts Equation 12, as

$$w^*_{\tau_{i+1}} \geq 2\sqrt{\tau_{i+1}} w^*_{\tau_i} > 5w^*_t,$$

because $\tau_{i+1} \geq 7$ means $2\sqrt{\tau_{i+1}} > 5$. Thus, we are guaranteed that t' always exists.

Next, we leverage the construction of t' to construct a sequence of times, $t_1, t_2, \ldots, t_{\ell'}$ satisfying that $d(x_{t_i}, \{x_{t_1}, \ldots, x_{t_{i-1}}\}) > 4w^*_{t_{i-1}}$ for all i and $\ell' \geq \Omega(\ell_n)$. It suffices to show that $x_{t_1}, x_{t_2}, \ldots, x_{t_{\ell'}}$ satisfies the criteria given in Definition 20 when $p = k + 1$. However, this is a consequence of the fact that the minimum diameter from clustering a set of points into k parts is at most double the optimal k-centers cost, and this finishes the proof. ■

C.2. Comparison with Other No-Substitution Algorithms

	Approx. Ratio	Center Complexity	Aux. Memory	Loss Function
LSS16	$O(\log n)$	$O(k \log n \log \gamma^*)$	$O(1)$	$\sum_{t=1}^n d(x_t, S_t)^2$
BR20	$O(1)$	$O(k \log n \log \gamma^*)$	$O(k)$	$\sum_{t=1}^n d(x_t, S_n)^2$
BM21	$O(k^3)$	$O(k \log k \log n \cdot \mathrm{OC}_k(X))$	$O(n)$	$\sum_{t=1}^n d(x_t, S_n)^2$
Our result	$O(1)$	$O(k \log^6 n \cdot \mathrm{OC}_{k+1}(X))$	$O(k)$	$\sum_{t=1}^n d(x_t, S_t)^2$

Table 1: Comparison of existing no-substitution algorithms with our algorithm. The memory column contains memory in addition to the selected centers. The factor, $\mathrm{OC}_k(X)$ is the lower bound parameter introduced by Bhattacharjee and Moshkovitz (2021), and the term γ^* is the aspect ratio, the ratio between the distances between the furthest two points and the closest two points in the stream. Bhattacharjee and Moshkovitz (2021) implies that $\mathrm{OC}_k(X) = O(\log \gamma^*)$, with γ^* being potentially far larger.

We summarize existing clustering algorithms in Table 1, with LSS16, BR20, and BM21 denoting Liberty et al. (2016), Bhaskara and Rwanpathirana (2020), and Bhattacharjee and Moshkovitz (2021) respectively. For each algorithm, we list their approximation ratio, which is the factor by which their loss differs from the optimal k-means cost in hindsight, their center complexity, which is the expected number of centers chosen at time n, and their additional memory, which measures the amount of auxiliary memory needed for their algorithm to be executed.

In addition, several algorithms in this setting relax the no-substitution cost from $\sum_{t=1}^n d(x_t, S_t)^2$ to $\sum_{t=1}^n d(x_t, S_n)^2$. The latter expression is a smaller cost function which only measures the cost at a given time n by using the resulting centers, S_n, in hindsight. Because centers can never be deleted, we have that $\sum_{t=1}^n d(x_t, S_t)^2 \geq \sum_{t=1}^n d(x_t, S_n)^2$ implying that any upper bound on the former bounds the latter. However, the reverse does not necessarily hold.

As it can be seen from Table 1, our algorithm matches the approximation ratio and the additional memory of the best known algorithms. Furthermore, it is the only one to do so for the more complex cost function, $\sum_{t=1}^n d(x_t, S_t)^2$. With regards to center complexity, our algorithm outperforms Bhaskara and Rwanpathirana (2020) and Liberty et al. (2016) as it avoids a dependence on the *aspect ratio*, γ^*, and instead uses the more refined term $\mathrm{OC}_{k+1}(X)$ (Definition 20). As indicated

by Bhattacharjee and Moshkovitz (2021), the term $\text{OC}_{k+1}(X)$ is potentially much smaller than the aspect ratio, and at best the two are comparable by a constant factor.

The only algorithm with a better center complexity than ours is that of Bhattacharjee and Moshkovitz (2021), which, in addition to having fewer log factors, also uses the *true* lower bound parameter, $\text{OC}_k(X_n)$, rather than $\text{OC}_{k+1}(X_n)$, which would technically correspond to using $k+1$ centers. However, for many datasets, $\text{OC}_{k+1}(X_n)$ is not much larger than $\text{OC}_k(X_n)$ owing to the fact the $(k-1)$-fold and k-fold diameters of complex datasets are typically not massively different.

On the other hand, our algorithm has significant improvements over that of Bhattacharjee and Moshkovitz (2021) as it achieves a true $O(1)$-approximation (rather than $O(k^3)$) and is far more efficient in terms of the additional memory required. It also uses the more difficult cost function, $\sum_{t=1}^{n} d(x_t, S_t)^2$. In their algorithm, Bhattacharjee and Moshkovitz (2021) utilize a full offline k-clustering of the entire dataset at every timestep, thus requiring all data to be memorized in auxiliary memory.