Streaming k-means on Well-Clusterable Data

Vladimir Braverman[*] Adam Meyerson[†] Rafail Ostrovsky[‡] Alan Roytman[§]
Michael Shindler[¶] Brian Tagiku[‖]

Abstract

One of the central problems in data-analysis is k-means clustering. In recent years, considerable attention in the literature addressed the *streaming* variant of this problem, culminating in a series of results (Har-Peled and Mazumdar; Frahling and Sohler; Frahling, Monemizadeh, and Sohler; Chen) that produced a $(1 + \varepsilon)$-approximation for k-means clustering in the streaming setting. Unfortunately, since optimizing the k-means objective is Max-SNP hard, all algorithms that achieve a $(1 + \varepsilon)$-approximation must take time exponential in k unless P=NP.

Thus, to avoid exponential dependence on k, some additional assumptions must be made to guarantee high quality approximation and polynomial running time. A recent paper of Ostrovsky, Rabani, Schulman, and Swamy (FOCS 2006) introduced the very natural assumption of *data separability*: the assumption closely reflects how k-means is used in practice and allowed the authors to create a high-quality approximation for k-means clustering in the non-streaming setting with polynomial running time even for large values of k. Their work left open a natural and important question: are similar results possible in a *streaming* setting? This is the question we answer in this paper, albeit using substantially different techniques.

We show a near-optimal streaming approximation algorithm for k-means in high-dimensional Euclidean space with sublinear memory and a single pass, under the same data separability assumption. Our algorithm offers significant improvements in both space and running time over previous work while yielding asymptotically best-possible performance (assuming that the running time must be fully polynomial and P $\neq$ NP).

The novel techniques we develop along the way imply a number of additional results: we provide a high-probability performance guarantee for online facility location (in contrast, Meyerson's FOCS 2001 algorithm gave bounds only in expectation); we develop a constant approximation method for the general class of semi-metric clustering problems; we improve (even without σ-separability) by a logarithmic factor space requirements for streaming constant-approximation for k-median; finally we design a "re-sampling method" in a streaming setting to convert any constant approximation for clustering to a $[1 + O(\sigma^2)]$-approximation for σ-separable data.

1 Introduction

In this paper, we consider the problem of Euclidean k-means in the streaming model. Points in Euclidean space are read sequentially; when the data stream finishes, we must select k of these to designate as *facilities*. Our cost is the sum of squared distances from each point in the stream to its nearest facility.

A series of recent results [21, 15, 12, 10] produced $1 + \varepsilon$ approximations for streaming k-means. The general approach first appeared in the paper of Har-Peled and Mazumdar in STOC 2004 [21]. They used the concept of a (k, ε)-coreset: a weighted set of points such that any set of k facilities has within $1 + \varepsilon$ of the same cost on the original points or the coreset. Subsequent results improved the time and space bounds for computing coresets. In 2005, Frahling and Sohler [15] designed a new way to construct coresets based on grids. Two years later, Chen [10] designed a new way to generate coresets by randomly sampling from rings around an approximate set of facilities. Later in the same year, Feldman, Monemizadeh, and Sohler [12] used the concept of a weak coreset (due to [6]), where the size of the coreset is independent of n.

While these recent results claimed a $1 + \varepsilon$ approximation for streaming k-means, this requires producing an exact solution on the coreset itself, which takes time

[*] Computer Science Department, UCLA, vova@cs.ucla.edu. Supported in part by NSF grants 0830803, 0916574.

[†] Computer Science Department, UCLA, awm@cs.ucla.edu. Research partially supported by NSF CIF Grant CCF-1016540.

[‡] Computer Science and Mathematics Departments, UCLA, rafail@cs.ucla.edu. Research partially supported by IBM Faculty Award, Xerox Innovation Award, the Okawa Foundation Award, Intel, Teradata, NSF grants 0830803, 0916574, BSF grant 2008411 and U.C. MICRO grant.

[§] Computer Science Department, UCLA, alanr@cs.ucla.edu. Research partially supported by NSF CIF Grant CCF-1016540.

[¶] Computer Science Department, UCLA, shindler@cs.ucla.edu

[‖] Computer Science Department, UCLA, btagiku@cs.ucla.edu

$2^{\tilde{O}(k/\varepsilon)}$. When k is part of the input, this is exponential time, and it cannot be substantially improved since the objective is Max-SNP hard to optimize [7].

In this paper, we are interested in algorithms with truly polynomial runtimes. We seek to produce good approximations while optimizing space and runtime requirements. Since we cannot obtain $1+\varepsilon$ in polynomial time, we will make the natural assumption of data separability, introduced by Ostrovsky, Rabani, Schulman, and Swamy [31]; this closely reflects how k-means is used in practice and allowed the authors to create a good approximation in the non-streaming setting. Our main result is a streaming algorithm where n data points arrive one at a time, which produces a set of k means while making only a single pass through the data. We guarantee that the space requirement and processing time per point are logarithmic in n, and we produce an approximation factor of $1+O(\varepsilon)+O(\sigma^2)$ when the original data is σ-separable. While it is possible to modify the prior coreset-based approaches to obtain similar approximation bounds, our algorithm improves substantially on both space and time requirements. In fact our algorithm requires less space (by a factor of $\log n$) than the best previous *constant* approximation for the problem. We give both results in expectation and with high probability; our results are compared to previous coreset-based results for k-means in table 1.

The techniques that we develop along the way establish additional results: we provide a high-probability performance guarantee for online facility location (Meyerson's results [30] gave bounds only in expectation); we develop a constant approximation method for the general class of semi-metric clustering problems; we improve (even without σ-separability) by a logarithmic factor space requirements the previous best streaming algorithm for k-median; finally we show a novel "resampling method" in a streaming setting to reduce any constant approximation for clustering to $1+O(\sigma^2)$.

1.1 Related Work

The k-means problem was considered as early as 1956 by Steinhaus [34]. A simple local search heuristic for the problem was proposed in 1957 by Lloyd [27]. The heuristic begins with k arbitrarily chosen points as facilities. At each stage, it allocates the points X into clusters (each point assigned to closest facility) and then computes the center of mass for each cluster. These centers of mass become the new facilities for the next phase, and the process repeats until the solution stabilizes. Lloyd's algorithm has a rich history including psychologists in 1959-67 [36] and from 1960 to the modern day in computer science literature [29, 28, 11, 26, 16, 17, 22, 35, 38, 14, 13, 20, 8, 23, 2, 32, 33, 25, 31]. Unfortunately, Lloyd's algorithm has no provable approximation bound, and arbitrarily bad examples exist. Furthermore, the worst-case running time is superpolynomial [3]. Despite these drawbacks, Lloyd's algorithm (frequently known simply as `k-means`) remains common in practice.

The best polynomial-time approximation factor for k-means is by Kanungo, Mount, Netanyahu, Piatko, Silverman, and Wu [25]. They base their result on the k-median algorithm of Arya, Garg, Khandekar, Meyerson, Munagala, and Pandit [5]. Both papers use local search; the k-means case produces a $9+\varepsilon$ approximation. However, Lloyd's experimentally observed runtime is superior, and this is a high priority for real applications.

Ostrovsky, Rabani, Schulman and Swamy [31] observed that the value of k is typically selected such that the data is "well-clusterable" rather than being an arbitrary part of the input. They defined the notion of σ-separability, where the input to k-means is said to be σ-separable if reducing the number of facilities from k to $k-1$ would increase the cost of the optimum solution by a factor $\frac{1}{\sigma^2}$. They designed an algorithm with approximation ratio $1+O(\sigma^2)$. They also showed that their notion of σ-separability is robust and generalizes a number of other intuitive notions of "well-clusterable" data. The main idea of their algorithm is a randomized seeding technique which guarantees (with high probability) one initial facility belonging to each optimum cluster. They then perform a "ball k-means" step (Lloyd-like re-clustering) using only points which are near facilities. Subsequently, Arthur and Vassilvitskii [4] showed that the same procedure produces an $O(\log k)$ approximation for arbitrary instances of k-means.

When a k-means type algorithm is run in practice, the goal is to group the data based on a natural clustering. Balcan, Blum, and Gupta [7] use this observation to extend the notion of σ-separability to η-closeness: two clusterings are η-close if they disagree on only η fraction of the points, and an instance of the problem has the (c,η) property if any c-approximation is η-close to the target clustering for that instance. Their main contribution is to show how to use an existing constant approximation to modify a solution on an agreeable dataset to be a better solution. When the (c,η) property assumption holds, they are able to find very accurate approximations to the subjective correct clustering. In particular, any instance of k-means that has a $(1+\alpha,\eta)$-property can be clustered to be $O(\eta/\alpha)$ close to the target. However, their approach is memory intensive and not amenable to direct adaptation to the streaming model.

Each of these algorithms assumed that the entire input was available for processing in any form the algorithm designer needed. Our work focuses instead

Result	Space Requirements (points)	Runtime	PROB
[21] + [31]	$O(k(\log^{2d+2} n\varepsilon^{-d}))$	$O_d(n(k^5 + \log^2(k\varepsilon^{-1})))$	EXP
[15] + [31]	$O(((\log\Delta + \log n)^3 k^2 \log^4 \Delta)\varepsilon^{-2d-6})$	$O_d(n \log^2 \Delta(\log\Delta + \log n))$	EXP
Ours	$O(k\varepsilon^{-1} \log n)$	$O_d(nk \log n)$	EXP
[12] + [31]	$O(k^2\varepsilon^{-5} \log^{10} n)$	$O_d(nk^2\varepsilon^{-1} \log^2 n)$	WHP
[10] + [31]	$O(d^2k^2\varepsilon^{-2} \log^8 n)$	$O_d(nk \log^2 n\ polylog(k\varepsilon^{-1}))$	WHP
Ours	$O(k\varepsilon^{-1} \log^2 n)$	$O_d(nk \log n)$	WHP

Table 1: Streaming $1 + O(\varepsilon) + O(\sigma^2)$ approximations to k-means

on the streaming model, where the set of points X to cluster is extremely large and the algorithm is required to make only a single in-order pass through this data. This is typically used to model the case where the data must be read in a circumstance that lacks random access, such as a large amount of data stored on a hard disk.

The early work on streaming k-service clustering focused on streaming k-median. In 2000, Guha, Mishra, Motwani, and O'Callaghan [19] produced an $O(2^{1/\varepsilon})$ approximation for streaming k-median using $O(n^{\varepsilon})$ memory. Their algorithm reads the data in blocks, clustering each using some non-streaming approximation, and then gradually merges these blocks when enough of them arrive. An improved result for k-median was given by Charikar, O'Callaghan, and Panigrahy in 2003 [9], producing an $O(1)$ approximation using $O(k \log^2 n)$ space. Their work was based on guessing a lower bound on the optimum k-median cost and running $O(\log n)$ parallel versions of the online facility location algorithm of Meyerson [30] with facility cost based on the guessed lower bound. When these parallel calls exceeded the approximation bounds, they would be terminated and the guessed lower bound on the optimum k-median cost would increase.

A recent result for streaming k-means, due to Ailon, Jaiswal, and Monteleoni [1], is based on a divide and conquer approach, similar to the k-median algorithm of Guha, Meyerson Mishra, Motwani, and O'Callaghan [18]. It uses the result of Arthur and Vassilvitskii [4] as a subroutine, finding $3k \log k$ centers and producing an approximation ratio of 64 with probability at least 1/4 in a non-streaming setting. By dividing the input stream and running this repeatedly on pieces of the stream, they achieve an $O(2^{O(1/\varepsilon)} \log k)$ approximation using $O(n^{\varepsilon})$ memory.

1.2 High Level Ideas

Our goal is to produce a fully polynomial-time streaming approximation for k-service clustering. A natural starting point is the algorithm of Charikar, O'Callaghan, and Panigrahy [9]; however their result as stated applies only to the k-median problem. Since their algorithm depends heavily on calls to the online facility location algorithm of Meyerson [30], we first consider (and improve) results for this problem.

We produce new high probability bounds on the performance of online facility location, showing that the algorithm achieves within constants of its expected behavior with probability $1 - \frac{1}{n}$ (Theorem 3.1). To achieve this result, we inductively bound the probability of any given service cost being obtained prior to opening a facility in each of a collection of facility-less regions. We combine this with deterministic bounds on the service cost subsequent to opening a facility in the local region, and with Chernoff bounds on the number of facilities opened. Coupling our result with the algorithm of Charikar, O'Callaghan, and Panigrahy [9] improves our memory bound and processing time per point by a $\Theta(\log n)$ factor. Our analysis extends to cases where the triangle inequality holds only approximately, allowing us to apply the streaming algorithm to k-means as well. This yields the first streaming constant-approximation for k-means and k-median to store only $O(k \log n)$ points in memory (Theorem 3.2).

The execution of the algorithm of [9] is divided into phases, each of which corresponds to a "guess" at the optimum cost value. Each phase induces overhead to merge the existing clusters from the previous phase. The number of these phases is bounded by $O(n)$; we show that a modification of the algorithm along with an appropriate choice of constants can guarantee that each phase processes at least $k(1 + \log n)$ new points from the data stream, thus reducing the number of phases to $O(n/k \log n)$. This reduction improves the overall running time to $O(nk \log n)$.

Next, we would like to improve our approximation result to an FPTAS for the important case of Euclidean k-means. This is hard in general, as the problem is Max-SNP hard [7]. We instead make the σ-separability assumption of Ostrovsky, Rabani, Schulman, and Swamy [31] and show that we can obtain a $1 + O(\varepsilon) + O(\sigma^2)$ approximation using space for $O(\frac{k}{\varepsilon}\log n)$ points and polynomial time.

The first step is to consider applying a ball k-means step to our constant-approximation; this involves selecting the points which are much closer to one of our facilities than to any other (the "ball" of that facility) and computing the center of mass on those points. We show that given any $O(1)$ approximation to k-means, applying the ball k-means step will reduce the approximation factor to $1 + O(\sigma^2)$. The idea is that the optimum facilities for such an instance must be far apart; any constant-approximation must include a facility close to each of the optimum ones. Combining these facts gives a one-to-one mapping between our facilities and optimums, and we show that the points which are very close to each of our facilities must therefore belong to distinct optimum ones. This would enable us to produce a $1 + O(\sigma^2)$ approximation to k-means via two passes through the stream – the first pass would run the algorithm of Charikar, O'Callaghan, and Panigrahy [9] with our modifications, then the second pass would run the ball k-means step.

Of course, we wish to compute our entire solution with only one pass through the data. To do this, we prove that sampling works well for computing center of mass. A random sample of constant size (independent of the size of the cluster) provides a constant approximation (Theorem 4.1). Our goal is thus to produce a suitable random sample of the points belonging to each of the "balls" for our final ball k-means step.

Unfortunately, we do not know what our final cluster centers will be until the termination of the stream, making it difficult to sample uniformly from the balls. Instead, we show that the clusters from our solution are formed by adding points one at a time to clusters and by merging existing clusters together. This process permits us to maintain at all times a random sample of the points belonging to each of our clusters (section 4). Of course, randomly sampling from the points in these clusters is not the same as randomly sampling from the balls in the ball k-means step. However, we then show that the set we are actually sampling from (our cluster about a particular facility) and the set we "should be" sampling from (the points which are much closer to this particular facility than any other one of our facilities) are roughly (within constants) the same set of points, and that as the separability value σ approaches zero, these sets of points converge and become effectively identical (Theorem 5.2).

Putting it all together, our overall result maintains a sample of size $\frac{1}{\varepsilon}$ from each of our clusters at all times. The number of clusters will never exceed $O(k \log n)$, so the total memory requirement is $O(\frac{k}{\varepsilon}\log n)$ points for a chosen constant ε. The approximation factor for our final solution is $1 + O(\varepsilon) + O(\sigma^2)$ for σ-separable data, and our overall running time is $O(nk \log n)$. While this result holds in expectation, we also give a similar result which holds with high probability (at least $1 - \frac{1}{n}$) in Appendix C. Our space requirement for the high probability result is $O(\frac{k}{\varepsilon}\log n \log(nd))$, and by applying the result of Johnson and Lindenstrauss [24] we can reduce this to $O(\frac{k}{\varepsilon}\log^2 n)$. We also note that the value of σ need not be known to our algorithm at runtime.

We stress that our result improves over *all* previous streaming algorithms for k-means (or k-median) in the memory requirement and running time, while obtaining very good approximation results provided the data set is "well-clusterable" (as per [31]).

1.3 Our Techniques vs. Prior Work Our improvement of the analysis from Meyerson's online facility location result [30] uses similar techniques to the original paper. As before, the optimum clusters are divided up into "regions" based on proximity to the optimum center, and arguments are made about the cost prior to and subsequent to opening a facility in each region. Extending this approach to handle approximate triangle inequality is straightforward. The main new idea involves producing a high-probability bound, specifically on the service cost paid prior to opening facilities in each region. Here we use induction to produce an upper bound on the actual probability of paying at least a given cost prior to opening the facilities; by setting the target probability appropriately, we can show that the chance of exceeding the expected cost by more than a constant is exponentially small in the number of regions. Combining this with a straightforward application of Chernoff bounds (for the number of facililties) completes the result.

While our overcall algorithm bears some similarity to the result of Charikar, O'Callaghan, and Panigrahy [9], our techniques are quite different. They break their process into phases, then show that each phase "succeeds" with reasonably high probability. They then require substantial work to bound the number of phases to be linear in the number of points. In contrast, we show that we only require "success" of a randomized algorithm at a particular critical phase; prior phases are always guaranteed to have bounded cost. This allows a substantial improvement, and unlike their work, our performance and success probability do not depend on the number of phases. Nonetheless, bounding the number of phases is important for the running time. We obtain a better-than-linear bound by simply requiring each phase to read in at least a logarithmic number of new points; this analysis is much simpler and enables us to perform a simple matching at the end of each phase (reducing the number of facilities sufficiently) rather than approximating k-means on the facilities of the prior phase. Of course, our ideas about using sampling and a "ball k-means" step to improve the approximation were not part of [9] at all, although the general idea (without the sampling/streaming aspect) appeared in Ostrovsky, Rabani, Schulman, and Swami [31].

1.4 Definitions

Definition 1.1. (k-service clustering) *We are given a finite set X of points, a possibly infinite set Y (with $X \subseteq Y$) of potential facilities, and a cost function $\delta : X \times Y \to \Re^+$. Our goal is to select $K \subseteq Y$ of size k to be designated as facilities, so as to minimize $\sum_{i \in X} \min_{j \in K}\{\delta(i,j)\}$. The cost function is known as the service cost to connect a point to a facility.*

This encapsulates a family of problems, including *k-median*, where δ is a metric on space Y, and *k-means*, where Y is Euclidian space and δ is the square of Euclidian distance. The related *facility location* is formed by removing the constraint that $|K| = k$, replacing it with a facility cost f, and adding $f|K|$ to the objective function. Note that the k-means service costs satisfy 2-approximate triangle inequality:

Definition 1.2. (α-Approximate Triangle Inequality) *If, for any points a, b, c the following applies: $\alpha[\delta(a,b) + \delta(b,c)] \geq \delta(a,c)$, then we say that α-approximate triangle inequality is satisfied.*

Definition 1.3. (σ-separable dataset) *A set of input data for the k-service clustering problem is said to be σ-separable if the ratio of the optimal k-service clustering cost to the optimal $k-1$-service clustering cost is at most σ^2.*

This captures the notion that the kth facility must be meaningful for the clustering to be as well. This has been applied to k-means by [31].

2 Streaming Algorithm for k-means

In this section, we will provide a constant-approximation for k-service clustering, for any instance in which $X \subseteq Y$ and where α-approximate triangle inequality applies to δ.

Algorithm 1 summarizes our entire process for streaming k-service clustering. It takes as input a data stream known to contain n points and a value k for the number of desired means. The algorithm is defined in terms of constants β, γ; we will give precise values for these constants in section 3.2. The algorithm as described also requires a (non-streaming) $O(1)$ approximation to k-service clustering to be available as a subroutine. One candidate algorithm for this when running k-means is the approximation of Kanungo *et al* [25].

At several points in our algorithm, we refer to placing points at the front of the data stream. An easy way to implement this is to maintain a stack structure. When placing an item at the front of the stream, push it to the stack. When reading from the stream, check first if the stack is empty: if it is not, read by popping from the stack. If the stack is empty, read from the stream as normal. This also allows us to place items with weight on the stream, and we consider each item from the stream to be of weight one.

Algorithm 1 One pass, constant approximation k-service clustering algorithm.

```
 1: $L_1 \leftarrow 1$
 2: $i \leftarrow 1$
 3: while solution not found do
 4:    $K \leftarrow \emptyset$
 5:    cost $\leftarrow 0$
 6:    $f \leftarrow L_i/(k(1 + \log n))$
 7:    while there are points still in the stream do
 8:       $x \leftarrow$ next point from stream
 9:       $y \leftarrow$ facility in $K$ that minimizes $\delta(x,y)$
10:       if probability $\min\{\frac{weight(x)\cdot\delta(x,y)}{f}, 1\}$ then
11:          $K \leftarrow K \cup \{x\}$
12:       else
13:          cost $\leftarrow$ cost + weight$(x) \cdot \delta(x,y)$
14:          weight$(y) \leftarrow$ weight$(y)$ + weight$(x)$
15:       if cost $> \gamma L_i$ or $|K| > (\gamma - 1)(1 + \log n)k$ then
16:          break and raise flag
17:    if flag raised then
18:       push facilities in $K$ onto stream
19:       $L_{i+1} \leftarrow \beta L_i$
20:       $i \leftarrow i + 1$
21:    else
22:       Cluster $K$ to yield exactly $k$ facilities
23:       Declare solution found
```

3 A constant approximation

Our algorithm is quite similar to that of Charikar, O'Callaghan, and Panigrahy [9]. Both approaches are based on running online facility location [30] (lines 5-12 in our algorithm) with facility costs based on gradually improving lower bounds on the optimum cost. We will show an improved online facility location analysis, which enables us to run only a single copy of online facility location (instead of $O(\log n)$ copies as in [9]) while maintaining a high probability of success. We will also show that we do not require the randomized

Symbol	Meaning	Symbol	Meaning
Δ	diameter of dataset	d	dimensionality of data
k	desired number of means	ε	parameter of algorithm
σ	separability of dataset	n	number of points

Table 2: Notation used in this paper

online facility location algorithm to "succeed" at every phase, only at the critical final phase of the algorithm; this allows us to improve our approximation factor from that of [9]. Finally, we will show that we can bound the number of phases by $O(n/k \log n)$ rather than just $O(n)$; this improves the running time of our algorithm substantially from that of [9], obtaining $O(nk \log n)$ time.

3.1 Improved Analysis of Online Facility Location The online facility location algorithm of [30] is used implicitly in lines 7-16 of algorithm 1 and works as follows. We are given a facility cost f. As each point arrives, we measure the service cost δ for assigning that point to the nearest existing facility. With probability $\min\{\frac{\delta}{f}, 1\}$ we create a new facility at the arriving point. Otherwise, we assign the point to the nearest existing facility and pay the service cost δ.

Theorem 3.1. *Suppose we run the online facility location algorithm of [30] with $f = \frac{L}{k(1+\log n)}$ where $L \leq OPT$ and that the service costs satisfy α-approximate triangle inequality. Then the expected total service cost is at most $(3\alpha+1)OPT$ and the expected number of facilities generated by the algorithm is at most $(3\alpha+1)k(1+\log n)\frac{OPT}{L}$. Further, with probability at least $1-\frac{1}{n}$ the service cost is at most $(3\alpha + \frac{2e}{e-1})OPT$ and the number of facilities generated is at most $(6\alpha+1)k(1+\log n)\frac{OPT}{L}$.*

Proof. Consider each optimum facility c_i^*. Let C_i^* be the points assigned by OPT to c_i^*, A_i^* be the total service cost of optimum cluster C_i^* and $a_i^* = A_i^*/|C_i^*|$. Let δ_p^* be the optimum service cost for point p. We divide the optimum cluster C_i^* into regions S_i^j for $j \geq 1$ where $|S_i^j| = |C_i^*|/2^j$ and all the points in S_i^j have optimum service cost at most the optimum service cost of points in S_i^{j+1}. This will probably produce "fractional" points (i.e. points which are split between many regions); however this does not affect the analysis. Let A_i^j be the total optimum service cost of points in S_i^j, such that $\Sigma_j A_i^j = A_i^*$.

For each region S_i^j we may eventually open a facility at some point q in this region. Once we do so, subsequent points p arriving in the region must have bounded service cost of at most $\alpha(\delta_p^* + \delta_q^*)$. Since $q \in S_i^j$ and all points in S_i^j have smaller optimum service cost than points in S_i^{j+1}, we can conclude that $\delta_q^* \leq A_i^{j+1}/|S_i^{j+1}|$. Summing the resulting expression over all points in S_i^j gives us service cost of at most $\alpha(A_i^j + 2A_i^{j+1})$. Summing this over all the regions give service cost at most $3\alpha A_i^*$ subsequent to the arrival of the first facility in the regions. Note that this is a deterministic guarantee.

It remains to bound the service cost paid prior to the first facility opened in each region. In expectation, each region will pay at most f in service cost before opening a facility. Further, regions labeled $j > \log n$ contain only one point in total, and the overall service cost for this point cannot exceed f. Thus the expected total service cost is at most $k(1+\log n)f + 3\alpha\Sigma_i A_i^* \leq L + 3\alpha OPT$. Since $L \leq OPT$, this gives expected service cost at most $1 + 3\alpha$ times optimum. For the high probability guarantee, let $P[x,y]$ be the probability that given x regions which do not yet have a facility, the remaining service cost due to points in these regions arriving prior to the region having a facility is more than yf. We will prove by induction that $P[x,y] \leq e^{x-y(\frac{e-1}{e})}$, where e is the base of the natural log. Note that this is immediate for $x = 0$ and for very small values of y (i.e. $y \leq x\frac{e}{e-1}$). To prove this is always true, suppose that x is the smallest value where this can be violated, and y is the smallest value where it can be violated for this x. Thus $P[x,y] > e^{x-y(\frac{e-1}{e})}$. Suppose that the first request in one of the facility-less regions computes a service cost of $\delta > 0$. Then we have: $P[x,y] = \frac{\delta}{f}P[x-1,y] + (1-\frac{\delta}{f})P[x, y-\frac{\delta}{f}]$.

The first term corresponds to opening a facility at this point, thus reducing the number of facility-less regions by one; the second term corresponds to paying the service cost. Applying the definition of x and y:

$$\begin{aligned} e^{x-y(\frac{e-1}{e})} &< P[x,y] \\ &\leq \frac{\delta}{f}e^{x-1-y(\frac{e-1}{e})} + (1-\frac{\delta}{f})e^{x-(y-\frac{\delta}{f})(\frac{e-1}{e})} \end{aligned}$$

Dividing both sides by the left-hand expression leaves $1 < \frac{\delta}{ef} + (1-\frac{\delta}{f})e^{\frac{\delta}{f}\frac{e-1}{e}}$. This provides a contradiction.

Thus the probability that the total cost prior to facilities over all the regions is more than $\frac{e}{e-1}(2k)(1+$

$\log n)f$ is at most $P[k(1+\log n), \frac{e}{e-1}(2k)(1+\log n)] \leq e^{-k(1+\log n)} \leq \frac{1}{2n}$. Substituting for f gives the bound claimed.

We now consider the facility count. The first in each region gives us a total of $k(1+\log n)$ facilities; this is a deterministic guarantee. Now we must bound the number of facilities opened in the various regions subsequent to the first. Each point p has probability δ_p/f to open a new facility, where δ_p is the service cost when p arrives. Note that we already had a deterministic guarantee that for points arriving after a facility in their region, we have $\Sigma_p \delta_p \leq 3\alpha OPT$. Thus we have a sum of effectively independent Bernoulli trials with expectation at most $\frac{3\alpha OPT}{f} = 3\alpha k(1+\log n)\frac{OPT}{L}$. We can now apply Chernoff bounds for the result.

3.2 Analysis of Algorithm 1 We first need to define the constants β, γ. Let c_{OFL} be the constant factor on the service cost obtained from online facility location with high probability from Theorem 3.1, and let k_{OFL} be such that online facility location guarantees to generate at most $k_{OFL}k(1+\log n)\frac{OPT}{L}$ facilities. Note that c_{OFL}, k_{OFL} are constants which depend on α and on the desired "high probability" bound for success. We now define the constants as $\beta = 2\alpha^2 c_{OFL} + 2\alpha; \gamma = \max\{4\alpha^3 c_{OFL}^2 + 2\alpha^2 c_{OFL}, \beta k_{OFL} + 1\}$.

We will assume that $c_{OFL} \geq 2\alpha$ from this point on; this is implicit in the proof of Theorem 3.1 and we can always replace c_{OFL} with a larger value since it is a worst-case guarantee.

Define a phase in Algorithm 1 to be a single iteration of the outermost loop. Within each phase i, we maintain a lower bound L_i on OPT and run the online facility location algorithm using facility cost $f = \frac{L_i}{k(1+\log n)}$. We try reading as many points as we can until either our service cost grows too high (more than γL_i) or we have too many facilities (more than $(\gamma-1)k(1+\log n)$). At this point, we conclude that our lower bound L_i is too small, so we increase it by a factor β and start a new phase.

In a phase, we pay at most $f = L_i/k(1+\log n)$ for a weighted point and there are at most $(\gamma-1)k(1+\log n)$ weighted points from the previous phase. Our service cost for these points can be at most $(\gamma-1)L_i$ so we successfully cluster all weighted points in a phase. Thus at the start of each phase, the stream looks like some weighted points from only the preceeding phase followed by unread points. Additionally, we can show that all these points on the stream have a clustering with service cost comparable to OPT.

Lemma 3.1. *Let X' be any subset of points in the stream at the start of phase i. Then the total service cost of the optimum k-service clustering of X' is at most $\alpha \cdot OPT + \gamma\left(\frac{\alpha^2}{\beta-\alpha}\right)L_i$.*

Proof. Consider an original point $x \in X$. Say that $y \in K$ *represents* x in phase ℓ if y is the assigned facility for x or for x's phase $\ell-1$ representative. Note that the weight of $y \in K$ is the number of points it represents. Moreover, once a point x becomes represented in phase ℓ, it is represented for all future phases.

At the start of phase i, the stream looks like the weighted facilities from phase $i-1$ followed by unread points. Let us examine our cost if we use the optimum facilities (for X) to serve all of these points. Fix a point $x \in X$ and let us bound the service cost due to this point. Let j be the phase in which x was first clustered. Let $y_j, y_{j+1}, \ldots, y_{i-1}$ be x's respective representatives in phases j up through $i-1$. Then the service cost due to x will be $\delta(y_{i-1}, y^*)$ where y^* is the cheapest optimum facility for y_{i-1}. By α-approximate triangle inequality

$$\begin{aligned}\delta(y_{i-1}, y^*) &\leq \alpha\delta(x, y^*) + \alpha\delta(x, y_{i-1}) \\ &\leq \alpha\delta(x, y^*) + \sum_{\ell=2}^{i-j} \alpha^\ell \delta(y_{i-\ell}, y_{i-\ell+1}) \\ &\quad + \alpha^{i-j}\delta(x, y_j)\end{aligned}$$

Thus, summing over all points x in or represented by points in X', and noting that our service cost in phase ℓ is bounded by $\gamma L_\ell \leq \gamma L_i \frac{1}{\beta^{i-\ell}}$, gives a total service cost of at most

$$\begin{aligned}cost &\leq \alpha \cdot OPT + \gamma\alpha L_i \sum_{\ell=1}^{i-1}\left(\frac{\alpha}{\beta}\right)^{i-\ell} \\ &= \alpha \cdot OPT + \gamma\left(\frac{\alpha^2}{\beta-\alpha}\right)L_i\end{aligned}$$

The above lemma shows that there exists a low cost clustering for the points at each phase, provided we can guarantee that $L_i \leq OPT$. Call the last phase where $L_i \leq OPT$ the *critical phase*. We will show that we in fact terminate at or before the critical phase with high probability.

Lemma 3.2. *With probability at least the success probability of online facility location from Theorem 3.1, Algorithm 1 terminates at or before the critical phase.*

Proof. Let i be the critical phase, and let OPT_i be the optimum cost of clustering all the points (weighted or not) seen on the stream at the start of phase i. By Lemma 3.1 and the fact that $OPT \leq \beta L_i$, we have

$$
\begin{aligned}
OPT_i &\leq \alpha \cdot OPT + \gamma\left(\frac{\alpha^2}{\beta-\alpha}\right) L_i \\
&\leq \left(\alpha\beta + \gamma\frac{\alpha^2}{\beta-\alpha}\right) L_i.
\end{aligned}
$$

Theorem 3.1 guarantees the online facility location algorithm yields a solution with at most $\beta k_{OFL}(1+\log n)k$ facilities and of cost at most $c_{OFL}OPT_i$ with high probability. Our definitions for β, γ guarantee that $c_{OFL}OPT_i \leq \gamma L_i$. In addition, our definition for γ guarantees that $(\gamma-1)k(1+\log n) \geq \beta k_{OFL}k(1+\log n)$. Thus if online facility location "succeeds," the critical phase will allow the online facility location algorithm to run to completion.

Corollary 3.1. *With high probability (same as that for online facility location), Algorithm 1 completes the final phase with a solution of cost at most $\frac{\alpha\beta\gamma}{\beta-\alpha} \cdot OPT$. Applying the values for the constants gives an approximation factor of $4\alpha^4 c_{OFL}^2 + 4\alpha^3 c_{OFL}$ provided $4\alpha^3 c_{OFL}^2 + 2\alpha^2 c_{OFL} \geq \beta k_{OFL} + 1$.*

Proof. Consider a point $x \in X$. As in the proof of Lemma 3.1, let $y_j, y_{j+1}, \ldots, y_{i-1}, y_i$ be x's respective representatives in phases j up through i. The service cost due to x is $\delta(x, y_i)$. By α-approximate triangle inequality, we can bound this by

$$\delta(x, y_i) \leq \alpha^{i-j}\delta(x, y_j) + \sum_{\ell=1}^{i-j} \alpha^\ell \delta(y_{i-\ell}, y_{i-\ell+1}).$$

Summing over all points x, and noting that our service cost in phase ℓ is bounded by γL_ℓ, combined with the knowledge that with high probability, we terminate at a phase where $L_i \leq OPT$, gives a total service cost of at most:

$$
\begin{aligned}
\text{cost} &\leq \alpha\gamma L_i + \alpha^2\gamma L_{i-1} + \alpha^3\gamma L_{i-2} + \cdots + \alpha^i \gamma L_1 \\
&\leq \alpha\gamma L_i \sum_{\ell=0}^{i-1}\left(\frac{\alpha}{\beta}\right)^\ell \\
&\leq \frac{\alpha\beta\gamma}{\beta-\alpha} \cdot OPT.
\end{aligned}
$$

However, this solution uses much more than k facilities. To prune down to exactly k facilities, we can use any non-streaming $O(1)$-approximation to cluster our final (weighted) facilities (line 22 of the algorithm). If this non-streaming clustering algorithm has an approximation ratio of c_{KS}, our overall approximation ratio increases to $(\alpha + 4\alpha^5 c_{OFL}^2 + 4\alpha^4 c_{OFL})c_{KS}$.

Theorem 3.2. *With high probability, our algorithm achieves a constant approximation to k-service clustering if α-approximate triangle inequality holds for fixed constant α. This uses exactly k facilities and stores $O(k \log n)$ points in memory.*

3.3 Pruning the Runtime

As presented, Algorithm 1 can see as many as $O(\log_\beta OPT)$ phases in expectation, which gives the runtime an undesireable dependence on OPT. We now show how to modify Algorithm 1 so that it has at most $O(n/k \log n)$ phases and running time bounded by $O(nk \log n)$.

Theorem 3.3. *For any fixed α, Algorithm 1 can be modified to run in $O(nk \log n)$-time.*

Proof. Consider any phase. The phase starts by reading the weighted facilities from the previous phase and paying a cost of at most $f = \frac{L_i}{k(1+\log n)}$ for each, at the end of which the cost is at most $(\gamma-1)L_i$. Each additional point gives us a service cost of at most $\frac{L_i}{k(1+\log n)}$, so the phase must read at least $k(1+\log n)$ additional unread points before it can terminate due to cost exceeding γL_i.

Now suppose that the phase ends due to having too many facilities without reading at least $k(1+\log n)$ additional points. Since each new point can create at most one facility, the previous phase must have had at least $(\gamma-2)k(1+\log n)$ facilities already. Consider an optimal k-service clustering over the set X' of all the weighted points during this phase. Let OPT' denote the total service cost of this solution and OPT'_r denote the optimum total service cost if we are instead restricted to only selecting points from X'. Note that by α-approximate triangle inequality, we have $OPT'_r \leq 2\alpha OPT'$. Thus, by Lemma 3.1, we have $OPT'_r \leq 2\alpha(\alpha + \gamma\frac{\alpha^2}{\beta-\alpha})OPT$.

Since OPT'_r is only allowed k facilities, it must pay non-zero service cost for at least $(\gamma-3)(1+\log n)k$ weighted points. Define the nearest neighbor function $\pi : X' \to X'$ where for each point $x \in X'$, $\pi(x)$ denotes closest other point (in terms of service costs) in X'. Then note that $\Delta_x = \text{weight}(x) \cdot \delta(x, \pi(x))$ gives a lower bound on the service cost for x if it is not chosen as a facility. Thus, the sum η of all but the k highest Δ_x gives a lower bound on OPT'_r. It follows that $\frac{\eta}{2\alpha(\alpha+\gamma\frac{\alpha^2}{\beta-\alpha})} \leq OPT$.

We will set L_i to the maximum of this new lower bound and βL_{i-1}, eliminate $k(1+\log n)$ facilities and increase service cost to at most L_i. This guarantees that the next time the number of facilities grows too large we will have read $\Omega(k \log n)$ new points, bounding the number of phases by $O(\frac{n}{k \log n})$.

Let $\hat{X} \subseteq X'$ denote the set of points with $\Delta_x \leq \eta[2\alpha(\alpha + \gamma\frac{\alpha^2}{\beta-\alpha})(1+\log n)k]^{-1}$. Suppose that $|\hat{X}| < 2k(1+\log n)$. The number of points which contribute to η is at least $(\gamma - 3)k(1+\log n)$, and at least $(\gamma-5)k(1+\log n)$ of these points would not belong to $\hat{X}$. Thus the sum of Δ_x for such points is bounded by $\frac{\eta(\gamma-5)}{2\alpha(\alpha+\gamma\frac{\alpha^2}{\beta-\alpha})} \leq \eta$. Canceling and solving this equation for γ yields $\gamma(1-\frac{2\alpha^3}{\beta-\alpha}) \leq 2\alpha^2+5$.

Plugging in the values for β and γ along with $c_{OFL} \geq 2\alpha$ and $\alpha \geq 1$ yields a contradiction. Thus it follows that $|\hat{X}| \geq 2k(1+\log n)$. We assume $|\hat{X}|$ even for simplicity of analysis. Some points in $\hat{X}$ have their nearest neighbor in $X' - \hat{X}$. For the remaining points in $\hat{X}$, consider the nearest neighbor graph induced by these points. This graph has no cycles of length 3 or longer. Thus, the graph is bipartite and we can find a vertex cover C of size at most $|\hat{X}|/2$. We can add additional points to C from $\hat{X}$ to get precisely $|\hat{X}|/2$ points. Note that all points in $\hat{X} - C$ have a nearest neighbor not in $\hat{X} - C$. Thus, we can remove $\hat{X} - C$ as facilities and increase our service cost by at most

$$\frac{\eta(1+\log n)k}{2\alpha(\alpha+\gamma\frac{\alpha^2}{\beta-\gamma})(1+\log n)k} = \frac{\eta}{2\alpha(\alpha+\gamma\frac{\alpha^2}{\beta-\alpha})} \leq L_i.$$

We can compute η in time $O(k^2\log^2 n)$, $\hat{X}$ in time $O(k\log n)$, and the vertex cover in time linear in $|\hat{X}|$ (using a greedy algorithm; note that it needn't be a minimum vertex cover). Additionally, all these can be computed using space to store $O(k\log n)$ points. The running time for reading a new (unweighted) point is $O(k\log n)$, so the total running time is the time to read unweighted points plus the overhead induced by starting new phases (and reading weighted points). Each of these is at most $O(nk\log n)$.

4 Maintaining samples during streaming k-means

Definition 4.1. *Let S be a set of cardinality n. Let $\Re^q = \{p \in S^q : \forall i,j \in [q] \quad p_i \neq p_j\}$. A random element E is a q-random sample without replacement from S if E has uniform distribution over $\Re^q$.*

First, we establish that we can maintain a uniform at random sample for each facility's service points. This is identical to the problem of sampling uniformly at random from a stream, as all the points assigned to the facility can be treated as a single stream. Methods for streaming sampling are well-known, see e.g., [37].

We must also show that we are able to maintain a uniformly at random sample from the union of two clusters, for the two times in our algorithm in which two clusters are combined, and that samples of optimal clusters are sufficient to find approximate centers of mass. The proofs of these appear in Appendix A.

Lemma 4.1. *There exists an algorithm that, given a stream X and q, maintains a q-random sample without replacement from X and uses $O(q)$ memory.*

Lemma 4.2. *Let S_1, S_2 be two disjoint sets. Given independent q-samples without replacement from S_1 and S_2 and $|S_1|, |S_2|$, it is possible to generate a q-sample without replacement from $S_1 \cup S_2$. The algorithm requires $O(q)$ time and $O(q)$ additional memory.*

Theorem 4.1. *Suppose we have a set X of points and are given some arbitrarily selected $Y \subseteq X$. If Z is a set of q points selected uniformly at random from Y (without replacement), then the center of mass for Z is a $(1+\frac{1}{q}-\frac{q-1}{q(|Y|-1)})(\frac{|X|}{|Y|})$-approximation to the optimum one-mean solution for X in expectation.*

There is an analog to this theorem that applies with high probability; it is detailed in Appendix C.

5 From Constant to Converging to One

Given a c-approximation to k-means (where c is constant) for a σ-separable point set, we now show that we can perform a single recentering step, called a *ball k-means step*, and obtain an approximation ratio of $(1+\sigma^2)$ which converges to one as σ approaches 0. While a full ball k-means step requires another pass through the point stream, we will establish that it is sufficient to use a smaller random sample of points.

Theorem 5.1. *Suppose we have a c-approximation to k-means, and an σ-separable data set where $\frac{1}{\sigma^2} > 2\gamma(c+1)+1$ for $\gamma \geq \frac{169}{4}$. Then we can apply a ball k-means step, by associating with each of our approximate means $\nu(i)$ the set of points $B_{\nu(i)}$, then computing a new mean $\nu(i)' = com(B_{\nu(i)})$. This yields an approximation to k-means which approaches one as σ approaches zero.*

Theorem 5.2. *Suppose we have a c-approximation to k-means for an σ-separable data set where $\frac{1}{\sigma^2} > 2\gamma(c+1)+1$ for $\gamma \geq \frac{169}{4}$. Additionally, suppose that instead of being given the entire point set, we are only given small random samples $Z_{\nu(i)}$ of size $\frac{1}{\varepsilon}$ of each cluster $C_{\nu(i)}$ in this approximate solution. Then we can apply a ball k-means step computing a new mean $\nu(i)'$ by computing the center of mass of $(B_{\nu(i)} \cap Z_{\nu(i)})$. This yields a $\Theta((1+\varepsilon)(1+\sigma^2 c))$-approximation to k-means which approaches one as ε and σ approach zero.*

References

[1] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In *NIPS*, 2009.

[2] Khaled Alsabti, Sanjay Ranka, and Vineet Singh. An efficient k-means clustering algorithm. In *Proc. 1st Workshop on High Performance Data Mining*, 1998.

[3] David Arthur and Sergei Vassilvitskii. How Slow is the k-means Method? In *SCG*, 2006.

[4] David Arthur and Sergei Vassilvitskii. k-means++: The Advantages of Careful Seeding. In *SODA*, 2007.

[5] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *STOC*, 2001.

[6] Mihai Bādoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *STOC*, 2002.

[7] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *SODA*, 2009.

[8] Geoffrey H. Ball. Data analysis in the social sciences: what about the details? In *AFIPS '65 (Fall, part I): Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I*, pages 533–559, New York, NY, USA, 1965. ACM.

[9] Moses Charikar, Liadan O'Callaghan, and Rina Panigrahy. Better streaming algorithms for clustering problems. In *STOC*, 2003.

[10] Ke Chen. On coresets for k-median and k-means clustering in metric and euclidean spaces and their applications. *SIAM J. Comput.*, 2009.

[11] Arthur Pentland Dempster, Nan McKenzie Laird, and Donald Bruce Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38, 1977.

[12] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *SCG*, 2007.

[13] Walter D. Fisher. On grouping for maximum homogeneity. *Journal of the American Statistical Association*, 53:789–798, 1958.

[14] Edward Forgey. Cluster analysis of multivariate data: efficiency vs. interpretability of classification. *Biometrics*, 21:768, 1965.

[15] Gereon Frahling and Christian Sohler. Coresets in dynamic geometric data streams. In *STOC*, 2005.

[16] Allen Gersho and Robert M. Gray. *Vector quantization and signal compression.* Kluwer, 1992.

[17] Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2384, October 1998.

[18] Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: Theory and practice. In *IEEE Transactions on Data and Knowledge Engineering (TDKE)*, 2003.

[19] Sudipto Guha, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams. In *FOCS*, 2000.

[20] David J. Hall and Geoffrey H. Ball. ISODATA: a novel method of data analysis and pattern classification. Technical report, Stanford Research Institute, 1965.

[21] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC '04*, pages 291–300, New York, NY, USA, 2004. ACM.

[22] Anil Kumar Jain, M Narasimha Murty, and Patrick Joseph Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3), September 1999.

[23] Robert C. Jancey. Multidimensional group analysis. *Australian Journal of Botany*, 14:127–130, 1966.

[24] W. Johnson and J. Lindenstrauss. Extensions of Lipschitz maps into a Hilbert space. In *Contemporary Mathematics*, 1984.

[25] Tapas Kanungo, David Mount, Nathan Netanyahu, Christine Piatko, Ruth Silverman, and Angela Wu. A local search approximation algorithm for k-means clustering. In *SCG*, 2002.

[26] Yoseph Linde, Andres Buzo, and Robert M. Gray. An algorithm for vector quantization design. *IEEE Transactions on Communication*, COM-28:84–95, January 1980.

[27] Stuart Lloyd. Least Squares Quantization in PCM. In *Special issue on quantization, IEEE Transactions on Information Theory*, volume 28, pages 129–137, 1982.

[28] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. on Math. Statistics and Probability*, pages 281–297, 1967.

[29] Joel Max. Quantizing for minimum distortion. *IEEE Transactions on Information Theory*, 1960.

[30] Adam Meyerson. Online facility location. In *FOCS*, 2001.

[31] Rafail Ostrovsky, Yuval Rabani, Leonard Schulman, and Chaitanya Swamy. The Effectiveness of Lloyd-Type Methods for the k-Means Problem. In *FOCS*, 2006.

[32] Dan Pelleg and Andrew Moore. Accelerating exact k-means algorithms with geometric reasoning. In *Proc. 5th ACM KDD*, pages 277–281, 1999.

[33] Steven J. Phillips. Acceleration of k-means and related clustering problems. In *ALENEX*, 2002.

[34] Hugo Steinhaus. Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci.*, C1. III vol IV:801–804, 1956.

[35] Laszlo Fejes Toth. Sur la representation d'une population infinie par un nombre fini d'elements. *Acta math. Acad. Sci. Hung.*, 10:76–81, 1959.

[36] Robert Choate Tryon and Daniel Edgar Bailey. *Cluster Analysis.* McGraw-Hill, 1970. Pages 147-150.

[37] Jeffrey S. Vitter. Random sampling with a reservoir. In *ACM Transactions on Mathematical Software*, 1985.

[38] Paul L. Zador. *Development and evaluation of procedures for quantizing multivariate distributions.* PhD thesis, Stanford University, 1963. Stanford University Department of Statistics Technical Report.

A Proofs of lemmas and Theorems from Section 4

For notational convenience, we use $com(A)$ to mean the center of mass for a point set A. Note the relation in any cluster of the difference in cost for replacing the center of mass with an arbitrary other point:

Fact A.1. $\sum_{x \in X} d^2(x, y) = |X| d^2(com(X), y) + \sum_x d^2(x, com(X))$

A.1 Proof of Lemma 4.1 For our purposes, it will be useful to prove the following alternative definition.

Lemma A.1. *Let S be a set and let $X_1, \ldots, X_q$ be the following random variables. X_1 is distributed uniformly on S, and, for $1 < i \leq q$ X_i is distributed uniformly over $S \setminus \{X_1, \ldots, X_{i-1}\}$. Then an ordered q-tuple $X = \langle X_1, \ldots, X_q \rangle$ is a q-random sample without replacement from S.*

Proof. Consider a fixed q-tuple $p = \langle p_1, \ldots, p_q \rangle \in \Re^q$. We shall show that $P(X = p) = \frac{1}{|\Re^q|} = \frac{1}{q! \binom{n}{q}}$.

$$
\begin{aligned}
&P(X = p) = \\
&\quad P(X_1 = p_1) \times P(X_2 = p_2 | X_1 = p_1) \\
&\quad \times \cdots \times P(X_q = p_q | \forall_{j<q} X_j = p_j).
\end{aligned}
$$

By the definition of X_1, $P(X_1 = p_1) = \frac{1}{n}$. By the definition of X_i and since $p_i \notin \{p_1, \ldots, p_{i-1}\}$, we have for all fixed $X_1, \ldots X_{i-1}$: $P(X_i = p_i | \forall_{j<i} X_j = p_j) = \frac{1}{n-i+1}$.
Thus, $P(X = p) = \prod_{i=1}^{q} \frac{1}{n-i+1} = \frac{1}{q! \binom{n}{q}}$.

A.2 Proof of Lemma 4.2 The algorithm is similar in spirit to the merging process in Merge Sort. Let $X = \{X_1, \ldots, X_q\}$ and $Y = \{Y_1, \ldots, Y_q\}$ be random samples from S_1 and S_2. Take the first element of X with probability equal to $\frac{|X|}{|X|+|Y|}$; otherwise, take from Y. Repeat this until q are chosen, and call the set formed by those taken Z.

The performance bounds follow from the description of the algorithm. To show the correctness, it is sufficient to show that Z_i is distributed uniformly over $(S_1 \cup S_2) \setminus \{Z_1, \ldots, Z_{j-1}\}$ and then use Lemma A.1. First, consider $i = 1$. By Lemma A.1, X_1 and Y_1 are distributed uniformly over S_1 and S_2 respectively. Consider arbitrary and fixed $w \in S_1 \cup S_2$. W.l.o.g., assume that $w \in S_1$. Then since the randomness of the algorithm is independent of X we have: $P(Z_1 = w) = \frac{n_1}{n_1+n_2} P(X_1 = w) = \frac{1}{n_1+n_2}$. Thus, Z_1 is distributed uniformly over $S_1 \cup S_2$.

Let $i > 1$; and consider any fixed $Z_1, \ldots, Z_{i-1}$. By Lemma A.1, X_{i_1} is distributed uniformly over $S_1 \setminus \{X_1, \ldots, X_{i_1-1}\}$ and Y_{i_2} is distributed uniformly over $S_1 \setminus \{Y_1, \ldots, Y_{i_2-1}\}$. Consider any fixed $w \in (S_1 \cup S_2) \setminus \{Z_1, \ldots, Z_{i-1}\}$. W.l.o.g., assume that $w \in S_1$. Since $\{Z_1, \ldots, Z_{j-1}\} = \{Y_1, \ldots, Y_{i_2-1}\} \cup \{X_1, \ldots, X_{i_1-1}\}$, it follows that $w \in S_1 \setminus \{X_1, \ldots, X_{i_1-1}\}$. Thus, we have

$$
\begin{aligned}
P(Z_i = w) &= \frac{n_1}{n_1+n_2} P(X_{i_1} = w) \\
&= \frac{n_1}{n_1+n_2} \left(\frac{1}{n_1}\right) \\
&= \frac{1}{n_1+n_2}.
\end{aligned}
$$

The values of n_1 and n_2 imply that the probability is uniform over $(S_1 \cup S_2) \setminus \{Z_1, \ldots, Z_{i-1}\}$. Indeed,

$$n_1 + n_2 = |S_1| - i_1 + 1 + |S_2| - i_2 + 1 = |S_1 \cup S_2| - i + 1.$$

Thus, we have shown that Z_1 is a uniform sample from $S_1 \cup S_2$ and for any $j > l$, Z_j is a sample from $(S_1 \cup S_2) \setminus \{Z_1, \ldots, Z_{j-1}\}$. The correctness follows from Lemma A.1.

A.3 Proof of Theorem 4.1

Lemma A.2. *Let X be a set of points and $Y \subseteq X$. Then $C(com(Y), X) \leq \frac{|X|}{|Y|} C(com(X), X)$.*

Proof. Let $C(a, A)$ be the one-means cost of using a as a mean for A. Let $d_1 = d(com(X), com(Y))$ and $d_2 = d(com(X-Y), com(X))$. From triangle inequality, $d(com(Y), com(X-Y)) \leq d_1 + d_2$. Applying **Fact A.1** repeatedly gives:

$$
\begin{aligned}
C(com(Y), X) = &C(com(Y), Y) \\
&+ C(com(X-Y), X-Y) \\
&+ |X-Y|(\delta_1 + \delta_2)^2
\end{aligned}
$$

$$
\begin{aligned}
C(com(X), X) = &C(com(Y), Y) \\
&+ |Y| d_1^2 \\
&+ C(com(X-Y), X-Y) \\
&+ |X-Y| d_2^2
\end{aligned}
$$

$$\frac{C(com(Y), X)}{C(com(X), X)} \leq \frac{|X-Y|(d_1+d_2)^2}{|Y| d_1^2 + |X-Y| d_2^2}$$

To maximize the ratio, we take the derivative with respect to d_2 and set the resulting expression to zero, obtaining $|Y| d_1^2 + |X-Y| d_2^2 = |X-Y|(d_1+d_2) d_2$; solving this yields $d_2 = \frac{|Y|}{|X-Y|} d_1$. Note that the other boundary conditions for the expression are at $d_2 = 0$ and d_2 tends to infinity, both of which easily satisfy the required inequality. Substitution gives:

$$\frac{C(com(Y), X)}{C(com(X), X)} \leq \frac{|X|^2/|X-Y|}{|Y| + (|Y|^2/|X-Y|)} \leq \frac{|X|}{|Y|}$$

By linearity of expectation, it is enough to show that the above holds in one-dimensional space. Applying **Fact A.1** gives us $C(com(Z), X) \leq |X| d^2(com(Z), com(X)) + C(com(X), X)$. We will need to bound the expected value of $d^2(com(Z), com(X))$. Since we can assume one dimensional space, we use the definition of center of mass to get:

$$E[d^2(com(Z), com(X)] = E[(\frac{\sum_{z \in Z} z}{|Z|} - \frac{\sum_{x \in X} x}{|X|})^2]$$

We can compute the square and use linearity of expectation, noticing that since the points of Z are uniformly chosen from Y, we have $E[(1/|Z|)\Sigma_{z \in Z} z] = (1/|Y|)\sum_{y \in Y} y$. We need to bound $E[(\sum_{z \in Z} z)^2]$. For each $y_1 \in Y$, there is a probability $|Z|/|Y|$ that this point appeared also in the randomly selected set Z. If so, we will obtain an expected contribution to the sum of squares which looks like $y_1^2 + y_1 E[\Sigma_{z \in Z - \{y_1\}} z | y_1 \in Z]$, where the latter term is just $\frac{|Z|-1}{|Y|-1} \sum_{y_2 \in Y, y_2 \neq y_1} y_2$. Summing these gives us:

$$E[(\sum_{z \in Z} z)^2] = \frac{|Z|}{|Y|} \sum_{y \in Y} y^2 + \frac{|Z|}{|Y|} \sum_{y_1 \in Y} \sum_{y_2 \in Y, y_2 \neq y_1} \frac{|Z|-1}{|Y|-1} y_1 y_2$$

We can rewrite this, adding and subtracting terms representing the sum of squared elements of Y, as:

$$E[(\sum_{z \in Z} z)^2] = \frac{|Z|}{|Y|}[(1 - \frac{|Z|-1}{|Y|-1}) \sum_{y \in Y} y^2 + \frac{|Z|-1}{|Y|-1} (\sum_{y \in Y} y)^2]$$

We have $C(com(Y), Y) = \Sigma_{y \in Y} y^2 - \frac{1}{|Y|}(\Sigma_{y \in Y} y)^2$, and we can substitute this to get:

$$E[(\sum_{z \in Z} z)^2] = \frac{|Z|}{|Y|}[(1 - \frac{|Z|-1}{|Y|-1}) C(com(Y), Y) + \frac{|Z|}{|Y|}(\sum_{y \in Y} y)^2]$$

We observe that $d^2(com(Y), X)$ can be formulated similarly to $d^2(com(Z), X)$, and when we combine the various terms we obtain the following bound:

$$E[d^2(com(Z), com(X)] = \frac{1}{|Y||Z|}[(1 - \frac{|Z|-1}{|Y|-1}) C(com(Y), Y)] + d^2(com(Y), com(X))$$

To compute the cost of $C(com(Z), X)$, we multiply by $|X|$ and add $C(com(X), X)$. We also observe that since $Y \subseteq X$, we will have $C(com(Y), Y) \leq C(com(X), X)$, and we apply theorem A.2 to reach:

$$E[C(com(Z), X)] \leq \frac{|X|}{|Y|}(1 + \frac{1}{|Z|} - \frac{|Z|-1}{|Z|(|Y|-1)}) C(com(X), X)$$

B Proofs from Section 5

B.1 Proof of Theorem 5.1 For each optimum mean i, let C_i^* be the points OPT assigns to i and let $\nu(i)$ be the closest mean to i in our c-approximate solution. We can show that i and $\nu(i)$ are in fact very close together:

LEMMA B.1. *For any i, we have $d(\nu(i), i) \leq \sqrt{\frac{2(c+1)OPT}{|C_i^*|}}$.*

Proof. Consider the points $S = \{x \in C_i^* \mid d(i, x) \leq d(i, \nu(i))\}$. For each $x \in S$, we can bound $d^2(i, \nu(i))$ using 2-approximate triangle inequality. Summing over all S gives

$$d^2(i, \nu(i)) \leq \frac{2}{|S|} \left(\sum_{x \in S} d^2(i, x) + \sum_{x \in S} d^2(x, \nu(i)) \right).$$

The first term in the right-hand side is bounded by $OPT - \sum_{x \notin S} d^2(i, x)$ whereas the second term is at most $c \cdot OPT$. Substituting and using the fact that $d^2(i, \nu(i)) < d^2(i, x)$ for all $x \notin S$ proves the claim.

Moreover, our point set being σ-separable implies that the optimum means must be fairly far apart:

LEMMA B.2. *In an σ-separable point set, any two optimum means i, j satisfy $d(i, j) \geq \sqrt{\frac{OPT}{\sigma^2 |C_i^*|} - \frac{OPT}{|C_i^*|}}$.*

Proof. For any two means i, j, we can always eliminate mean i and reassign any points in C_i^*. This produces a solution using $k-1$ means and, since i is the center of mass of C_i^*, increases cost by at most $|C_i^*| d^2(i, j)$. By σ-separability, the total cost of this solution must be at least $\frac{OPT}{\sigma^2}$ which gives the above bound on $d(i, j)$.

Lemmas B.1 and B.2 show that for sufficiently small σ, each optimum mean i has unique $\nu(i)$ which is much closer i than to any other optimum mean. In particular, if $\frac{1}{\sigma^2} > 2\gamma(c+1)+1$ for some γ to be specified later, it follows that for any optimum mean i, the next closest optimum mean is at least distance $\sqrt{\gamma}\sqrt{\frac{2(c+1)OPT}{|C_i^*|}}$ away and the closest mean $\nu(i)$ is at most $\sqrt{\frac{2(c+1)OPT}{|C_i^*|}}$ away.

Lemma B.3. *Define $B_{\nu(i)}$ to consist of all points x such that $2d(x,\nu(i)) \leq d(x,\nu(j))$ for any $j \neq i$. Then $B_{\nu(i)} \subseteq C_i^*$ and $|B_{\nu(i)}| \geq (1-\frac{9}{2(c+1)(\sqrt{\gamma}-5)^2})|C_i^*|$ when $\gamma \geq \frac{169}{4}$.*

Proof. We first show $B_{\nu(i)} \subseteq C_i^*$. Fix i, j and suppose $x \in B_{\nu(i)}$. By definition, $2d(x,\nu(i)) \leq d(x,\nu(j))$. Applying triangle inequalities gives $2d(x,i) \leq 2d(i,\nu(i)) + d(j,\nu(j)) + d(x,j)$. If $d(x,j) \leq d(x,i)$, then it will follow that $d(x,i) \leq 2d(i,\nu(i)) + d(j,\nu(j))$. Each of these is bounded according to Lemma B.1, so we can conclude that

$$\begin{aligned} d(i,j) &\leq 2d(x,i) \\ &\leq 4\sqrt{\frac{2(c+1)OPT}{|C_i^*|}} + 2\sqrt{\frac{2(c+1)OPT}{|C_j^*|}} \\ &\leq 6\sqrt{\frac{2(c+1)OPT}{\min\{|C_i^*|,|C_j^*|\}}}. \end{aligned}$$

However, Lemma B.2 implies that $d(i,j) \geq \sqrt{\frac{2\gamma(c+1)OPT}{\min\{|C_i^*|,|C_j^*|\}}}$. Provided that $\gamma > 36$, this is a contradiction. We conclude that $d(x,i) < d(x,j)$ and that therefore $x \in C_i^*$ and $B_{\nu(i)} \subseteq C_i^*$.

Notice that the service cost of points in C_i^* is at most OPT. By Markov's inequality, for any $\mu \geq \frac{1}{2(c+1)}$ there are at least $\left(1-\frac{1}{2(c+1)\mu}\right)|C_i^*|$ points of C_i^* within distance $\sqrt{\frac{2\mu(c+1)OPT}{|C_i^*|}}$ of i. For any such point x and optimum mean $j \neq i$, triangle inequality along with Lemmas B.1 and B.2 gives

$$\begin{aligned} d(x,\nu(j)) &\geq d(i,j) - d(i,\nu(i)) - d(j,\nu(j)) - d(x,\nu(i)) \\ &\geq (\sqrt{\gamma}-2)\sqrt{\frac{2(c+1)OPT}{\min\{|C_i^*|,|C_j^*|\}}} - d(x,\nu(i)) \end{aligned}$$

Setting $\mu = \frac{1}{9}(\sqrt{\gamma}-5)^2$ ensures that $d(x,\nu(j)) \geq 2d(x,\nu(i))$ and that $x \in B_{\nu(i)}$. However, this imposes the additional constraint that $\gamma \geq \frac{169}{4}$ in order to ensure that $\mu \geq \frac{1}{4} \geq \frac{1}{2(c+1)}$.

By Lemma B.3, we have $B_{\nu(i)} \subseteq C_i^*$ and $|B_{\nu(i)}| \geq |C_i^*|(1-\frac{9}{2(c+1)(\sqrt{\gamma}-5)^2})$. It follows from Theorem 4.1 that we obtain an approximation to k-means of ratio at worst $1+\frac{9}{2(c+1)(\sqrt{\gamma}-5)^2-9}$. As σ becomes smaller γ becomes larger, and thus the approximation ratio converges to one.

B.2 Proof of Theorem 5.2 We have shown that we can perform a ball k-means step on our approximate solution to achieve an approximation ratio which approaches 1 as σ approaches 0. While performing a full ball k-means step requires another pass through the point set, we can avoid this second pass if we are given a random sample of $\frac{1}{\varepsilon}$ points from each of the balls $B_{\nu(i)}$ and performing the Ball k-means step on just these sample points. By Theorem 4, this gives us an approximation ratio of $(1+\sigma-\frac{\frac{1}{\varepsilon}-1}{\frac{1}{\varepsilon}(|B_{\nu(i)}\cap C_{\nu(i)}|-1)})\frac{|C_i^*|}{|B_{\nu(i)}\cap C_{\nu(i)}|}$ within each cluster.

However, our algorithm only returns a random sample of q points in each of our clusters $C_{\nu(i)}$. Thus, we need to show that, in expectation, a constant fraction of these points are in $B_{\nu(i)}$. Indeed, we now show that this fraction approaches 1 and that $|B_{\nu(i)} \cap C_{\nu(i)}|$ approaches $|B_{\nu(i)}|$ as σ tends toward 0. Thus, our overall approximation ratio still converges to 1 as ε and σ approach 0.

We first give an upper bound on the number of points in $B_{\nu(i)}$ that aren't in $C_{\nu(i)}$. These points are never candidates in the randomly selected points from $C_{\nu(i)}$ and so may hurt our approximation if there are too many. Fortunately, we can prove that there is only a small number of them:

Lemma B.4. *$|B_{\nu(i)} - C_{\nu(i)}| \leq \frac{c}{8(\sqrt{\gamma}-2)^2(c+1)}|C_i^*|$.*

Proof. Consider $x \in B_{\nu(i)} - C_{\nu(i)}$ and let $\nu(j)$ be the mean such that $x \in C_{\nu(j)}$. Triangle inequality and the fact that $x \notin B_{\nu(j)}$ gives $d(x,\nu(j)) \geq d(i,j) - d(i,\nu(i)) - d(j,\nu(j)) - \frac{1}{2}d(x,\nu(j))$. Solving for $d(x,\nu(j))$, applying Lemmas B.1 and B.2 and squaring gives

$$\begin{aligned} d(x,\nu(j))^2 &\geq \left(2(\sqrt{\gamma}-2)\sqrt{\frac{2(c+1)OPT}{\min\{n_i,n_j\}}}\right)^2 \\ &\geq 4(\sqrt{\gamma}-2)^2\left(\frac{2(c+1)OPT}{|C_i^*|}\right). \end{aligned}$$

If we sum over all such x, we should get no more than $c \cdot OPT$ since we have a c-approximation. This bounds $|B_{\nu(i)} - C_{\nu(i)}|$ as desired.

We can now use Lemmas B.3 and B.4 to give a lower bound on fraction of $B_{\nu(i)}$ contained in $C_{\nu(i)}$. Accordingly, this fraction approaches 1 as σ diminishes, showing that roughly the entirety of $B_{\nu(i)}$ is in our sample space. We can also bound the cardinality of $B_{\nu(i)} \cap C_{\nu(i)}$ in terms of C_i^* which will become useful later.

Corollary B.1. *$|B_{\nu(i)} \cap C_{\nu(i)}| \geq \left(1 - \frac{c(\sqrt{\gamma}-5)^2}{4(\sqrt{\gamma}-2)^2(2(c+1)(\sqrt{\gamma}-5)^2-9)}\right)|B_{\nu(i)}|$.*

Corollary B.2. *$|B_{\nu(i)} \cap C_{\nu(i)}| \geq \left(1 - \frac{9}{2(c+1)(\sqrt{\gamma}-5)^2} - \frac{c}{8(\sqrt{\gamma}-2)^2(c+1)}\right)|C_i^*|$.*

Though roughly all of $B_{\nu(i)}$ lies in $C_{\nu(i)}$, there are other points in $C_{\nu(i)}$. If there are too many of these points, then we would expect that a very small fraction of the q' sampled points are actually in $B_{\nu(i)}$, driving our approximation ratio upwards. Thus, we must show that the number of these points tends towards 0.

Lemma B.5. *$|C_{\nu(i)} \cap B_{\nu(i)}| \geq \frac{\left(1-\frac{9}{2(c+1)(\sqrt{\gamma}-5)^2}-\frac{c}{8(\sqrt{\gamma}-2)^2(c+1)}\right)}{\left(1+\frac{c}{2(c+1)(\sqrt{\gamma}-2)^2}\right)}|C_{\nu(i)}|$.*

Proof. Consider an $x \in C_{\nu(i)} - B_{\nu(i)}$. Since $j \notin B_{\nu(i)}$, we must have $2d(x, \nu(i)) \geq d(x, \nu(j))$ for some j. Proceeding in a fashion similar to the proof of Lemma B.4 shows

$$|C_{\nu(i)} - B_{\nu(i)}| \leq \frac{c}{2(c+1)(\sqrt{\gamma}-2)^2}|C_i^*|.$$

Thus, we can bound the number of elements in our cluster by

$$\begin{aligned}|C_{\nu(i)}| &= |C_{\nu(i)} \cap B_{\nu(i)}| + |C_{\nu(i)} - B_{\nu(i)}| \\ &\leq \left(1 + \frac{c}{2(c+1)(\sqrt{\gamma}-2)^2}\right)|C_i^*|.\end{aligned}$$

Combining with Corollary B.2 gives the desired result.

C High Probability Guarantee

We now prove an analog of Theorem 4.1 to give a high probability guarantee. We do this by giving a series of lemmas.

Lemma C.1. *If $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{R}$ and $Y = Z - S$, where Z is a random sample from from X and $S = com(X) = \frac{1}{n}\sum_i x_i$, then $E[Y] = 0$ and $Var[Y] = \frac{1}{n}OPT$ (here OPT is the optimal 1-means solution for X).*

Proof. The expected value of Y is $E[Y] = E[Z - S] = E[Z] - S = S - S = 0$. The variance of Y is

$$\begin{aligned}Var[Y] &= E[Y^2] - E[Y]^2 \\ &= E[Y^2] \\ &= \frac{1}{n}\sum_{i=1}^{n}(x_i - S)^2 \\ &= \frac{1}{n}OPT\end{aligned}$$

Now consider taking the mean of q random samples from X, with replacement:

Lemma C.2. *If $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{R}$ and $Y = \frac{1}{q}(Z_1 + \cdots + Z_q) - S$, where each Z_i is a random sample from X (with replacement) and $S = com(X)$, then $E[Y] = 0$ and $Var[Y] = \frac{1}{qn}OPT$ (here OPT is the optimal 1-means solution for X).*

Proof. The expected value of Y is $E[Y] = E[\frac{1}{q}\sum_{i=1}^{q} Z_i - S] = \frac{1}{q}\sum_{i=1}^{q} E[Z_i] - S = S - S = 0$ (here we used Lemma C.1). Notice that we can rewrite Y as $Y = \frac{1}{q}\sum_{i=1}^{q} Y_i$, where the $Y_i = Z_i - S$ are independent random variables. Hence, the variance of Y (by Lemma C.1) is given by

$$Var[Y] = \frac{1}{q^2}\sum_{i=1}^{q} Var[Y_i] = \frac{1}{q^2}\frac{q}{n}OPT$$

Using the same notation, we now have the following constant probability bound:

Lemma C.3. *If $B = \frac{1}{q}(Z_1 + \cdots + Z_q)$, where $q = \frac{100}{\varepsilon}$, then $P[|B - S| \geq \sqrt{\frac{\varepsilon OPT}{n}}] \leq \frac{1}{100}$.*

Proof. By Chebyshev's inequality, we have:

$$\begin{aligned}P[|B - S| \geq \sqrt{\frac{\varepsilon OPT}{n}}] &= Pr[|Y| \geq \sqrt{\frac{\varepsilon OPT}{n}}] \\ &\leq \frac{nVar[Y]}{\varepsilon OPT} \\ &= \frac{1}{q\varepsilon} = \frac{1}{100}\end{aligned}$$

where Y is the same as in Lemma C.2.

We now take the median of means:

Lemma C.4. *Let $B_1, \ldots, B_t$ be independent random variables, where $t = O(\log nd)$ and each B_i is as in Lemma C.3. Let $B = median(B_1, \ldots, B_t)$. Then $P[|B - S| \geq \sqrt{\frac{\varepsilon OPT}{n}}] \leq \frac{1}{nd}$.*

Proof. This follows from a standard application of Chernoff bounds.

We now concentrate on points from $\mathbb{R}^d$ and give some notation and definitions. Let $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$ and let x_{ij} denote the j-th coordinate of x_i. Let $S = \frac{1}{n}\sum_i x_i$, and define $S_j = \frac{1}{n}\sum_i x_{ij}$ (so that $S = (S_1, \ldots, S_d)$). Define $OPT_j = \sum_{i=1}^{n}(x_{ij} - S_j)^2$. Notice that $OPT = \sum_{i=1}^{n}\sum_{j=1}^{d}(x_{ij} - S_j)^2 = \sum_{j=1}^{d} OPT_j$, where OPT is the optimal 1-means solution for X. We now come to our main lemma:

Lemma C.5. *Let $U \in \mathbb{R}^d$ be a vector with coordinates $U = (U_1, \ldots, U_d)$, where the U_i are independent and each has the same distribution as B from Lemma C.4 with respect to the set of j-th coordinates $\{x_{1j}, \ldots, x_{nj}\}$. Let $A = \sum_{i=1}^{n} d^2(U, x_i)$. Then $A \leq (1+\varepsilon)OPT$ with probability at least $1 - \frac{1}{n}$.*

Proof. By Lemma C.4, we know that $P[|U_j - S_j| \geq \sqrt{\frac{\varepsilon OPT_j}{n}}] \leq \frac{1}{dn}$. By applying the union bound, we know that with probability at least $1 - \frac{1}{n}$ we have $|U_j - S_j|^2 \leq \frac{\varepsilon OPT_j}{n}$ holds over all dimensions (i.e. for all $1 \leq j \leq d$). By Fact A.1, we know that $A = \sum_{i=1}^{n} d^2(U, x_i) = nd^2(U, S) + OPT$. We have the following upper bound on $d^2(U, S)$:

$$d^2(U,S) = \sum_{j=1}^{d}(U_j - S_j)^2 \leq \sum_{j=1}^{d}\frac{\varepsilon OPT_j}{n} = \frac{\varepsilon OPT}{n}$$

The lemma follows.

We can now apply similar methods used in our result for the expectation guarantee and achieve the same result for sufficiently large Y which are subsets of the set X.