

---

# DISTRIBUTED ALGORITHMS FOR EUCLIDEAN CLUSTERING

005 **Anonymous authors**

006 Paper under double-blind review

## ABSTRACT

011 We study the problem of constructing  $(1 + \varepsilon)$ -coresets for Euclidean  $(k, z)$ -  
012 clustering in the distributed setting, where  $n$  data points are partitioned across  
013  $s$  sites. We focus on two prominent communication models: the coordinator  
014 model and the blackboard model. In the coordinator model, we design a pro-  
015 tocol that achieves a  $(1 + \varepsilon)$ -strong coresnet with total communication complexity  
016  $\tilde{O}\left( sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log(n\Delta) \right)$  bits, improving upon prior work (Chen et  
017 al., NeurIPS 2016) by eliminating the need to communicate explicit point coordi-  
018 nates in-the-clear across all servers. In the blackboard model, we further reduce  
019 the communication complexity to  $\tilde{O}\left( s \log(n\Delta) + dk \log(n\Delta) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} \right)$   
020 bits, achieving better bounds than previous approaches while upgrading from  
021 constant-factor to  $(1 + \varepsilon)$ -approximation guarantees. Our techniques combine  
022 new strategies for constant-factor approximation with efficient coresnet construc-  
023 tions and compact encoding schemes, leading to optimal protocols that match both  
024 the communication costs of the best-known offline coresnet constructions and ex-  
025 isting lower bounds (Chen et al., NeurIPS 2016, Huang et. al., STOC 2024), up to  
026 polylogarithmic factors.

## 1 INTRODUCTION

030 Clustering is the process of partitioning a dataset by grouping points with similar properties and sep-  
031 arating those with differing properties. The study of clustering dates back to the 1950s (Steinhaus  
032 et al., 1956; MacQueen, 1967), and its many variants find applications in fields such as bioinfor-  
033 matics, combinatorial optimization, computational geometry, computer graphics, data science, and  
034 machine learning. In the Euclidean  $(k, z)$ -clustering problem, the input consists of a set  $X$  of  $n$   
035 points  $x_1, \dots, x_n \in \mathbb{R}^d$ , along with a cluster count  $k > 0$  and an exponent  $z > 0$ . The objective is  
036 to find a set  $\mathcal{C}$  of at most  $k$  centers that minimizes the clustering cost:

$$038 \min_{\mathcal{C} \subset \mathbb{R}^d, |\mathcal{C}| \leq k} \text{cost}(X, \mathcal{C}) := \min_{\mathcal{C} \subset \mathbb{R}^d, |\mathcal{C}| \leq k} \sum_{i=1}^n \min_{c \in \mathcal{C}} \|x_i - c\|_2^z.$$

040 When  $z = 1$  and  $z = 2$ , the  $(k, z)$ -clustering problem reduces to the classical  $k$ -median and  $k$ -means  
041 clustering problems, respectively.

043 Due to the substantial growth in modern datasets, focus has shifted toward large-scale computational  
044 models that can process data across multiple machines without requiring centralized access to the  
045 full dataset. The distributed model of computation has become a popular framework for handling  
046 such large-scale data. In the distributed setting, the points  $x_1, \dots, x_n$  of  $X$  are partitioned across  
047  $s$  different machines, and given an input accuracy parameter  $\varepsilon > 0$ , the goal is for the machines  
048 to collectively find a clustering  $\mathcal{C}$  of  $X$  with cost that is a  $(1 + \varepsilon)$ -multiplicative approximation of  
049 the optimal clustering of  $X$ , while minimizing the total communication between machines. As in  
050 other models, a set  $\mathcal{C}$  of  $k$  centers implicitly defines the clustering, since each point is assigned to  
051 its nearest center. Since transmitting the entire dataset or an explicit cluster label for each point  
052 would incur communication linear in  $n = |X|$ , it is more feasible to design protocols that exchange  
053 only succinct summaries, such as small sets of representative points or coresets. We also note that  
because of finite precision, input points are assumed to lie within the grid  $\{1, \dots, \Delta\}^d$ , which can  
be communicated efficiently using a small number of bits per coordinate.

054 A standard strategy for efficient distributed clustering is to have each machine construct a small  
 055 weighted subset of its local points, i.e., a coresset, that preserves the clustering cost for any choice of  
 056  $k$  centers. These local coressets can then be merged at a coordinator or hierarchically aggregated to  
 057 approximate the clustering objective over the full dataset. Naturally, smaller coressets correspond to  
 058 lower communication costs and faster centralized processing. In the offline setting, where the full  
 059 dataset  $X$  is available on a single machine without resource constraints, coresset constructions are  
 060 known that select  $\tilde{\mathcal{O}}\left(\min\left(\frac{1}{\varepsilon^2} \cdot k^{2-\frac{z}{z+2}}, \frac{1}{\min(\varepsilon^4, \varepsilon^{2+z})} \cdot k\right)\right)$  weighted points of  $X$  (Cohen-Addad  
 061 et al., 2021; 2022a;b; Huang et al., 2024)<sup>12</sup>. As is common, we assume  $\Delta = \text{poly}(n)$  so that  
 062 each coordinate can be stored in  $\mathcal{O}(\log(nd\Delta))$  bits, allowing efficient communication and storage.  
 063 Thus, offline coresset constructions have size independent of  $n$ , an essential feature given that modern  
 064 datasets often contain hundreds of millions of points.

## 066 1.1 OUR CONTRIBUTIONS

068 We study the construction of  $(1+\varepsilon)$ -coresets for Euclidean  $(k, z)$ -clustering in the distributed setting,  
 069 focusing on the coordinator and blackboard communication models.

071 **The coordinator model.** In the coordinator (message passing) model, the  $s$  sites, i.e., servers,  
 072 communicate only with the coordinator over private channels, using private randomness. Protocols  
 073 are assumed without loss of generality to be sequential, round-based, and self-delimiting, i.e., in each  
 074 round the coordinator speaks to some number of players and awaits their responses before initiating  
 075 the next round and all parties know when each message has been completely sent. Given  $\varepsilon > 0$ , the  
 076 goal is a protocol II where the coordinator outputs a  $(1 + \varepsilon)$ -coreset for Euclidean  $(k, z)$ -clustering  
 077 that minimizes the communication cost, i.e., the total number of bits exchanged in the worst case.  
 078 We achieve optimal communication protocols for clustering in this model, matching known lower  
 079 bounds from Chen et al. (2016); Huang et al. (2024):

080 **Theorem 1.1** (Communication-optimal clustering in the coordinator model, informal). *Given accuracy*  
 081 *parameter  $\varepsilon \in (0, 1)$ , there exists a protocol on  $n$  points distributed across  $s$  sites that produces*  
 082 *a  $(1 + \varepsilon)$ -strong coresset for  $(k, z)$ -clustering that uses  $\tilde{\mathcal{O}}\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log(n\Delta)\right)$  total*  
 083 *bits of communication in the coordinator model.*

085 We note that our results also yield the optimal bounds for  $k$ -median, matching Huang et al. (2024);  
 086 Bansal et al. (2024); for clarity, we omit this special case in the remainder of the paper. By com-  
 087 parison, the protocol of Balcan et al. (2013), combined with more recent coresset constructions such  
 088 as Cohen-Addad et al. (2021; 2022a); Bansal et al. (2024), achieves a  $(1 + \varepsilon)$ -strong coresset using  
 089  $\mathcal{O}\left(\frac{skd}{\min(\varepsilon^4, \varepsilon^{2+z})} \log(n\Delta)\right)$  bits of communication. At first glance, one might expect our bounds  
 090 to follow by straightforward generalizations of these approaches; however, as we explain in Ap-  
 091 pendix A, this is not the case. For example, our result shows that no points need to be communicated  
 092 “in the clear” across sites—there is no  $\mathcal{O}(sd \log(n\Delta))$  dependence in the communication. This is  
 093 surprising, since one might expect the coordinates of a constant-factor approximation to be broad-  
 094 cast to all users. Moreover, the  $\frac{1}{\varepsilon}$  factors in our bounds do not multiply either the number of sites  
 095  $s$  or the  $\log(n\Delta)$  cost of transmitting a single coordinate. Finally, our results extend to arbitrary  
 096 connected communication topologies, even when certain pairs of sites are not allowed to interact.  
 097 The formal statement is deferred to Theorem E.7 in Appendix E.

098 **The blackboard model.** Next, we consider the blackboard model of communication, where each of  
 099 the  $s$  sites can broadcast messages that are visible to all of the other sites. Formally in the blackboard  
 100 model, communication occurs through a shared public blackboard that is visible to all servers. Each  
 101 server again has access to private sources of randomness. Unlike the coordinator model, there is no  
 102 designated coordinator to relay messages; instead, any server may write messages directly onto the  
 103 blackboard. All servers can immediately observe the entire contents of the blackboard at any point  
 104 in time. As before, we assume without loss of generality that the protocol is sequential and round-  
 105 based, meaning that in each round, one or more servers write to the blackboard, and all servers can

<sup>1</sup>We write  $\tilde{\mathcal{O}}(f(n, d, k, \Delta, \frac{1}{\varepsilon}))$  to denote  $\mathcal{O}(f(n, d, k, \Delta, \frac{1}{\varepsilon})) \cdot \text{polylog}(f(n, d, k, \Delta, \frac{1}{\varepsilon}))$ .

<sup>2</sup>The bound further refines to  $\tilde{\mathcal{O}}\left(\min\left(\frac{k^{4/3}}{\varepsilon^2}, \frac{k}{\varepsilon^3}\right)\right)$  for  $k$ -median.

| Coordinator model  | Communication cost (bits)  |
|--|--|
| Merge-and-reduce, with (Bansal et al., 2024)<br>(Balcan et al., 2013), $z = 2$<br>(Balcan et al., 2013) in conjunction with (Bansal et al., 2024)<br>Theorem 1.1 (this work) | $\tilde{\mathcal{O}}\left(\frac{skd}{\min(\varepsilon^4, \varepsilon^{2+z})} \log(n\Delta)\right)$<br>$\mathcal{O}\left(\frac{d^2k}{\varepsilon^4} \log(n\Delta) + sdk \log(sk) \log(n\Delta)\right)$<br>$\mathcal{O}\left(\frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} \log(n\Delta) + sdk \log(sk) \log(n\Delta)\right)$<br>$\tilde{\mathcal{O}}\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log(n\Delta)\right)$ |
| Blackboard model   | Communication cost (bits)  |
| (Chen et al., 2016)<br>Theorem 1.2 (this work)   | $\tilde{\mathcal{O}}\left((s + dk) \log^2(n\Delta)\right)$<br>$\tilde{\mathcal{O}}\left(s \log(n\Delta) + dk \log(n\Delta) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$   |

Fig. 1: Table of  $(k, z)$ -clustering algorithms in the distributed setting. We remark that (Chen et al., 2016) only achieves a constant-factor-approximation, whereas we achieve a  $(1 + \varepsilon)$ -approximation.

then read the newly posted messages before the next round begins. Accordingly, messages must be self-delimiting so that all servers can correctly parse when a message has been fully posted.

Given an accuracy parameter  $\varepsilon$ , the objective is to perform a protocol  $\Pi$  such that a  $(1 + \varepsilon)$ -coreset for Euclidean  $(k, z)$ -clustering is explicitly written onto the blackboard. The communication cost of  $\Pi$  is defined as the total number of bits written to the blackboard over the course of the protocol, measured in the worst case. We note that the blackboard model can sometimes yield lower communication costs compared to the coordinator model, as messages need only be written once to be accessible to all servers simultaneously. Indeed we achieve a distributed protocol for the blackboard model with substantially less communication than our protocol for the coordinator model, tight with existing lower bounds (Chen et al., 2016; Huang et al., 2024).

**Theorem 1.2** (Communication-optimal clustering in the blackboard model, informal). *There exists a protocol on  $n$  points distributed across  $s$  sites that produces a  $(1 + \varepsilon)$ -strong coresset for  $(k, z)$ -clustering that uses  $\tilde{\mathcal{O}}\left(s \log(n\Delta) + dk \log(n\Delta) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  total bits of communication in the blackboard model.*

By comparison, the state-of-the-art protocol achieves a constant factor approximation using  $\tilde{\mathcal{O}}\left((s + dk) \log^2(n\Delta)\right)$  total communication (Chen et al., 2016). Thus compared to the work of Chen et al. (2016), not only do we achieve a  $(1 + \varepsilon)$ -coreset construction in the blackboard setting, but also we remove extraneous  $\log(n\Delta)$  factors. Conceptually, the message of Theorem 1.2 is similar to that of Theorem 1.1: some amount of exact coordinates need to be communicated (perhaps only to a small number of sites in the coordinator setting) for a constant-factor communication, but no further overhead is necessary for improving to a  $(1 + \varepsilon)$ -approximation. Finally, we remark that for both the blackboard model and the coordinator model, our coressets have size  $\tilde{\mathcal{O}}\left(\frac{k}{\varepsilon^4}\right)$  for the important cases of  $k$ -median and  $k$ -median due to sensitivity sampling (Bansal et al., 2024) and in general not only match known lower bounds (Huang et al., 2024) but also can be further optimized in certain regimes of  $n$  and  $k$  (Cohen-Addad et al., 2021; 2022a).

**Technical and algorithmic novelties.** We remark that our work introduces several algorithmic and technical innovations that may be of independent interest. For instance, we describe a number of existing approaches and why they do not work in Appendix A.

Traditional sensitivity sampling selects entire data points with probability proportional to their importance. While effective in centralized settings, this approach does not translate efficiently to distributed environments: transmitting full points or high-dimensional centers incurs prohibitive communication costs, and naive adaptive sampling requires frequent updates from all sites.

To address these challenges, we introduce several novel techniques. First, in the blackboard model, we show that adaptive sampling is robust to outdated information, leading to a “lazy” adaptive sampling protocol where sites only update the blackboard when their local weight estimates change significantly. Sampling from this outdated distribution still guarantees a constant-factor approximation, reducing both the number of transmitting sites per round and the total number of rounds. An additional  $L_1$  sampling subroutine further detects significant changes in global weight without querying all sites, improving communication efficiency.

---

162 In the coordinator model, we introduce a communication-efficient subroutine based on coordinate-  
163 wise sampling. Rather than sending full high-dimensional centers, the coordinator and a site perform  
164 a distributed binary search on the site’s sorted coordinates to find the closest match. Only a small  
165 offset is transmitted, decoupling communication from the dimension  $d$ .

166 Overall, we combine these techniques with coordinate-wise sensitivity sampling: each point is de-  
167 composed along its coordinates, and dimensions are sampled based on their significance. This al-  
168 lows the coordinator to send compact summaries to each site, with servers requesting additional  
169 information only when necessary. However, the reconstructed samples may not correspond to any  
170 actual point in the dataset, requiring careful analysis to show that the overall clustering costs are  
171 not significantly distorted. We believe these techniques could also benefit other distributed settings,  
172 such as regression and low-rank approximation. This fits into a general body of work on methods  
173 for quantizing data for better memory and communication efficiency, which is often a bottleneck for  
174 large language and other models.

## 176 2 DISTRIBUTED CLUSTERING PROTOCOLS IN THE BLACKBOARD MODEL

177  
178 Recall that in the blackboard model of communication, each of the  $s$  sites has access to private  
179 randomness and can directly broadcast to a public platform in sequential, round-based steps, with  
180 self-delimiting messages immediately visible to all. Throughout this section, we assume without  
181 loss of generality that there is a central coordinator managing the process. Our goal is to design  
182 efficient protocols for  $(k, z)$ -clustering in this setting.

### 184 2.1 CONSTANT-FACTOR BICRITERIA ALGORITHM

185  
186 In this section, we present a new algorithm that achieves an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approxima-  
187 tion; we will use it to construct a  $(1 + \varepsilon)$ -coreset in [Section 2.2](#). The resulting scheme yields  
188 a  $(1 + \varepsilon)$ -approximation with  $\tilde{\mathcal{O}}\left(s \log n + dk \log n + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  bits of communication and  
189  $\mathcal{O}(\log n \log k)$  rounds, with additional optimizations for the case  $k = \mathcal{O}(\log n)$  given in the ap-  
190 pendix. Existing bicriteria algorithms in the blackboard model suffer from communication bottle-  
191 necks, e.g., the classical Mettu–Plaxton protocol ([Mettu & Plaxton, 2004](#)) and the subsequent adap-  
192 tations ([Chen et al., 2016](#)) sample  $\mathcal{O}(k)$  points in each of  $\mathcal{O}(\log n)$  rounds, incurring  $\mathcal{O}(dk \log^2 n)$   
193 bits of communication. These costs are prohibitive for our target guarantees.

194 To overcome these barriers, we adapt the adaptive sampling framework originally developed for  
195  $k$ -median in the centralized setting ([Aggarwal et al., 2009; Balcan et al., 2013](#)). The procedure  
196 repeatedly samples points in a manner reminiscent of kmeans++ ([Arthur & Vassilvitskii, 2007;](#)  
197 [Bahmani et al., 2012](#)), but now distributed across  $s$  servers. In each iteration, we first sample a server  
198  $j$  with probability proportional to  $D_j$ , the sum of the  $z$ -th power of the distances from its points to  
199 the current sample, and then sample points within server  $j$  according to the adaptive distribution.  
200 A naïve implementation requires reporting each  $D_j$  after every iteration, leading to  $\mathcal{O}(sk \log n)$   
201 communication, which is too expensive. To implement this step efficiently, we design the subroutine  
202 **LAZYSAMPLING**, which draws from the adaptive distribution using only approximate values  $\tilde{D}_j$   
203 maintained on the blackboard. Our key innovation is to update these estimates lazily: each site  
204 reports a new value only when its true  $D_j$  changes by more than a constant factor, which we show  
205 suffices for the purposes of adaptive sampling.

206 **Lemma 2.1.** *There exists an algorithm **LAZYSAMPLING** that samples from the adaptive sampling*  
207 *distribution with probability at least 0.99 provided that  $\sum D_j \leq \sum \tilde{D}_j < \lambda \sum D_j$  for a fixed*  
208 *constant  $\lambda > 1$ . The algorithm uses  $\tilde{\mathcal{O}}(\log s + d \log n)$  bits of communication.*

209  
210 The **LAZYSAMPLING** algorithm is a communication-efficient subroutine for adaptive sampling in  
211 the blackboard model, formally described in [Algorithm 4](#) in [Appendix B](#). Each server  $j$  maintains an  
212 approximate weight  $\tilde{D}_j$  for its local dataset, satisfying  $D_j \leq \tilde{D}_j \leq \lambda D_j$ , where  $D_j$  is the sum of the  
213  $z$ -th powers of distances from its points to the current sample. The coordinator first selects a server  
214  $j$  with probability proportional to  $\tilde{D}_j / \sum_i \tilde{D}_i$ , then requests a point  $y \in X_j$ , sampled with probabili-  
215 ty  $d_y / \tilde{D}_j$ . By updating the  $\tilde{D}_j$  lazily, i.e., only when  $D_j$  changes significantly, **LAZYSAMPLING**  
ensures points are drawn close to the true adaptive distribution while drastically reducing commu-

216 nication. The algorithm either returns a sampled point or  $\perp$ , and uses only  $\tilde{\mathcal{O}}(\log s + d \log n)$  bits  
217 per round, enabling scalable execution of the bicriteria algorithm across  $s$  servers. This ensures that  
218 each site communicates only  $\mathcal{O}(\log n)$  updates, so the total communication for the constant-factor  
219 approximation is reduced to  $\tilde{\mathcal{O}}(s \log n + kd \log n)$ , since each server can only update  $\tilde{D}_j$  a total of  
220  $\mathcal{O}(\log n)$  times assuming all points lie in a grid with side length  $\text{poly}(n)$ . Because lazy updates  
221 rely on approximate values, the protocol must occasionally verify whether the aggregate estimate  
222  $\sum_j \tilde{D}_j$  is still close to the true sum  $\sum_j D_j$ . For this purpose, we use the subroutine L1SAMPLING,  
223 which tests whether the two sums differ by more than a constant factor.

224 **Lemma 2.2.** *There exists an algorithm L1SAMPLING that takes input  $\{\tilde{D}_j\}_{j \in [s]}$  so that  $\tilde{D}_j \geq D_j$   
225 for all  $j \in [s]$ . Let  $\tilde{D} = \sum_{j \in [s]} \tilde{D}_j$  and  $D = \sum_{j \in [s]} D_j$  and let  $\mu > 1$  be a parameter. If  $\mu^2 D \leq \tilde{D}$ ,  
226 then L1SAMPLING returns `True` with probability at least  $1 - \delta$ . If  $\mu D > \tilde{D}$ , it returns `False` with  
227 probability at least  $1 - \delta$ . The algorithm uses  $\tilde{\mathcal{O}}((\log s + \log \log n) \log \frac{1}{\delta})$  bits of communication.*

228 L1SAMPLING checks whether the sum of approximate site costs  $\tilde{D}$  is within a constant factor of the  
229 true total  $D$  by sampling a few sites and aggregating the rescaled sampled values. If the aggregate  
230 is sufficiently close to  $\tilde{D}$ , the algorithm returns `True`; otherwise, it returns `False`, providing a  
231 high-probability guarantee that the lazy estimates remain accurate; the algorithm is presented as  
232 [Algorithm 9](#) in [Appendix D.2](#). While this lazy strategy could require up to  $\mathcal{O}(s \log n)$  rounds, we  
233 further reduce the round complexity by delaying all updates until the global sum  $\sum_j D_j$  decreases  
234 by a constant factor. This event is naturally detected when no new point is sampled in a round,  
235 and synchronizing updates in this way brings the total number of rounds down to  $\mathcal{O}(\log n \log k)$ .  
236 Finally, to reduce communication further, we use the subroutine POWERAPPROX, which encodes  
237 each  $D_j$  to within a constant factor using only  $\mathcal{O}(\log \log n)$  bits:  
238

239 **Theorem 2.3.** *Given  $m = \text{poly}(n)$  and a constant  $\lambda > 1$ , there exists an algorithm  
240 POWERAPPROX( $m, \lambda$ ) that outputs  $\tilde{m}$  encoded in  $\mathcal{O}(\log \log n)$  bits, such that  $m \leq \tilde{m} < \lambda m$ .*

241 Given  $m$  and a base  $\lambda > 1$ , POWERAPPROX computes the smallest integer  $i$  such that  $\lambda^i$  approx-  
242 imates  $m$  from above, i.e.,  $m \leq \lambda^i < \lambda m$ , and returns  $i$ . This allows each site to communicate a  
243 concise representation of its cost using only  $\mathcal{O}(\log \log n)$  bits, which can then be decoded to obtain  
244 a constant-factor approximation of the original value; we give the full details in [Algorithm 5](#) in [Appendix B](#). Combined with LAZYSAMPLING and L1SAMPLING, this ensures accuracy while keep-  
245 ing communication near-linear. The protocol proceeds by sampling points with LAZYSAMPLING,  
246 verifying accuracy with L1SAMPLING, and refreshing estimates via POWERAPPROX only when  
247 necessary. To limit round complexity, sites synchronize updates by reporting only when the global  
248 sum  $\sum_j D_j$  decreases by a constant factor, detected whenever a round fails to sample a new point.  
249 This reduces the number of rounds from  $\mathcal{O}(s \log n)$  to  $\mathcal{O}(\log n \log k)$ . [Figure 2](#) informally summa-  
250 rizes the procedure, with the formal procedure appearing in [Appendix D.3](#). Altogether, this yields  
251 the first  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation for  $(k, z)$ -clustering in the blackboard model with  
252 near-linear communication and polylogarithmic rounds:  
253

254 **Lemma 2.4.** *There exists an algorithm that outputs a set  $S$  such that  $|S| = \mathcal{O}(k)$  and  $\text{cost}(S, X) \leq$   
255  $\mathcal{O}(1) \cdot \text{cost}(C_{\text{OPT}}, X)$  with probability at least 0.98, where  $C_{\text{OPT}}$  is the optimal  $(k, z)$ -clustering of  
256  $X$ . The algorithm uses  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  bits of communication and  $\mathcal{O}(\log n \log k)$  rounds of  
257 communication with probability at least 0.99.*

## 2.2 $(1 + \varepsilon)$ -CORESET CONSTRUCTION

258 To achieve a  $(1 + \varepsilon)$ -coreset for  $(k, z)$ -clustering on an input dataset  $X$  given an  $(\mathcal{O}(1), \mathcal{O}(1))$ -  
259 bicriteria approximation  $S$ , we use the following notion of sensitivity sampling. For each center  
260  $s_j \in S$ , let  $C_j \subset X$  be the cluster centered at  $s_j$ . For a point  $x \in C_j$ , let  $\Delta_p := \text{cost}(C_j, S)/|C_j|$   
261 denote the average cost of  $C_j$ . For  $x \in C_j$ , we define

$$262 \mu(x) := \frac{1}{4} \cdot \left( \frac{1}{k|C_j|} + \frac{\text{cost}(x, S)}{k \text{cost}(C_j, S)} + \frac{\text{cost}(x, S)}{\text{cost}(X, S)} + \frac{\Delta_x}{\text{cost}(X, S)} \right).$$

263 We define sensitivity sampling to be the process where each point  $x$  is sampled with probability  
264 proportional to a constant-factor approximation to  $\mu(x)$ . Then we have the following guarantees:

---

270           **Algorithm: Bicriteria Approximation for  $(k, z)$ -Clustering (Simplified)**

271

272   1. **Input:** Dataset  $X_i$  for each site  $i \in [s]$ .

273   2. **Output:** Set  $S$  that is an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation.

274   3. Initialize:

275       • Sample one point into  $S$  and compute approximate distances  $\{\tilde{D}_j\}$  for each site.

276       • Set counters:  $N = \mathcal{O}(k)$ ,  $M = 0$ .

277   4. **While**  $M < N$  (**sample roughly  $k$  points**):

278       • Sample new points into  $S$  using LAZYSAMPLING.

279       • Check accuracy of approximate distances  $\{\tilde{D}_j\}$  with L1SAMPLING.

280       • If distances are accurate, sample more aggressively.

281       • Otherwise, refine approximate distances  $\{\tilde{D}_j\}$  using POWERAPPROX.

282       • Update  $M$  with the number of points successfully added.

283

284   5. **Return**  $S$ .

285

287           Fig. 2: Informal version of bicriteria approximation through adaptive sampling.

288

289

290           **Theorem 2.5.** (Bansal et al., 2024) Sampling  $\tilde{\mathcal{O}}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  points from constant-factor approximations to the sensitivity sampling probability distribution and then reweighting provides a  $(1 + \varepsilon)$ -coreset for Euclidean  $(k, z)$ -clustering with probability at least 0.99.

291

292

293

294           We remark that Bansal et al. (2024) obtained optimal bounds for  $k$ -median, which immediately extend to our framework as well; we omit further discussion as it naturally generalizes. To apply the sensitivity sampling framework, we require constant-factor approximations of the cluster sizes  $|C_j|$  and costs  $\text{cost}(C_j, S)$ , given a bicriteria solution  $S$ . However, directly uploading these quantities from  $s$  servers for  $\mathcal{O}(k)$  clusters would cost  $\mathcal{O}(sk \log \log n)$  bits, which is too high. Instead, we adapt Morris counters (Morris, 1978) for the purposes of distributed approximate counting:

300           **Lemma 2.6.** Suppose each server  $i$  holds  $k$  numbers  $n_{i,j}$  and we have  $|N_j| = \sum_{i=1}^s n_{i,j} =$   
301           poly( $n$ ). There exists an algorithm DISTMORRIS that outputs  $\{\tilde{N}_j\}$  such that  $\tilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$   
302           for all  $j \in [k]$ , with probability 0.99 using  $\tilde{\mathcal{O}}(s + k \log n)$  bits of communication.

303

304           Our Morris counter protocol collectively maintains a counter  $r_j$  and increments it with probability  
305            $\frac{1}{2^r}$  for each item in a cluster  $C_j$ . This provides a constant approximation to the cluster size  $|C_j|$ .  
306           Crucially, if a site does not change the global counters  $\{r_j\}$ , then it only needs to send a single bit to  
307           signal no update. Otherwise, each of the  $\mathcal{O}(k)$  counters can only increase at most  $\mathcal{O}(\log n)$  times, so  
308           the total upload cost for these approximations is  $\mathcal{O}(k \log n)$  bits. We can perform a similar protocol  
309           to approximate the cost of each cluster  $\text{cost}(C_j, S)$ , so that overall, the total communication for these  
310           approximations is  $\tilde{\mathcal{O}}(s + k \log n)$  bits. Given these approximations, the servers can then perform  
311           sensitivity sampling locally. Finally, we require an efficient encoding of each point  $x$  sampled by  
312           sensitivity sampling. Informally, each point  $x$  is encoded as  $x' = \pi_S(x) + y'$ , where  $\pi_S(x)$  is the  
313           nearest center in  $S$  to  $x$ , and  $y'$  is the offset vector  $x - \pi_S(x)$  whose coordinates are rounded to  
314           the nearest power of  $(1 + \varepsilon')$ , where  $\varepsilon' = \text{poly}\left(\varepsilon, \frac{1}{d}, \frac{1}{\log(n\Delta)}\right)$ . The formal details are given in  
315           Appendix B. Putting everything together, Algorithm 1 achieves the guarantees in Theorem 1.2.

316

### 3 DISTRIBUTED CLUSTERING PROTOCOLS IN THE COORDINATOR MODEL

317

318

319           We now turn to the coordinator (message passing) model, where each server communicates only  
320           with the coordinator over private channels. A direct simulation of our blackboard protocol would  
321           require  $\mathcal{O}(dsk \log n)$  bits, since  $\mathcal{O}(k)$  rounds of adaptive sampling would need to be executed  
322           across  $s$  servers. To avoid this prohibitive cost, we design a protocol that simulates adaptive sampling  
323           without sending points explicitly to all sites. We first apply a Johnson–Lindenstrauss transformation  
324           to reduce the dimension to  $d' = \mathcal{O}(\log(sk))$ , preserving pairwise distances up to  $(1 \pm \varepsilon)$ . We

---

324   **Algorithm 1**  $(1 + \varepsilon)$ -coreset for the blackboard model

325

326   **Input:** A bicriteria set of centers  $S$  with constant-factor approximation and  $|S| = \mathcal{O}(k)$

327   **Output:** A  $(1 + \varepsilon)$ -coreset  $A$

328   1: Use DISTMORRIS to get  $\mathcal{O}(1)$ -approximation for  $|C_j|$  and  $\text{cost}(C_j, S)$  for all  $j \in [k]$  on the blackboard

329   2:  $m \leftarrow \tilde{\mathcal{O}}\left(\frac{k}{\varepsilon^2} \min\{\varepsilon^{-2}, \varepsilon^{-z}\}\right)$  ▷ Set coreset size

330   3: **for**  $i \leftarrow 1$  to  $s$  ▷ Send local approximations to sensitivities **do**

331    4:    $A_i \leftarrow \emptyset$

332    5:   Compute  $\tilde{\mu}(x)$  as an  $\mathcal{O}(1)$ -approximation of  $\mu(x)$  locally for all  $x \in X_i$

333    6:   Upload  $\tilde{\mu}(X_i) = \sum_{x \in X_i} \tilde{\mu}(x)$  to blackboard

334    7: **end for**

335    8: Sample site  $i$  with probability  $\frac{\tilde{\mu}(X_i)}{\sum_{i=1}^s \tilde{\mu}(X_i)}$  independently for  $m$  times. ▷ Sensitivity sampling

336    9: Let  $m_i$  be the number of times site  $i$  is sampled and write  $m_i$  on blackboard

337   10: **for**  $i \leftarrow 1$  to  $s$  ▷ Iterate through sites to produce samples **do**

338    11:    $A_i \leftarrow \emptyset$

339    12:   **for**  $j \in [m_i]$  ▷ Sample  $m_i$  points from site  $i$  **do**

340    13:     Sample  $x$  with probability  $p_x = \frac{\tilde{\mu}(x)}{\tilde{\mu}(X_i)}$

341    14:     **if**  $x$  is sampled ▷ Efficiently encode each sample **then**

342    15:       Let  $x'$  be  $x$  efficiently encoded by  $S$  and accuracy  $\varepsilon' = \text{poly}(\varepsilon)$

343    16:        $A \leftarrow A_i \cup \{(x', \frac{1}{m\hat{\mu}(x)})\}$ , where  $\hat{\mu}(x)$  is a  $(1 + \frac{\varepsilon}{2})$ -approximation of  $\tilde{\mu}(x)$

344    17:     **end if**

345    18:   **end for**

346    19:   Upload  $A_i$  to the blackboard

347    20: **end for**

348    21:  $A \leftarrow \cup_{i=1}^s A_i$

349   **return**  $A$

---

350

351   can then perform adaptive sampling in the projected space, so that when a point  $s$  is selected, only

352   its projection  $\pi(s)$  is communicated. To efficiently approximate such locations, we introduce the

353   subroutine EFFICIENTCOMMUNICATION, which transmits an approximate version  $\tilde{y}$  of any point

354    $y$  using only  $d \log k \text{polylog}(\log n, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits, rather than the  $\mathcal{O}(d \log n)$  bits required for exact

355   communication:

356   **Lemma 3.1.** *Given a point  $y$  and a dataset  $X$ , there exists an algorithm*

357   **EFFICIENTCOMMUNICATION** *that uses  $d \log k \text{polylog}(\log n, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits of communication*

358   *and sends  $\tilde{y}$  such that  $\|y - \tilde{y}\|_2 \leq \min_{x \in X} \varepsilon \|x - y\|_2$  with probability at least  $1 - \delta$ .*

359

360   Intuitively, EFFICIENTCOMMUNICATION allows a site to locate an approximate version of the co-

361   ordinator's point  $y$  using very little communication. For each coordinate  $i$ , the site first identifies

362   the closest local value  $x_{i_s}^{(i)}$  to  $y^{(i)}$  via a binary search using the HIGHPROBGREATERTHAN proto-

363   col. If  $y^{(i)}$  does not exactly match, an exponential search determines a small offset  $\Delta y^{(i)}$  so that

364    $x_{i_s}^{(i)} + \Delta y^{(i)}$  approximates  $y^{(i)}$  within a factor of  $(1 + \varepsilon)$ . By doing this coordinate-wise, the site can

365   efficiently reconstruct a point  $\tilde{y}$  that is close to  $y$ , guaranteeing  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  for any local

366   point  $x \in X$ , while sending only a small number of bits. This approach combines the intuition of

367   searching for the “nearest neighbor” along each coordinate with controlled, approximate refinement,

368   as illustrated in Figure 3.

369   Given a bicriteria approximation  $S$  obtained from the above efficient implementation of adaptive

370   sampling, we next perform sensitivity sampling to construct a  $(1 + \varepsilon)$ -coreset. Unfortunately, each

371   server can now assign a point  $x$  to another center  $s'$  instead of the closest center  $s$  if  $\text{cost}(x, s')$  is very

372   close to  $\text{cost}(x, s)$ . Consequently, the sizes  $|C_j|$  and costs  $\text{cost}(C_j, S)$  for the purposes of sensitivity

373   sampling may be incorrectly computed by the servers. However, this does not compromise the

374   correctness: the sensitivity analysis of Bansal et al. (2024) only requires that each point be assigned

375   to a center whose clustering cost is within a constant factor of the optimal assignment. Thus, our

376   procedure still achieves a  $(1 + \frac{\varepsilon}{4})$ -coreset by sensitivity sampling. Once points are sampled, each

377   site encodes the coordinates of its sampled points using the same efficient encoding scheme as in the

blackboard model, c.f., Lemma B.13, ensuring that only a compact representation is sent back to the

---

378  
379  
380   **Algorithm: EFFICIENTCOMMUNICATION**( $X, y, \varepsilon, \delta$ ) **(informal)**  
381   1. **Input:**  
382     • A set of points  $X = \{x_1, \dots, x_l\}$  owned by one site.  
383     • A point  $y$  from the coordinator.  
384     • Accuracy parameter  $\varepsilon \in (0, 1)$  and failure probability  $\delta$ .  
385   2. **Goal:** Send an approximate location  $\tilde{y}$  to the site such that  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  for any  $x \in X$  with probability  $\geq 1 - \delta$ .  
386   3. For each coordinate  $i = 1$  to  $d$ :  
387     • Sort points in  $X$  by their  $i$ -th coordinate.  
388     • Use a local binary search via HIGHPROBGREATERTHAN to find the closest point  $x_{i_s}^{(i)}$  to  $y^{(i)}$ .  
389     • If  $y^{(i)}$  equals  $x_{i_s}^{(i)}$ , set  $\Delta y^{(i)} = 0$ .  
390     • Otherwise:  
391       ◦ Determine the direction  $\gamma = \text{sign}(y^{(i)} - x_{i_s}^{(i)})$ .  
392       ◦ Use exponential search with HIGHPROBGREATERTHAN to find a value  $\Delta y^{(i)}$  so that  
393       ◦  $x_{i_s}^{(i)} + \Delta y^{(i)}$  approximates  $y^{(i)}$  within factor  $(1 + \varepsilon)$ .  
394     • Set  $\tilde{y}^{(i)} = x_{i_s}^{(i)} + \Delta y^{(i)}$ .  
395   4. **Return**  $\tilde{y} = (\tilde{y}^{(1)}, \dots, \tilde{y}^{(d)})$ .

401  
402   Fig. 3: Informal version of efficient communication in the message-passing algorithm. For full  
403   algorithm, see [Algorithm 15](#).  
404  
405

406   coordinator. This combination of approximate center assignments and efficient encoding preserves  
407   both accuracy and communication efficiency. We give the algorithm in full in [Figure 4](#), which  
408   achieves the guarantees of [Theorem 1.1](#), deferring full details to [Appendix E](#).  
409  
410

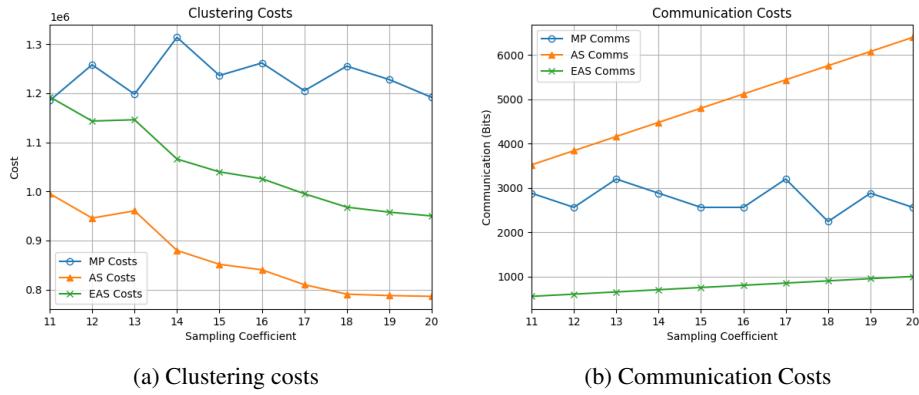
411   **Algorithm:  $(1 + \varepsilon)$ -coreset for the coordinator model (informal)**  
412   (1) Each site computes a local  $(1 + \varepsilon/2)$ -coreset  $P_i$ .  
413   (2) Coordinator broadcasts a JL transform  $\pi$  to all sites.  
414   (3) Initialize solution set  $S = \{s_0\}$  with a random point.  
415   (4) For  $i = 1$  to  $i = \mathcal{O}(k)$  iterations (for bicriteria solution):  
416     • Using EFFICIENTCOMMUNICATION, send approximation  $\tilde{s}_{i-1}^{(j)}$  of center  $s_i$  to site  $j$ .  
417     • Sites compute approximate costs  $\tilde{D}_j$  and send to coordinator.  
418     • Coordinator selects next center  $s_i$  into  $S$  using LAZYSAMPLING.  
419   (5) Sites compute cluster sizes and costs, send constant approximations to coordinator.  
420   (6) Coordinator computes total approximations and broadcasts to sites.  
421   (7) Sites compute approximate sensitivities  $\tilde{\mu}(x)$ , send total to coordinator.  
422   (8) Coordinator samples  $m = \tilde{\mathcal{O}}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^2 + \varepsilon)}\right)$  points across sites based on sensitivities;  
423     sites sample points and encode with EFFICIENTCOMMUNICATION.  
424   (9) Merge sampled points into final coresset  $A'$  and return.  
425  
426  
427  
428  
429

430   Fig. 4: Informal version of message-passing algorithm. For full algorithm, see [Algorithm 16](#).  
431

---

## 432 4 EMPIRICAL EVALUATIONS

434 In this section, we present a number of simple experimental results on both synthetic and real-  
435 world datasets that complement our theoretical guarantees. We consider  $k$ -means clustering in the  
436 blackboard setting, using the previous algorithm of Chen et al. (2016) based on Mettu-Plaxton, de-  
437 noted MP, as a baseline. We also implement two versions of our distributed protocol, with varying  
438 complexity. We first implement our constant-factor approximation algorithm based on adaptive sam-  
439 pling, denoted AS. Additionally, we implement our constant-factor approximation algorithm based  
440 on our compact encoding after adaptive sampling, denoted EAS. All experiments were conducted  
441 on a Dell OptiPlex 7010 Tower desktop equipped with an Intel Core i7-3770 3.40 GHz quad-core  
442 processor and 16 GB of RAM. We provide all code in the supplementary material.



455 Fig. 5: Experiments for clustering costs and communication costs on DIGITS dataset

### 460 4.1 REAL-WORLD DATASET

461 To evaluate our algorithms, we conducted our  $k$ -means clustering algorithms on the DIGITS  
462 dataset (Alpaydin & Kaynak, 1998), which consists of 1,797 images of handwritten digits (0-9)  
463 and thus naturally associates with  $k = 10$ . Each image has dimension  $8 \times 8$ , represented by 64 fea-  
464 tures, corresponding to the pixel intensities. This dataset is available both through scikit-learn  
465 and the UCI repository, and is a popular choice for clustering tasks due to its moderate size and well-  
466 defined classes, allowing for a clear evaluation of the clustering performance.

467 For a parameter  $c$ , our baseline MP uniformly selects  $k$  initial points and then iteratively discards  
468 candidates over  $c \log_2 n$  rounds. Our AS algorithm iteratively samples a single center in each of  $ck$   
469 rounds, with probability proportional to its distance from the previously sampled centers. Finally,  
470 EAS quantizes the coordinates of each chosen center of AS to the nearest power of two, mimicking  
471 low-precision communication. We compare the clustering costs of the algorithms in Figure 5a and  
472 their total communication costs in Figure 5b, both across  $c \in \{11, 12, \dots, 20\}$ .

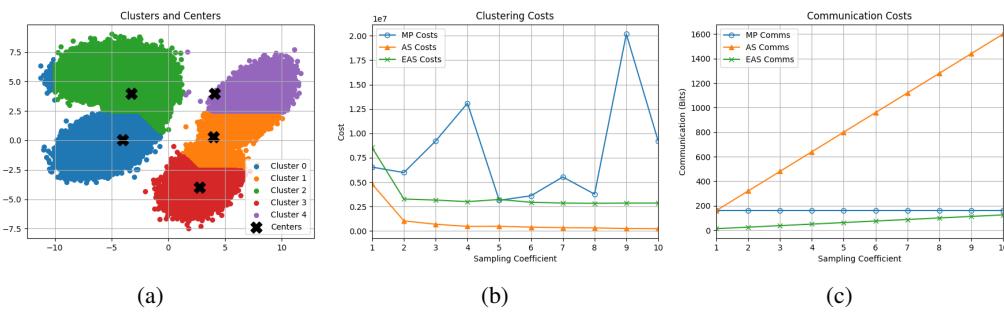
473 Our results indicate that although adaptive sampling (AS) always outperforms Mettu-Plaxton (MP),  
474 when the number of samples is small, the gap is relatively small, so that the rounding error incurred  
475 by our efficient encoding in EAS has similar clustering costs for  $c = 11$  in Figure 5a. Surprisingly,  
476 the communication cost of MP did not seem to increase with  $c$ , indicating that all possible points  
477 have already been removed and no further samples are possible. Nevertheless, for all  $c > 11$ , our  
478 algorithm in EAS clearly outperforms MP and therefore the previous work of Chen et al. (2016) for  
479 both clustering cost and communication cost, c.f., Figure 5b. Our algorithm EAS also exhibits clear  
480 tradeoffs in the clustering cost and the communication cost compared to our algorithm AS, as the  
481 former is simply a rounding of the latter.

### 483 4.2 SYNTHETIC DATASET

485 To facilitate visualization, we generated synthetic datasets consisting of two-dimensional Gaussian  
486 mixtures, where the low dimensionality (2D) was chosen to enable visualization of the resulting

486 clusters. Specifically, we created  $k = 5$  Gaussian clusters, each containing  $n = 100 \times 2^{10}$  points,  
 487 for a total of 512,000 data points. Each cluster was sampled from a distinct Gaussian distribution  
 488 with a randomly selected mean in the range  $[-10, 10]^2$  and a randomly generated positive-definite  
 489 covariance matrix to ensure diverse cluster shapes.

490 We implemented the baseline **MP** by sampling  $k$  points uniformly and pruning candidates based on  
 491 a distance threshold that doubles each round, across  $c \log_2 n$  rounds, where  $c$  is a hyperparameter.  
 492 We implemented our algorithm **AS** by iteratively sampling one center per round across  $ck$  rounds,  
 493 selecting points with probability proportional to their distance from existing centers. **EAS** then  
 494 modifies **AS** by rounding each selected center's coordinates to the nearest power of  $q = 2^{0.25}$ ,  
 495 simulating low-precision communication.



506 Fig. 6: Experiments for clustering costs and communication costs on synthetic dataset

507 We varied the sampling coefficient  $c \in \{11, \dots, 20\}$  and measured the  $k$ -means clustering cost (i.e.,  
 508 the total squared distance from each point to its assigned center). The results, shown in Figure 6,  
 509 highlight the performance trade-offs between sampling strategies as  $c$  increases. Communication  
 510 costs were also computed (though not plotted), assuming 32 bits per coordinate for **MP** and **AS**, and  
 511 5 bits per coordinate for **EAS** due to quantization.

512 Our results for synthetic data echo the trends for the DIGITS dataset. Namely, for all  $c > 2$ , our  
 513 algorithm in **EAS** clearly outperforms **MP** for both clustering cost and communication cost, c.f.,  
 514 Figure 6c, while also demonstrating clear tradeoffs in the clustering cost and the communication  
 515 cost compared to our algorithm **AS**. We plot the resulting clustering by **EAS** in Figure 6a.

## 516 REFERENCES

517 Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for  $k$ -means clustering.  
 518 In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pp.  
 519 15–28. Springer, 2009. 4, 13, 16, 19, 20, 21

520 E. Alpaydin and C. Kaynak. Optical Recognition of Handwritten Digits. UCI Machine Learning  
 521 Repository, 1998. DOI: <https://doi.org/10.24432/C50P49>. 9, 40

522 David Arthur and Sergei Vassilvitskii.  $k$ -means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pp.  
 523 1027–1035, 2007. 4, 13

524 Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable  $k$ -means++. *Proc. VLDB Endow.*, 5(7):622–633, 2012. 4, 13

525 Maria-Florina Balcan, Steven Ehrlich, and Yingyu Liang. Distributed  $k$ -means and  $k$ -median clustering on general communication topologies. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, pp. 1995–2003, 2013.  
 526 2, 3, 4, 13, 14, 39, 40

527 Nikhil Bansal, Vincent Cohen-Addad, Milind Prabhu, David Saulpic, and Chris Schwiegelshohn.  
 528 Sensitivity sampling for  $k$ -means: Worst case and stability optimal coresset bounds. *CoRR*,  
 529 abs/2405.01339, 2024. 2, 3, 6, 7, 12, 14, 15, 16, 31

---

540 Jiecao Chen, He Sun, David P. Woodruff, and Qin Zhang. Communication-optimal distributed  
541 clustering. In *Advances in Neural Information Processing Systems 29: Annual Conference on*  
542 *Neural Information Processing Systems*, pp. 3720–3728, 2016. 2, 3, 4, 9, 12  
543

544 Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coresset framework for  
545 clustering. In *STOC: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 169–  
546 182, 2021. 2, 3, 12

547 Vincent Cohen-Addad, Kasper Green Larsen, David Saulpic, and Chris Schwiegelshohn. Towards  
548 optimal lower bounds for k-median and k-means coresets. In *STOC '22: 54th Annual ACM*  
549 *SIGACT Symposium on Theory of Computing*, pp. 1038–1051, 2022a. 2, 3, 12, 15  
550

551 Vincent Cohen-Addad, Kasper Green Larsen, David Saulpic, Chris Schwiegelshohn, and  
552 Omar Ali Sheikh-Omar. Improved coresets for euclidean k-means. In *NeurIPS*,  
553 2022b. URL [http://papers.nips.cc/paper\\_files/paper/2022/hash/120c9ab5c58ba0fa9dd3a22ace1de245-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/120c9ab5c58ba0fa9dd3a22ace1de245-Abstract-Conference.html). 2, 12, 15  
554

555 Vincent Cohen-Addad, David P. Woodruff, and Samson Zhou. Streaming euclidean  $k$ -median and  
556  $k$ -means with  $o(\log n)$  space. In *64th IEEE Annual Symposium on Foundations of Computer*  
557 *Science, FOCS*, pp. 883–908, 2023. 12

558 Vincent Cohen-Addad, Liudeng Wang, David P. Woodruff, and Samson Zhou. Fast, space-optimal  
559 streaming algorithms for clustering and subspace embeddings. *CoRR*, abs/2504.16229, 2025. 18,  
560 19

561 Lingxiao Huang and Nisheeth K. Vishnoi. Coresets for clustering in euclidean spaces: importance  
562 sampling is nearly optimal. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on*  
563 *Theory of Computing, STOC*, pp. 1416–1429, 2020. 12

564 Lingxiao Huang, Jian Li, and Xuan Wu. On optimal coreset construction for euclidean  $(k, z)$ -  
565 clustering. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC*,  
566 pp. 1594–1604, 2024. 2, 3, 15

567 Zachary Izzo, Sandeep Silwal, and Samson Zhou. Dimensionality reduction for wasserstein  
568 barycenter. In *Advances in Neural Information Processing Systems 34: Annual Conference on*  
569 *Neural Information Processing Systems, NeurIPS*, 2021. 12, 15

570 William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space.  
571 conference in modern analysis and probability (new haven, conn., 1982), 189–206. In *Contemp.*  
572 *Math*, volume 26, 1984. 15

573 Daniel M Kane and Jelani Nelson. A derandomized sparse johnson-lindenstrauss transform. *arXiv*  
574 *preprint arXiv:1006.3585*, 2010. 16

575 Yurii Lyubarskii and Roman Vershynin. Uncertainty principles and vector quantization. *IEEE*  
576 *Trans. Inf. Theory*, 56(7):3491–3501, 2010. doi: 10.1109/TIT.2010.2048458. URL <https://doi.org/10.1109/TIT.2010.2048458>. 12

577 J MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math.*  
578 *Statist. Probability*, pp. 281–297, 1967. 1

579 Konstantin Makarychev, Yury Makarychev, and Ilya P. Razenshteyn. Performance of johnson-  
580 lindenstrauss transform for  $k$ -means and  $k$ -medians clustering. In *Proceedings of the 51st Annual*  
581 *ACM SIGACT Symposium on Theory of Computing, STOC*, pp. 1027–1038, 2019. 12, 15

582 Ramgopal R. Mettu and C. Greg Plaxton. Optimal time bounds for approximate clustering. *Mach.*  
583 *Learn.*, 56(1-3):35–60, 2004. 4, 12, 13

584 Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*,  
585 21(10):840–842, 1978. 6, 30

586 Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is*  
587 *Eighty*, 1(301-315):6, 1993. 16, 33

---

594 Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation:  
595 Goodbye dimension. In *59th IEEE Annual Symposium on Foundations of Computer Science*,  
596 *FOCS*, pp. 802–813, 2018. [12](#), [15](#)

597 Hugo Steinhaus et al. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci.*, 1(804):  
598 801, 1956. [1](#)

599 Xiaoyi Zhu, Yuxiang Tian, Lingxiao Huang, and Zengfeng Huang. Space complexity of euclidean  
600 clustering. *IEEE Trans. Inf. Theory*, 71(6):4515–4536, 2025. [12](#)

601

602

603

## A TECHNICAL OVERVIEW

604 In this section, we provide a technical overview for our distributed protocols, for both the blackboard  
605 model and the coordinator/message-passing model.

606 **Previous approaches and why they do not work.** A natural question is whether simple  
607 approaches could achieve similar communication bounds. One might hope, for instance, that com-  
608 bining the space upper bounds of [Zhu et al. \(2025\)](#) with existing distributed clustering algorithms  
609 would suffice. However, it is not immediately clear how to translate space bounds into better dis-  
610 tributed clustering bounds. For instance, as an application of their space bounds, [Zhu et al. \(2025\)](#)  
611 achieved a distributed protocol with total communication  $\tilde{\mathcal{O}}\left(\frac{sk}{\varepsilon^4}\right)$ , for appropriate ranges of  $k$   
612 and  $\varepsilon$ , which is actually worse than the third row of [Figure 1](#) and thus worse than our bounds of  
613  $\tilde{\mathcal{O}}\left(sk + \frac{dk}{\varepsilon^4} + dk \log n\right)$  for the same regime.

614 Another approach might be that each server computes a local coreset, e.g., [Huang & Vishnoi \(2020\)](#);  
615 [Cohen-Addad et al. \(2021; 2022a;b\)](#); [Bansal et al. \(2024\)](#), possibly with the incorporation of di-  
616 mensionality reduction, e.g., [Sohler & Woodruff \(2018\)](#); [Makarychev et al. \(2019\)](#); [Izzo et al. \(2021\)](#),  
617 and then either broadcasting these coresets in the blackboard model or sending these coresets to  
618 the coordinator in the message-passing model. However, this uses communication  $\mathcal{O}\left(\frac{sk}{\varepsilon^4} \log n\right)$   
619 bits, compared to our bounds of  $\tilde{\mathcal{O}}\left(sk + \frac{dk}{\varepsilon^4} + dk \log n\right)$ . Similarly, any approach based on locality  
620 sensitivity hashing, possibly with the aid of dimensionality reduction or other compressions, e.g.,  
621 Kashin representations ([Lyubarskii & Vershynin, 2010](#)), would require at least each server comput-  
622 ing a local coreset and thus succumbing to the same pitfalls. These limitations highlight that over-  
623 coming the inherent overheads in naïve strategies is far from straightforward, and motivate the need  
624 for new techniques. Our framework is designed precisely to address these challenges by simulating  
625 centralized algorithms through communication-optimal primitives, thereby breaking the bottlenecks  
626 that defeat such strawman approaches.

627  **$(1 + \varepsilon)$ -coreset for the blackboard model.** Our starting point is the distributed protocol of [Chen  
628 et al. \(2016\)](#), which first adapts the central RAM algorithm of [Mettu & Plaxton \(2004\)](#) to publish  
629 a weighted set  $S$  of  $\mathcal{O}(k \log n)$  centers that is a constant-factor bicriteria approximation for  $(k, z)$ -  
630 clustering. It can be shown that a constant-factor approximation to  $(k, z)$ -clustering on  $S$  achieves  
631 a constant-factor approximation  $C$  to the optimal clustering on the input set  $X$ . Since  $S$  is already  
632 available on the blackboard, this step is immediate. The total communication cost of their algorithm  
633 is  $\mathcal{O}((s + kd) \log^2 n)$ .

634 We observe that the sites can easily calculate the number of points in  $X$  served by each center  
635  $c \in C$ . We recall that using this information as well as  $\text{cost}(X, C)$  can be used to find constant-  
636 factor approximations of the sensitivity  $s(x)$  of each point  $x$ , c.f. ([Cohen-Addad et al., 2023](#)).  
637 Thus, the sites can subsequently sample  $\tilde{\mathcal{O}}\left(\min\left(\frac{k^2}{\varepsilon^2}, \frac{k}{\min(\varepsilon^4, \varepsilon^2 + z)}\right)\right)$  points of  $X$  using sensitivity  
638 sampling to achieve a  $(1 + \varepsilon)$ -coreset. We can then use an efficient encoding scheme to express each  
639 sampled point in  $\mathcal{O}(\log k + d \log(\frac{1}{\varepsilon}, d, \log(n\Delta)))$  bits. This achieves the desired dependency for  
640 the points acquired through sensitivity sampling, but the communication from the constant-factor  
641 approximation is sub-optimal. Thus, it remains to find a more communication-efficient protocol for  
642 the constant-factor approximation.

643 **Constant-factor approximation in the blackboard model.** Although there are many options  
644 for the constant-factor approximation in the blackboard model, they all seem to have their various

---

648 shortcomings. For example, the aforementioned Mettu-Plaxton protocol (Mettu & Plaxton, 2004) re-  
649 quires  $\mathcal{O}(\log n)$  rounds of sampling, which results in  $\mathcal{O}(k \log n)$  points. A natural approach would  
650 be to form coresets locally at each site so that the total number of weighted points is  $\mathcal{O}(sk)$  rather  
651 than  $n$ . Unfortunately, when generalized to weighted points, Mettu-Plaxton requires  $\mathcal{O}(\log W)$   
652 rounds, where  $W$  is total weight of the points, which is  $\text{poly}(n)$  for our purposes.

653 Another well-known prototype is the adaptive sampling framework (Aggarwal et al., 2009; Balcan  
654 et al., 2013), which can be used to implement the well-known kmeans++ algorithm (Arthur & Vas-  
655 silvitskii, 2007) in the distributed setting (Bahmani et al., 2012). This approach iteratively samples  
656 a fixed number of points with probability proportional to the  $z$ -th power of the distances from the  
657 previously sampled points; adaptive sampling samples  $\mathcal{O}(k)$  points for  $\mathcal{O}(1)$ -approximation while  
658 kmeans++ samples  $k$  points for  $\mathcal{O}(\log k)$ -approximation. However, since all sites  $i \in [s]$  must  
659 report the sum  $D_i$  of the  $z$ -th power of the distances of their points to the sampled points after each  
660 iteration, these approaches naively require  $\mathcal{O}(sk \log n)$  bits of communication, which is prohibitive  
661 for our goal.

662 Instead, we perform lazy updating of the sum  $D_i$  of the  $z$ -th power of the distances of their points  
663 to the sampled points. The blackboard only holds estimates  $\widehat{D}_i$  for  $D_i$ , based on the last time the  
664 site  $i$  reported its value. The sites then attempt adaptive sampling, where the site  $i$  is sampled  
665 with a probability proportional to  $\widehat{D}_i$ . When sampled, site  $i$  uses  $\widehat{D}_i$  to attempt to sample a point  
666 locally, but this can fail because  $\widehat{D}_i \geq D_i$ , which means that a randomly chosen integer in  $\widehat{D}_i$   
667 may not correspond to an integer in  $D_i$ . Fortunately, the failure probability would be constant if  
668  $\widehat{D}_i$  is a constant approximation of  $D_i$ , so the total number of rounds of sampling is still  $\mathcal{O}(k)$  by  
669 the Markov inequality. If the site  $i$  fails to sample a point, we expect  $D_i$  to be a constant fraction  
670 smaller than  $\widehat{D}_i$ , prompting site  $i$  to update its value  $D_i$  on the blackboard. This can happen at most  
671  $\mathcal{O}(s \log n)$  times. If each site  $i$  rounds  $D_i$  to a power of 2, then  $D_i$  can be approximated within a  
672 factor of 2 by transmitting only the exponent of the rounding, using  $\mathcal{O}(\log \log n)$  bits. Thus, the  
673 total communication for the constant-factor approximation is  $\tilde{\mathcal{O}}(s \log n + kd \log n)$ .

674 The primary downside is that the total number of communication rounds could reach  $\mathcal{O}(s \log n)$ ,  
675 since each of the  $s$  sites may update its  $D_i$  value up to  $\mathcal{O}(\log n)$  times.

676 **Communication round reduction in the blackboard model.** A natural approach to round re-  
677 duction would be to have all  $s$  sites report their updated  $D_i$  values when we fail to sample a point.  
678 However, because we sample  $\mathcal{O}(k)$  points, this results in communication complexity  $\mathcal{O}(sk)$ .

679 Instead, we check whether the global weight  $\sum_{i \in [s]} D_i$  has decreased by  $\frac{1}{64}$  compared to the amount  
680  $\sum_{i \in [s]} \widehat{D}_i$  on the blackboard. To do this, we first perform the  $L_1$  sampling in each round. That is,  
681 the blackboard picks the site  $i$  with probability  $p_i = \frac{\widehat{D}_i}{\sum_{i \in [s]} \widehat{D}_i}$  and computes  $\frac{D_i}{p_i}$ . Observe that in  
682 expectation this quantity is  $\sum_{i \in [s]} D_i$  and its variance is at most  $(\sum_{i \in [s]} D_i)^2$ , up to a constant fac-  
683 tor. Thus, repeating  $\mathcal{O}(1)$  times and taking the average, we get a  $\sum_{i \in [s]} D_i$  additive approximation  
684 to  $\sum_{i \in [s]} D_i$ , which is enough to identify whether the sum has decreased by a factor of  $\frac{1}{64}$  from  
685  $\sum_{i \in [s]} \widehat{D}_i$ . We can then take  $\mathcal{O}(\log \log n)$  instances to union bound over  $\mathcal{O}(\log n \log k)$  iterations.

686 Now the other observation is that as long as our sampling probabilities have not decreased by  $\frac{1}{64}$ ,  
687 then we can simultaneously take many samples at the same time. Thus, we start at  $i = 0$  and collect  
688  $2^i$  samples from the  $s$  sites using the adaptive sampling distribution. We then check if the total  
689 weight  $\sum_{i \in [s]} D_i$  is less than  $\frac{1}{64} \sum_{i \in [s]} \widehat{D}_i$ , and if not, we increment  $i$  and sample  $2^i$  samples, and  
690 repeat until either  $2^i > k$  or the total weight  $\sum_{i \in [s]} D_i$  is less than  $\frac{1}{64} \sum_{i \in [s]} \widehat{D}_i$ .

691 Note that this takes  $\mathcal{O}(\log n \log k)$  rounds of sampling, which is enough for the failure probability  
692 of the  $L_1$  sampling procedure. Moreover, since we can use  $\mathcal{O}(\log \log n)$  bits of communication to  
693 approximate  $L_1$  sampling by rounding the values of  $D_i$  to a power of 2 and returning the exponent.  
694 Since each time the total weight  $\sum_{i \in [s]} D_i$  is less than  $\frac{1}{64} \sum_{i \in [s]} \widehat{D}_i$ , we require all  $s$  sites to update  
695 their weights, then the communication is  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  across the  $\mathcal{O}(\log n \log k)$  rounds of  
696 communication.

---

702 **Applying sensitivity sampling in the blackboard model.** To apply the sensitivity sampling  
 703 framework of [Bansal et al. \(2024\)](#), a constant approximation of the number of each cluster  $|C_j|$   
 704 and the cost of each cluster  $\text{cost}(C_j, S)$  is necessary, where  $S$  is a  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approxi-  
 705 mation of the optimal solution. Each site needs  $\mathcal{O}(\log \log n)$  bits to upload such a constant approxi-  
 706 mation for a cluster. Since there are  $s$  sites and  $\mathcal{O}(k)$  clusters, it would lead to  $\mathcal{O}(sk \log \log n)$  bits  
 707 of communication, which is prohibitive for our goal. Instead, we adapt Morris counters, which are  
 708 used for approximate counting in the streaming model, to the distributed setting. We use a counter  $r$   
 709 to store the logarithm of the number/cost of the cluster, and increase the counter by 1 with probabili-  
 710 ty  $\frac{1}{2^r}$  for every time we count the number/cost of the cluster. Each site applies the Morris counters  
 711 sequentially and only uploads the increment on the blackboard. Similar to Morris counters, such  
 712 a subroutine can return a constant approximation of the number/cost of the cluster. If all counters  
 713 remain the same after counted by a site, that site needs to use  $\mathcal{O}(1)$  bits to tell the next site that  
 714 nothing needs to be updated, which would use at most  $\mathcal{O}(1)$ . Otherwise,  $\mathcal{O}(\log \log n)$  bits are  
 715 needed to update a changed counter. Since each counter can increase at most  $\mathcal{O}(\log n)$  times and  
 716 there are  $\mathcal{O}(k)$  numbers of number/cost of the cluster to be counted, it would use  $\mathcal{O}(k \log n)$  bits  
 717 to upload all the updates. Therefore, at most  $\tilde{\mathcal{O}}(s + k \log n)$  bits are needed to upload a constant  
 718 approximation of the number of each cluster  $|C_j|$  and the cost of each cluster  $\text{cost}(C_j, S)$ .

719 Each site can apply the sensitivity sampling locally after knowing a constant approximation of the  
 720 number of each cluster  $|C_j|$  and the cost of each cluster  $\text{cost}(C_j, S)$ . Combined with the efficient  
 721 encoding for the  $(1 + \varepsilon)$ -coreset, we can upload the coresets using  $\mathcal{O}\left(\frac{(d \log \log n + \log k)k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  bits, which  
 722 achieves our desired bounds for the blackboard model.

724 **Coordinator/message-passing model.** For the message-passing/coordinator model, our starting  
 725 point is once again adaptive sampling, which can be performed over  $\mathcal{O}(k)$  rounds to achieve a  
 726 constant-factor approximation  $C$  using  $\mathcal{O}(skd \log n)$  total bits of communication [\(Balcan et al.,](#)  
 727 [2013\)](#). We can again use sensitivity sampling on  $C$  along with our efficient encoding scheme to  
 728 achieve the desired communication for the subsequent  $(1 + \varepsilon)$ -coreset. Thus, the main question  
 729 again is how to improve the constant-factor protocol.

730 To that end, we first introduce a subroutine `EFFICIENTCOMMUNICATION` that can send the location  
 731 of the point using low communication cost. The intuition behind `EFFICIENTCOMMUNICATION` is  
 732 similar to the efficient encoding, which uses the points a site owns to encode the point we want to  
 733 send. The only difference is that we use different points to efficiently encode the coordinates for  
 734 each dimension, rather than just using a single point to encode the point we want to send.

735 Recall that there exists `GREATERTHAN` protocol, which given two  $\log n$  bit integers, figures out  
 736 which one is greater with constant probability, using  $\tilde{\mathcal{O}}(\log \log n)$  bits of communication. More-  
 737 over, this procedure can be repeated  $\mathcal{O}(\log(sk))$  times to achieve a failure probability  $\frac{1}{\text{poly}(sk)}$  to  
 738 union bound over  $\text{poly}(sk)$  possible steps. With the help of `GREATERTHAN`, we can compare the  
 739 coordinates of the points in the  $i$ -th dimension  $x_j^{(i)}$  a site owns with the coordinate of the point to  
 740 receive  $y^{(i)}$ . By a binary search, we can find the closest coordinate  $x_j^{(i)}$  to  $y^{(i)}$ , which requires  
 741  $\mathcal{O}(\log n)$  search times, since the site owns at most  $n$  points. By another binary search, we can  
 742 further find  $\Delta y^{(i)}$  that is a  $(1 + \varepsilon)$ -approximation of  $|x_j^{(i)} - y^{(i)}|$ . Since  $|x_j^{(i)} - y^{(i)}| = \text{poly}(n)$ ,  
 743 we need  $\mathcal{O}(\log \log n)$  searches to find  $\Delta y^{(i)}$ . Since the coordinator needs to send the approximate  
 744 coordinates of the points sampled by adaptive sampling to each site, the total communication cost  
 745 for adaptive sampling is  $\tilde{\mathcal{O}}(dsk \log n)$  bits, which is still prohibitive for our goal.

746 To address this, we first generate a  $(1 + \mathcal{O}(\varepsilon))$ -coreset for each site, so that the size of the dataset  
 747 of each site is  $\mathcal{O}(k)$ , which leads  $\mathcal{O}(\log n)$  searches to find the closest coordinate  $x_j^{(i)}$  to  $y^{(i)}$ . Fur-  
 748 thermore, we observe that we only need the approximate cost for each point for adaptive sampling.  
 749 Hence, we can apply Johnson-Lindenstrauss to project the dataset down to  $\mathcal{O}(\log(sk))$  dimensions.  
 750 Therefore, the total communication cost for adaptive sampling is  $sk \text{polylog}(\log n, s, k)$  bits.

751 After achieving an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation for the optimal solution, we can commu-  
 752 nicate a constant approximation of the number and cost of each cluster using a total  $\mathcal{O}(sk \log \log n)$   
 753 bits of communication. However, each site may assign the points to an incorrect cluster since  
 754 it only has an approximate location of  $S$ . Fortunately, the sensitivity sampling procedure of

---

756 Bansal et al. (2024) still returns a  $(1 + \varepsilon)$ -coreset if we assign each point to a center so that  
757 the distance between them is close to the distance between the point and the center closest to it.  
758 Therefore, combined with EFFICIENTCOMMUNICATION, we can upload the  $(1 + \varepsilon)$ -coreset using  
759  $\frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} \text{polylog}(\log n, k, \frac{1}{\varepsilon})$  bits, which achieves our desired bounds for the blackboard model.  
760

761 **B PRELIMINARIES**  
762

763 For a positive integer  $n$ , we denote by  $[n]$  the set  $\{1, \dots, n\}$ . We use  $\text{poly}(n)$  to represent an  
764 arbitrary polynomial function in  $n$ , and  $\text{polylog}(n)$  to denote a polynomial function in  $\log n$ . An  
765 event is said to occur with high probability if it holds with probability at least  $1 - 1/\text{poly}(n)$ .  
766

767 Throughout this paper, we focus on Euclidean  $(k, z)$ -clustering. Given vectors  $x, y \in \mathbb{R}^d$ , we let  
768  $\text{dist}(x, y)$  denote their Euclidean distance, defined as  $\|x - y\|_2$ , where  $\|x - y\|_2^2 = \sum_{i=1}^d (x_i - y_i)^2$ .  
769 For a point  $x$  and a set  $S \subset \mathbb{R}^d$ , we extend the notation  $\text{dist}(x, S) := \min_{s \in S} \text{dist}(x, s)$ . We also  
770 use  $\|x\|_z$  to denote the  $L_z$  norm of  $x$ , given by  $\|x\|_z = \sum_{i=1}^d x_i^z$ . Given a fixed exponent  $z \geq 1$  and  
771 finite sets  $X, C \subset \mathbb{R}^d$  with  $X = \{x_1, \dots, x_n\}$ , we define the clustering cost  $\text{cost}(X, C)$  as  
772

773 
$$\text{cost}(X, C) := \sum_{i=1}^n \text{dist}(x_i, C)^z.$$
  
774  
775

776 We now recall generalized versions of the triangle inequality.  
777

778 **Fact B.1** (Generalized triangle inequality). *For any  $z \geq 1$  and any points  $x, w, y \in \mathbb{R}^d$ , it holds that*

779 
$$\text{dist}(x, y)^z \leq 2^{z-1}(\text{dist}(x, w)^z + \text{dist}(w, y)^z).$$

780 **Fact B.2** (Claim 5 in Sohler & Woodruff (2018)). *Suppose  $z \geq 1$ ,  $x, y \geq 0$ , and  $\varepsilon \in (0, 1]$ . Then*

781 
$$(x + y)^z \leq (1 + \varepsilon) \cdot x^z + \left(1 + \frac{2z}{\varepsilon}\right)^z \cdot y^z.$$
  
782  
783

784 Next, we define the notion of a strong coresnet for  $(k, z)$ -clustering.  
785

786 **Definition B.3** (Coresnet). *Let  $\varepsilon > 0$  be an approximation parameter, and let  $X = \{x_1, \dots, x_n\} \subset$   
787  $\mathbb{R}^d$  be a set of points. A coresnet for  $(k, z)$ -clustering consists of a weighted set  $(S, w)$  such that for  
788 every set  $C \subset \mathbb{R}^d$  of  $k$  centers,*

789 
$$(1 - \varepsilon) \sum_{t=1}^n \text{dist}(x_t, C)^z \leq \sum_{q \in S} w(q) \text{dist}(q, C)^z \leq (1 + \varepsilon) \sum_{t=1}^n \text{dist}(x_t, C)^z.$$
  
790  
791

792 We will use the following known coresnet construction for  $(k, z)$ -clustering:  
793

794 **Theorem B.4.** (Cohen-Addad et al., 2022a; Huang et al., 2024; Cohen-Addad et al., 2022b) *For  
795 any  $\varepsilon \in (0, 1)$ , there exists a coresnet construction for Euclidean  $(k, z)$ -clustering that samples  
796  $\tilde{\mathcal{O}}\left(\min\left(\frac{1}{\varepsilon^2} k^{2-\frac{z}{z+2}}, \frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)\right)$  weighted points from the input dataset.*  
797

798 We also recall the classical Johnson-Lindenstrauss lemma, which enables dimensionality reduction  
799 while approximately preserving pairwise distances:  
800

801 **Theorem B.5** (Johnson-Lindenstrauss lemma). (Johnson & Lindenstrauss, 1984) *Let  $\varepsilon \in (0, 1/2)$   
802 and  $m = \mathcal{O}\left(\frac{1}{\varepsilon^2} \log n\right)$ . Given a set  $X \subset \mathbb{R}^d$  of  $n$  points, there exists a family of random linear  
803 maps  $\Pi : \mathbb{R}^d \rightarrow \mathbb{R}^m$  such that with high probability over the choice of  $\pi \sim \Pi$ , for all  $x, y \in X$ ,*

804 
$$(1 - \varepsilon) \|x - y\|_2 \leq \|\pi x - \pi y\|_2 \leq (1 + \varepsilon) \|x - y\|_2.$$
  
805

806 We remark that exist more efficient dimensionality reduction techniques for  $(k, z)$ -  
807 clustering (Makarychev et al., 2019; Izzo et al., 2021) though for the purposes of our guarantees,  
808 Johnson-Lindenstrauss suffices.  
809

Finally, we recall Hoeffding's inequality, a standard concentration bound:

---

810    **Theorem B.6** (Hoeffding's inequality). Let  $X_1, \dots, X_n$  be independent random variables with  $a_i \leq$   
 811     $X_i \leq b_i$  for each  $i$ . Let  $S_n = \sum_{i=1}^n X_i$ . Then for any  $t > 0$ ,

813    
$$\mathbb{P}(|S_n - \mathbb{E}[S_n]| > t) \leq 2 \exp \left( -\frac{t^2}{\sum_{i=1}^n (b_i - a_i)^2} \right).$$

816    We next recall the following formulation of sensitivity sampling paradigm, given in [Algorithm 2](#).

---

818    **Algorithm 2** SENSITIVITY SAMPLING

819    **Input:** Dataset  $X = \{x_1, \dots, x_n\} \subset [\Delta]^d$  and integer  $k$

820    **Output:** Weighted set  $A = \{(a_i, w_{a_i})\}$

821    1: Compute an  $\mathcal{O}(1)$ -approximation  $S = \{s_1, s_2, \dots, s_k\}$ . Let  $C_j \subset X$  be the cluster centered at  
 822     $s_j$ . For a point  $x \in C_j$ , let  $\Delta_p := \text{cost}(C_j, S)/|C_j|$  denote the average cost of  $C_j$

823    2: For  $x \in C_j$ , let

824    
$$\mu(x) := \frac{1}{4} \cdot \left( \frac{1}{k|C_j|} + \frac{\text{cost}(x, S)}{k \text{cost}(C_j, S)} + \frac{\text{cost}(x, S)}{\text{cost}(X, S)} + \frac{\Delta_x}{\text{cost}(X, S)} \right).$$

825    3:  $m \leftarrow \tilde{\mathcal{O}}(k/\varepsilon^2 \cdot \min(\varepsilon^{-2}, \varepsilon^{-z}))$ ,  $A \leftarrow \emptyset$

826    4: **for**  $i \leftarrow 1$  to  $m$  **do**

827    5:    Sample point  $a_i$  independently from the distribution  $\mu$ ,  $A \leftarrow A \cup \{(a_i, \frac{1}{m \cdot \mu(a_i)})\}$

828    6: **end for**

829    7: **return**  $A$

---

833    **Theorem B.7.** ([Bansal et al., 2024](#)) Sensitivity sampling, c.f., [Algorithm 2](#), outputs a  $(1 + \varepsilon)$ -coreset  
 834    of size  $\tilde{\mathcal{O}}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  for Euclidean  $(k, z)$ -clustering with probability at least 0.99.

835    We next recall the following distributed protocol for determining the larger of two integers.

836    **Theorem B.8.** ([Nisan, 1993](#)) Given two  $\mathcal{O}(\log n)$  bit integers  $X$  and  $Y$ , there exists a protocol  
 837    GREATER THAN that uses  $\tilde{\mathcal{O}}(\log \log n)$  total bits of communication and determines whether  $X =$   
 838     $Y$ ,  $X < Y$  or  $X > Y$ .

839    We recall the following protocol for applying the Johnson-Lindenstrauss transformation using low  
 840    communication cost.

841    **Theorem B.9.** ([Kane & Nelson, 2010](#)) For any integer  $d > 0$ , and any  $0 < \varepsilon, \delta < \frac{1}{2}$ , there exists a  
 842    family  $\mathcal{A}$  of matrices in  $\mathbb{R}^{\ell \times d}$  with  $\ell = \Theta(\varepsilon^{-2} \log \frac{1}{\delta})$  such that for any  $x \in \mathbb{R}^d$ ,

843    
$$\Pr_{A \in \mathcal{A}} [\|Ax\|_2 \notin [(1 - \varepsilon)\|x\|_2, (1 + \varepsilon)\|x\|_2]] < \delta.$$

844    Moreover,  $A \in \mathcal{A}$  can be sampled using  $\mathcal{O}(\log(\frac{1}{\delta}) \log d)$  random bits and every matrix  $A \in \mathcal{A}$  has  
 845    at most  $\alpha = \Theta(\varepsilon^{-1} \log(\frac{1}{\delta}) \log(\frac{1}{\delta}))$  non-zero entries per column, and thus  $Ax$  can be evaluated in  
 846     $\mathcal{O}(\alpha \cdot \|x\|_0)$  time if  $A$  is written explicitly in memory.

847    **Adaptive sampling.** We now introduce the adaptive sampling algorithm from [Aggarwal et al.](#)  
 848    ([2009](#)) and generalize the analysis to  $(k, z)$ -clustering.

849    ([Aggarwal et al., 2009](#)) showed that adaptive sampling achieves a bicriteria approximation for  $k$ -  
 850    means, i.e.,  $z = 2$ . For completeness, we shall extend the proof to show that it works for weighted  
 851    case and any  $z \geq 1$ , though we remark that the techniques are standard.

852    **Theorem B.10.** There exists an algorithm, c.f., [Algorithm 3](#) that outputs a set  $S$  of  $\mathcal{O}(k)$  points  
 853    such that with probability 0.99,  $\text{cost}(S, X) \leq \mathcal{O}(1) \cdot \text{cost}(C_{\text{OPT}}, X)$ , where  $C_{\text{OPT}}$  is an optimal  
 854     $(k, z)$ -clustering of  $X$ .

855    As stated, the version of adaptive sampling in [Algorithm 3](#) requires updating  $\text{cost}(x, S)$  after each  
 856    update, leading to a high cost of communication. We will introduce the lazy sampling algorithm.  
 857    The combination of our updated adaptive sampling algorithm and the lazy sampling algorithm can

---

864   **Algorithm 3** ADAPTIVESAMPLING  
 865  
 866   **Input:** Dataset  $X = \{x_1, \dots, x_n\} \subset [\Delta]^d$ ; approximation parameter  $\gamma \geq 1$   
 867   **Output:** Bicriteria approximation  $S$  for  $(k, z)$ -clustering on  $X$   
 868   1:  $S \leftarrow \emptyset, N \leftarrow \mathcal{O}(k), \gamma \leftarrow \Theta(1)$   
 869   2: **for**  $t \leftarrow 1$  to  $N$  **do**  
 870   3:   Choose  $s_t$  to be  $x_i \in X$  with probability  $\frac{d_i}{\sum d_i}$ , where  $\frac{1}{\gamma} \cdot d_i \leq \text{cost}(x_i, S) \leq \gamma \cdot d_i$   
 871   4:    $S \leftarrow S \cup \{s_t\}$   
 872   5: **end for**  
 873   6: **return**  $S$

---

874  
 875   provide a lower number of updates required, and thus a lower overall communication cost, while  
 876   still guaranteeing the result of the  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation.  
 877

878   Suppose that we have  $s$  sites, every site has a local dataset  $X_i$  and every data point  $x$  has a sampling  
 879   weight  $d_x$ . Let  $D_i = \sum_{x \in X_i} d_x$  and  $D = \sum_{i=1}^s D_i$ . Our goal is to sample a point  $x$  with prob-  
 880   ability  $p_x = \frac{d_x}{D}$ . In the case of adaptive sampling,  $d_x$  is just  $\text{cost}(x, S)$ . To avoid a high cost of  
 881   communication, our strategy is that the coordinator only holds  $\widetilde{D}_i$ , an approximation of real  $D_i$  that  
 882    $D_i \leq \widetilde{D}_i < \lambda D_i$ , and only updates  $\widetilde{D}_i$  when it deviates far from  $D_i$ . Then we can use  $\frac{d_x}{\widetilde{D}}$  to sample  
 883   the point, where  $\widetilde{D} = \sum_{i=1}^s \widetilde{D}_i$ .  
 884

---

885   **Algorithm 4** LAZYSAMPLING  
 886  
 887   **Input:**  $\{d_x\}_{x \in X}$ , the sampling weight;  $\{\widetilde{D}_i\}_{i \in [s]}$ , a constant approximation to  $\{D_i\}_{i \in [s]}$  that  $D_i \leq$   
 888    $\widetilde{D}_i \leq \lambda D_i$ , where  $D_i = \sum_{x \in X_i} d_x$  for every site  $i$   
 889   **Output:** Either  $\perp$  or a point  $x$  sampled under distribution  $\frac{d_x}{\widetilde{D}}$ , where  $D = \sum_{x \in X} d_x$   
 890   1:  $\widetilde{D} \leftarrow \sum_{i=1}^s \widetilde{D}_i, x \leftarrow \perp$   
 891   2: The coordinator generates an index  $i$  with probability  $= \frac{\widetilde{D}_i}{\widetilde{D}}$   
 892   3: The coordinator sends sampling request to site  $i$   
 893   4:  $x \leftarrow y$  with probability  $\frac{d_y}{\widetilde{D}_i}$   
 894   5: **return**  $x$

---

895  
 896  
 897   Since  $\widetilde{D}$  may be larger than  $D$ , the site  $i$  will send  $\perp$  with probability  $\frac{\widetilde{D}_i - D_i}{\widetilde{D}_i}$ . Fortunately, the  
 898   probability of returning  $\perp$  is constant, and the probability that we sample  $x$  conditioned on not  
 899   returning  $\perp$  is just  $\frac{d_x}{\widetilde{D}}$ . Therefore, by Markov's inequality, we can sample at least  $N$  points under  
 900   the distribution  $\frac{d_x}{\widetilde{D}}$  with probability at least 0.99 after repeating LAZYSAMPLING a total of  $\mathcal{O}(N)$  times.  
 901  
 902

903   **Lemma B.11.** *Let  $x$  be the result returned by Algorithm 4 and  $\mathcal{E}$  be the event that Algorithm 4 does  
 904   not return  $\perp$ . If  $\sum D_j \leq \sum \widetilde{D}_j < \lambda \sum D_j$ ,  $\Pr[\mathcal{E}] \geq \frac{1}{\lambda}$  and  $\Pr[x = y | \mathcal{E}] = \frac{d_y}{\widetilde{D}}$ . Furthermore,  
 905   there exists some constant  $\gamma \geq 1$  such that after running Algorithm 4 a total of  $\gamma N$  times, it will  
 906   return at least  $N$  points sampled by the distribution  $\frac{d_x}{\widetilde{D}}$  with probability at least 0.99.*  
 907

908  
 909   *Proof.* Let  $\mathcal{F}_i$  be the event that the result is sampled by the site  $i$ . Since  $\Pr[\mathcal{F}_i] = \frac{\widetilde{D}_i}{\widetilde{D}}$  and  
 910    $\Pr[\mathcal{E} | \mathcal{F}_i] = \sum_{x \in X_i} \frac{d_x}{\widetilde{D}_i} = \frac{D_i}{\widetilde{D}_i}$ , then by the law of total probability,  
 911

$$\Pr[\mathcal{E}] = \sum_{i=1}^s \Pr[\mathcal{E} | \mathcal{F}_i] \cdot \Pr[\mathcal{F}_i] = \sum_{i=1}^s \frac{D_i}{\widetilde{D}_i} \cdot \frac{\widetilde{D}_i}{\widetilde{D}} = \frac{D}{\widetilde{D}} \geq \frac{D}{\lambda D} = \frac{1}{\lambda}.$$

912  
 913  
 914  
 915   For any  $y \in X_i$ , we have

$$\Pr[x = y | \mathcal{E}] = \frac{\Pr[x = y]}{\Pr[\mathcal{E}]} = \frac{1}{\Pr[\mathcal{E}]} \cdot \frac{\widetilde{D}_i}{\widetilde{D}} \cdot \frac{d_y}{\widetilde{D}_i} = \frac{\widetilde{D}}{D} \cdot \frac{\widetilde{D}_i}{\widetilde{D}} \cdot \frac{d_y}{\widetilde{D}_i} = \frac{d_y}{D}.$$

918 Furthermore, let

$$Z_i = \begin{cases} 0, & \text{if LAZYSAMPLING returns } \perp \\ 1, & \text{otherwise.} \end{cases}$$

919 After running LAZYSAMPLING for  $\gamma N$  times, the number of times LAZYSAMPLING does not return  
920  $\perp$  is just  $\sum_{i=1}^{\gamma N} Z_i$ . Since  $\mathbb{E}[Z_i] = \Pr[\mathcal{E}] \geq \frac{1}{\gamma}$ , we have  $\mathbb{E}\left[\sum_{i=1}^{\gamma N} Z_i\right] \geq \frac{\gamma}{\lambda} N$ . By Markov's  
921 inequality, there exists some  $\gamma \geq 1$  such that  $\Pr\left[\sum_{i=1}^{\gamma N} Z_i < N\right] \leq 0.01$ . Hence, LAZYSAMPLING  
922 will sample at least  $N$  points with probability at least 0.99.  $\square$

923 Finally, in [Algorithm 5](#) we recall the standard algorithm that provides an efficient approximate en-  
924 coding of a number by storing the exponent of the number after rounding to a power of a fixed base  
925  $\lambda$ .

---

926 **Algorithm 5** POWERAPPROX

---

927 **Input:** Number to be encoded  $m$ , base  $\lambda > 1$

928 **Output:** An integer  $i$  that  $\lambda^i$  is a  $\lambda$  approximation of  $m$

929 1: Let  $i$  be the integer such that  $m \leq \lambda^i < \lambda m$

930 2: **return**  $i$

---

931 **Theorem B.12.** *The output from [Algorithm 5](#) can be used to compute a number  $\hat{m}$  such that  $m \leq \hat{m} < \lambda m$ .*

932 **Efficient encoding for coresnet construction for  $(k, z)$ -clustering.** We recall the following effi-  
933 cient encoding for a given coresnet for  $(k, z)$ -clustering given by [Cohen-Addad et al. \(2025\)](#). Given a  
934 dataset  $X$ , which may represent either the original inputs or a weighted coresnet derived from some  
935 larger dataset, we begin by computing a constant-factor approximation  $C'$  for the  $(k, z)$ -clustering  
936 problem on  $X$ . For every  $x \in X$ , let  $\pi_{C'}(x)$  denote the nearest center in  $C'$  to  $x$ . Each point  $x$   
937 can then be decomposed as  $x = \pi_{C'}(x) + (x - \pi_{C'}(x))$ , separating it into its closest center and a  
938 residual vector.

939 To obtain a  $(1 + \varepsilon)$ -approximation for  $(k, z)$ -clustering, it suffices to round each coordinate of the  
940 offset vector  $x - \pi_{C'}(x)$  to the nearest power of  $(1 + \varepsilon')$ , where  $\varepsilon' = \text{poly}\left(\varepsilon, \frac{1}{d}, \frac{1}{\log(n\Delta)}\right)$ . Let  $y'$   
941 denote this rounded vector.

942 We then encode each point as  $x' = \pi_{C'}(x) + y'$ , storing both the index of the center  $\pi_{C'}(x)$  and the  
943 exponent values of the rounded offset. This representation uses  $\mathcal{O}\left(\log k + d \log\left(\frac{1}{\varepsilon}, d, \log(n\Delta)\right)\right)$   
944 bits per point. The full algorithm is detailed in [Algorithm 6](#).

---

945 **Algorithm 6** Compact Encoding for Coresnet Generation in  $(k, z)$ -Clustering

---

946 **Input:** Dataset  $X \subset [\Delta]^d$  with weights  $w(\cdot)$ , accuracy parameter  $\varepsilon \in (0, 1)$ , number of clusters  $k$ ,  
947 parameter  $z \geq 1$ , failure probability  $\delta \in (0, 1)$

948 **Output:**  $(1 + \varepsilon)$ -approximate coresnet for  $(k, z)$ -clustering

949 1:  $\varepsilon' \leftarrow \frac{\text{poly}(\varepsilon^z)}{\text{poly}(k, \log(nd\Delta))}$

950 2: Compute a constant-factor  $(k, z)$ -clustering solution  $C'$  on  $X$

951 3: **for** each  $x \in X$  **do**

952 4: Set  $c'(x)$  as the nearest center to  $x$  in  $C'$

953 5: Compute the residual vector  $y' = x - c'(x)$

954 6: Round each coordinate of  $y'$  to the nearest power of  $(1 + \varepsilon')$  to obtain  $y$

955 7: Define  $x' = (c'(x), y)$ , where  $y$  stores the exponent for each coordinate

956 8: Add  $x'$  to the new set:  $X' \leftarrow X' \cup \{x'\}$

957 9: **end for**

958 10: **return**  $(C', X')$

---

959 We have the following guarantees for the efficient encoding from [Cohen-Addad et al. \(2025\)](#):

---

972    **Lemma B.13.** *Cohen-Addad et al. (2025)* Let  $\varepsilon \in (0, \frac{1}{2})$ , and let  $X'$  be the weighted dataset  $S$ ,  
 973    constructed using the offsets from the center set  $C'$  as defined in [Algorithm 6](#). Then, for any set of  
 974    centers  $C \subset [\Delta]^d$  with  $|C| \leq k$ , the following holds:  
 975

$$(1 - \varepsilon) \cdot \text{cost}(C, X) \leq \text{cost}(C, X') \leq (1 + \varepsilon) \cdot \text{cost}(C, X).$$

977    **Lemma B.14.** *Cohen-Addad et al. (2025)* Let  $X$  be a coresset where the point weights lie within the  
 978    range  $[1, \text{poly}(nd\Delta)]$ . Then the transformed set  $X'$  forms a  $(1 + \varepsilon)$ -strong coresset for  $X$ , and its  
 979    total space requirement is  $\mathcal{O}(dk \log(n\Delta)) + |X| \cdot \text{polylog}(k, \frac{1}{\varepsilon}, \log(nd\Delta), \log \frac{1}{\delta})$  bits.  
 980

981    **C ADAPTIVE SAMPLING FOR  $(k, z)$ -CLUSTERING**

984    We first give a high-level overview of why ADAPTIVESAMPLING returns a bicriteria approximation.  
 985    Let  $\{A_j\}_{j=1}^k$  be the clusters that correspond to an optimal  $(k, z)$ -clustering. Then we can  
 986    achieve an  $\mathcal{O}(1)$ -approximation to the optimal cost if the cost for every cluster  $A_j$  induced by  $S$   
 987    is already an  $\mathcal{O}(1)$ -approximation for  $\text{cost}(A_j, C_{\text{OPT}})$ , where  $C_{\text{OPT}}$  is the set of centers for the  
 988    optimal solution. We define a cluster  $A_j$  as a good cluster if its cost is an  $\mathcal{O}(1)$ -approximation to  
 989     $\text{cost}(A_j, C_{\text{OPT}})$ , and otherwise we call  $A_j$  a bad cluster. We can show that with constant probability,  
 990    ADAPTIVESAMPLING samples a center that can transform a bad cluster to a good one. Then, by  
 991    Markov's inequality, it follows that we can eliminate all bad clusters in  $\mathcal{O}(k)$  rounds of sampling,  
 992    resulting in a constant-factor approximation.

993    We use the following definition of good and bad clusters.

994    **Definition C.1.** Let  $S_i$  be the sampled set  $S$  in the  $i$ -th round in [Algorithm 3](#) and let  $C_{\text{OPT}} =$   
 995     $\{c_1, c_2, \dots, c_k\}$  be an optimal solution for  $(k, z)$ -clustering.

996    For  $j \in [k]$ , let  $A_j$  be the points in  $X$  assigned to  $c_j$  in the optimal clustering, breaking ties arbitrarily, i.e.,  $A_j = \{x \in X : \text{dist}(x, c_j) \leq \text{dist}(x, c_\ell), \forall \ell \in [k]\}$ . We define

$$\begin{aligned} \mathbf{Good}_i &= \{A_j : \text{cost}(A_j, S_i) \leq \gamma_z \cdot \text{cost}(A_j, C_{\text{OPT}})\} \\ \mathbf{Bad}_i &= \{A_1, A_2, \dots, A_k\} \setminus \mathbf{Good}_i \end{aligned}$$

1001    where  $\gamma_z = 2 + (3 + 6z)^z$ .  
 1002

1003    By definition, if every cluster is a good cluster, we can guarantee that  $S$  is an  $(\mathcal{O}(1), \mathcal{O}(1))$ -  
 1004    bicriteria approximation. We will prove that either  $S$  is already an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approx-  
 1005    imation, or else we will reduce the number of bad clusters after every sampling with some constant  
 1006    probability, which would imply that we can achieve an  $\mathcal{O}(1)$ -approximation after sampling  $\mathcal{O}(k)$   
 1007    points.

1008    **Lemma C.2.** Suppose  $\text{cost}(X, S_i) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{\text{OPT}})$ , then

$$\Pr[|\mathbf{Bad}_{i+1}| < |\mathbf{Bad}_i|] \geq \delta$$

1011    for some constant  $\delta > 0$ .

1012    **Lemma C.2** is the generalization of [Lemma 5](#) in [Aggarwal et al. \(2009\)](#). We will prove **Lemma C.2**  
 1013    by breaking the proof into several lemmas. We first consider a specific parameterization of the  
 1014    generalized triangle inequality.  
 1015

1016    **Lemma C.3.** For any  $x, y, \mu \in \mathbb{R}^d$ ,

$$\text{dist}(x, y)^z \leq 2 \cdot \text{dist}(x, \mu)^z + (1 + 2z)^z \cdot \text{dist}(y, \mu)^z.$$

1019    *Proof.* The claim follows from applying [Fact B.2](#) with  $\varepsilon = 1$ , so that by the triangle inequality,  
 1020     $\text{dist}(x, y)^z \leq (\text{dist}(x, \mu) + \text{dist}(y, \mu))^z \leq 2 \cdot \text{dist}(x, \mu)^z + (1 + 2z)^z \cdot \text{dist}(y, \mu)^z$ .  $\square$   
 1021

1022    We first show that we will sample a point from a bad cluster with constant probability every round.  
 1023    This statement is analogous to [Lemma 1](#) in [Aggarwal et al. \(2009\)](#).

1025    **Lemma C.4.** In the  $i$ -th round of our algorithm, either  $\text{cost}(X, S_i) \leq 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{\text{OPT}})$  or  
 1026    else the probability of picking a point from some cluster in  $\mathbf{Bad}_i$  is at least  $\frac{1}{2}$ .

1026 *Proof.* Suppose  $\text{cost}(X, S_{i-1}) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{\text{OPT}})$ . Then the probability of picking  $x$  from  
1027 some bad cluster is

$$1029 \quad \Pr[x \in \mathbf{Bad}_i] = \frac{\sum_{A_j \in \mathbf{Bad}_i} \sum_{x_l \in A_j} d_l}{\sum_{x_l \in X} d_l} = 1 - \frac{\sum_{A_j \in \mathbf{Good}_i} \sum_{x_l \in A_j} d_l}{\sum_{x_l \in X} d_l}.$$

1031 Here, we remark that  $d_l$  is the distance from  $x_l$  to  $S_{i-1}$  as defined in [Algorithm 3](#). Since  
1032  $\text{cost}(A_j, S_{i-1}) \leq \gamma_z \text{cost}(A_j, C_{\text{OPT}})$  if  $A_j \in \mathbf{Good}_i$ , and  $\text{cost}(X, S_{i-1}) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{\text{OPT}})$   
1033 by the condition we hold,

$$1035 \quad \Pr[x \in \mathbf{Bad}_i] \geq 1 - \frac{\gamma^2 \cdot \gamma_z \sum_{A_j \in \mathbf{Good}_i} \text{cost}(A_j, C_{\text{OPT}})}{2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{\text{OPT}})} \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

1038  $\square$

1040 Consider a fixed bad cluster  $A \in \mathbf{Bad}_i$ . Let  $m = w(A) = \sum_{x \in A} w(x)$ , where  $w(x)$  is the weight  
1041 of  $x$ , and define  $r = \left(\frac{\text{cost}(A, C_{\text{OPT}})}{m}\right)^{1/z}$ . Denote  $\mu$  as the center in  $C_{\text{OPT}}$  corresponding to  $A$ . Let  $y$   
1042 be the point closest to  $\mu$  in  $S_{i-1}$ . We will show that  $y$  (and thus every sampled point in  $S_{i-1}$ ) is far  
1043 from  $\mu$ .

1044 **Lemma C.5.**  $\text{dist}(y, \mu) \geq 3r$ .

1045 *Proof.* Because  $A$  is a fixed bad cluster,

$$1049 \quad \gamma_z \text{cost}(A, C_{\text{OPT}}) \leq \text{cost}(A, S_{i-1}) = \sum_{x \in A} w(x) \min_{c \in S_{i-1}} \text{dist}(x, c)^z \leq \sum_{x \in A} w(x) \text{dist}(x, y)^z.$$

1050 By [Lemma C.3](#),

$$1053 \quad \sum_{x \in A} w(x) \text{dist}(x, y)^z \leq \sum_{x \in A} (2 \cdot w(x) \cdot \text{dist}(x, \mu)^z + (1 + 2z)^z \cdot w(x) \cdot \text{dist}(y, \mu)^z) \\ 1054 \quad = 2 \text{cost}(A, C_{\text{OPT}}) + (1 + 2z)^z \cdot m \cdot \text{dist}(y, \mu)^z.$$

1055 Hence

$$1056 \quad (1 + 2z)^z \cdot m \cdot \text{dist}(y, \mu)^z \geq (\gamma_z - 2) \text{cost}(A, C_{\text{OPT}}).$$

1057 Since  $\gamma_z = 3^z(1 + 2z)^z + 2$ ,

$$1058 \quad 3r = \left( \frac{\gamma_z - 2}{(1 + 2z)^z} \cdot \frac{\text{cost}(A, C_{\text{OPT}})}{m} \right)^{1/z} \leq \text{dist}(y, \mu).$$

1059  $\square$

1060 Define  $B(\alpha) = \{x \in A : \text{dist}(x, \mu) \leq \alpha r\}$  as the points in the fixed bad cluster  $A$  that are within  
1061 distance  $\alpha r$  from  $\mu$ . We will show that the bad cluster  $A$  can be transformed into a good cluster if  
1062 we sample some point  $x$  close to the center  $\mu$ . This statement is analogous to Lemma 2 in [Aggarwal et al. \(2009\)](#).

1063 **Lemma C.6.** Let  $A$  be any fixed bad cluster defined by  $C_{\text{OPT}}$  and let  $b \in B(\alpha)$ , for  $0 \leq \alpha \leq 3$ .  
1064 Then

$$1065 \quad \text{cost}(A, S_{i-1} \cup \{b\}) \leq \gamma_z \cdot \text{cost}(A, C_{\text{OPT}}).$$

1066 *Proof.*

$$1067 \quad \text{cost}(A, S_{i-1} \cup \{b\}) = \sum_{x \in A} w(x) \min_{c \in S_{i-1} \cup \{b\}} \text{dist}(x, c)^z \leq \sum_{x \in A} w(x) \cdot \text{dist}(x, b)^z.$$

1068 By [Lemma C.3](#),

$$1069 \quad \sum_{x \in A} w(x) \cdot \text{dist}(x, b)^z \leq \sum_{x \in A} (2 \cdot w(x) \cdot \text{dist}(x, \mu)^z + (1 + 2z)^z \cdot w(x) \cdot \text{dist}(b, \mu)^z).$$

1080 Since  $b \in B(\alpha)$  and  $0 \leq \alpha \leq 3$ ,  $\text{dist}(b, \mu) \leq \alpha r \leq 3r$ . Hence

$$\begin{aligned} 1082 \quad \text{cost}(A, S_{i-1} \cup \{b\}) &\leq 2 \text{cost}(A, C_{\text{OPT}}) + m(1+2z)^z (3r)^z \\ 1083 &= 2 \text{cost}(A, C_{\text{OPT}}) + 3^z (1+2z)^z \text{cost}(A, C_{\text{OPT}}) \\ 1084 &= \gamma_z \text{cost}(A, C_{\text{OPT}}). \end{aligned}$$

1085  $\square$

1087 We next show that most of the weights of the points of the fixed bad cluster  $A$  fall into  $B(\alpha)$ . Recall  
1088 that we define  $m = w(A) = \sum_{x \in A} w(x)$ . Let  $w(B(\alpha)) = \sum_{x \in B(\alpha)} w(x)$ . This statement is  
1089 analogous to Lemma 3 in [Aggarwal et al. \(2009\)](#).

1090 **Lemma C.7.**

$$1092 \quad w(B(\alpha)) \geq m \left(1 - \frac{1}{\alpha^z}\right), \text{ for } 1 \leq \alpha \leq 3.$$

1094 *Proof.*

$$1096 \quad \text{cost}(A, C_{\text{OPT}}) \geq \text{cost}(A \setminus B(\alpha), C_{\text{OPT}}) = \sum_{x \in A \setminus B(\alpha)} w(x) \min_{c \in C_{\text{OPT}}} \text{dist}(x, c)^z.$$

1099 Since for any  $x \in A$ ,  $\mu$  is the nearest center to  $x$  in  $C_{\text{OPT}}$ , then

$$1100 \quad \sum_{x \in A \setminus B(\alpha)} w(x) \min_{c \in C_{\text{OPT}}} \text{dist}(x, c)^z = \sum_{x \in A \setminus B(\alpha)} w(x) \text{dist}(x, \mu)^z.$$

1103 Since for any  $x \in A \setminus B(\alpha)$ ,  $\text{dist}(x, \mu) \geq \alpha r$ , then

$$1104 \quad \sum_{x \in A \setminus B(\alpha)} w(x) \text{dist}(x, \mu)^z \geq w(A \setminus B(\alpha)) \cdot (\alpha r)^z = \left(1 - \frac{w(B(\alpha))}{m}\right) m(\alpha r)^z.$$

1107 Since  $r = \left(\frac{\text{cost}(A, C_{\text{OPT}})}{m}\right)^{1/z}$ ,

$$1110 \quad \text{cost}(A, C_{\text{OPT}}) \geq \left(1 - \frac{w(B(\alpha))}{m}\right) m(\alpha r)^z = \left(1 - \frac{w(B(\alpha))}{m}\right) \alpha^z \text{cost}(A, C_{\text{OPT}}).$$

1112 Therefore,

$$1113 \quad w(B(\alpha)) \geq m \left(1 - \frac{1}{\alpha^z}\right).$$

1115  $\square$

1117 We next show that the cost of points in  $B(\alpha)$  is at least a constant fraction of the cost of the bad  
1118 cluster  $A$  and thus we will sample a point near  $\mu$  with constant probability if we sample a point from  
1119  $A$ . This statement is analogous to Lemma 4 in [Aggarwal et al. \(2009\)](#).

1120 **Lemma C.8.**

$$1122 \quad \Pr[x \in B(\alpha) | x \in A \text{ and } A \in \mathbf{Bad}_i] \geq \frac{1}{\gamma^2} \left(1 - \frac{1}{\alpha^z}\right) \frac{(3-\alpha)^z}{\gamma_z}.$$

1124 *Proof.* Recall that  $y$  is the point closest to  $\mu$  in  $S_{i-1}$ . We have

$$1126 \quad \text{cost}(A, S_{i-1}) = \sum_{x \in A} w(x) \min_{c \in S_{i-1}} \text{dist}(x, c)^z \leq \sum_{x \in A} w(x) \text{dist}(x, y)^z.$$

1128 Let  $d = \text{dist}(y, \mu)$ . By [Lemma C.3](#),

$$1130 \quad \text{cost}(A, S_{i-1}) \leq \sum_{x \in A} \left(2w(x) \text{dist}(x, \mu)^z + (1+2z)^z w(x) \text{dist}(y, \mu)^z\right) = 2 \text{cost}(A, C_{\text{OPT}}) + m(1+2z)^z d^z.$$

1133 Since  $r = \left(\frac{\text{cost}(A, C_{\text{OPT}})}{m}\right)^{1/z}$ , then  $\text{cost}(A, S_{i-1}) \leq 2mr^z + m(1+2z)^z d^z$ .

1134 On the other hand,  $\text{cost}(B(\alpha), S_{i-1}) = \sum_{x \in B(\alpha)} w(x) \min_{c \in S_{i-1}} \text{dist}(x, c)^z$ . By triangle inequality,  
 1135  $\text{dist}(x, c) \geq \text{dist}(c, \mu) - \text{dist}(x, \mu) \geq \text{dist}(y, \mu) - \text{dist}(x, \mu)$ . Since  $x \in A$  and  $A$  is a bad cluster,  
 1136 then  $\text{dist}(y, \mu) - \text{dist}(x, \mu) \geq 0$ , so that  
 1137

$$1138 \text{cost}(B(\alpha), S_{i-1}) \geq \sum_{x \in B(\alpha)} w(x) (\text{dist}(c, \mu) - \text{dist}(x, \mu))^z \geq \sum_{x \in B(\alpha)} w(x) (\text{dist}(y, \mu) - \text{dist}(x, \mu))^z.$$

1140 For  $x \in B(\alpha)$ , we have  $\text{dist}(x, \mu) \leq \alpha r$ , so that  $\text{cost}(B(\alpha), S_{i-1}) \geq \sum_{x \in B(\alpha)} w(x) (\text{dist}(y, \mu) -$   
 1141  $\alpha r)^z$ . By [Lemma C.7](#),  $w(B(\alpha)) \geq m \left(1 - \frac{1}{\alpha^z}\right)$ , so  $\text{cost}(B(\alpha), S_{i-1}) \geq m \left(1 - \frac{1}{\alpha^z}\right) (\text{dist}(y, \mu) -$   
 1142  $\alpha r)^z$ .  
 1143

1144 Since  $\frac{1}{\gamma} \cdot d_i \leq \text{cost}(x_i, S) \leq \gamma \cdot d_i$ , cf. [Algorithm 3](#), therefore  
 1145

$$1147 \Pr[x \in B(\alpha) | x \in A \text{ and } A \in \mathbf{Bad}_i] = \frac{\sum_{x_i \in B(\alpha)} d_i}{\sum_{x_i \in A} d_i} \geq \frac{\frac{1}{\gamma} \text{cost}(B(\alpha), S_{i-1})}{\gamma \text{cost}(A, S_{i-1})} \\ 1148 \geq \frac{m}{\gamma^2} \left(1 - \frac{1}{\alpha^z}\right) \cdot \frac{(\text{dist}(y, \mu) - \alpha r)^z}{m(2r^z + (1+2z)^z \text{dist}(y, \mu)^z)}.$$

1152 By computing the derivative, we can observe that  
 1153

$$1154 \frac{(\text{dist}(y, \mu) - \alpha r)^z}{2r^z + m(1+2z)^z \text{dist}(y, \mu)^z}$$

1156 is an increasing function of  $\text{dist}(y, \mu)$  for  $\text{dist}(y, \mu) \geq 3r \geq \alpha r$ . Hence  
 1157

$$1158 \Pr[x \in B(\alpha) | x \in A \text{ and } A \in \mathbf{Bad}_i] \geq \frac{m}{\gamma^2} \left(1 - \frac{1}{\alpha^z}\right) \cdot \frac{(3r - \alpha r)^z}{m(2r^z + (1+2z)^z (3r)^z)} \\ 1159 = \frac{1}{\gamma^2} \left(1 - \frac{1}{\alpha^z}\right) \frac{(3 - \alpha)^z}{\gamma_z}.$$

1163 □  
 1164

1165 Therefore, we will transform a bad cluster  $A$  into a good cluster if we sample a point in  $B(\alpha) \subset A$   
 1166 for  $a \leq 3$ , and we will sample such a point in  $A$  with constant probability, which means that we will  
 1167 reduce the number of bad clusters with constant probability.

1168 **Lemma C.9.** *Suppose the point  $x$  picked by our algorithm in the  $i$ -th round is from  $A \in \mathbf{Bad}_i$  and  
 1169  $S_i = S_{i-1} \cup \{x\}$ . Then*

$$1170 \Pr[\text{cost}(A, S_i) \leq \gamma_z \cdot \text{cost}(A, C_{OPT}) | x \in A \text{ and } A \in \mathbf{Bad}_i] \geq \delta,$$

1171 where

$$1173 \delta = \max \frac{1}{\gamma^2} \left(1 - \frac{1}{\alpha^z}\right) \frac{(3 - \alpha)^z}{\gamma_z}.$$

1175 *Proof.* The claim follows from [Lemma C.6](#) and [Lemma C.8](#). □  
 1176

1177 If  $\text{cost}(X, S_i) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ , we will sample a point from a bad cluster with constant  
 1178 probability, and a point sampled from a bad cluster will transform it to a good one with probability.  
 1179 Hence, we will reduce the number of bad clusters with probability.  
 1180

1181 We have proven that we will sample a point from a bad cluster with constant probability. We also  
 1182 proved that a sampled center from a bad cluster will transfer that bad cluster to a good one with  
 1183 constant probability. These two facts lead us to the conclusion that we will transfer a bad cluster to  
 1184 a good one with constant probability for every sampling, which is what [Lemma C.2](#) claimed.

1185 **Lemma C.2.** *Suppose  $\text{cost}(X, S_i) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ , then*

$$1186 \Pr[|\mathbf{Bad}_{i+1}| < |\mathbf{Bad}_i|] \geq \delta$$

1187 for some constant  $\delta > 0$ .

---

1188 *Proof.* If  $\text{cost}(X, S_i) > 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ , by [Lemma C.4](#), we will sample a point  $x$  from  
 1189 some bad cluster with probability at least  $\frac{1}{2}$ . By [Lemma C.9](#), if  $x \in A$  is from some bad cluster,  
 1190  $\text{cost}(A, S_i) \leq \gamma_z \cdot \text{cost}(A, C_{OPT})$  with probability at least  $\delta$ . Hence  $A$  will become a good cluster  
 1191 in  $\mathbf{Good}_{i+1}$  with probability at least  $\frac{\delta}{2}$ . Therefore,  $\Pr[|\mathbf{Bad}_{i+1}| < |\mathbf{Bad}_i|] \geq \frac{\delta}{2}$ .  $\square$   
 1192

1193 Now we justify the correctness of adaptive sampling for general  $z \geq 1$ .  
 1194

1195 **Theorem B.10.** *There exists an algorithm, c.f., [Algorithm 3](#) that outputs a set  $S$  of  $\mathcal{O}(k)$  points  
 1196 such that with probability 0.99,  $\text{cost}(S, X) \leq \mathcal{O}(1) \cdot \text{cost}(C_{OPT}, X)$ , where  $C_{OPT}$  is an optimal  
 1197  $(k, z)$ -clustering of  $X$ .*

1198  
 1199 *Proof.* By [Lemma C.3](#), the number of bad clusters will decrease with constant probability every  
 1200 round, unless  $\text{cost}(X, S_i) \leq 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ . Thus, by Markov's inequality, with probabili-  
 1201 ty 0.99, the number of bad clusters will be reduced to 0 after  $\mathcal{O}(k)$  rounds unless  $\text{cost}(X, S_i) \leq 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ . Consequently, either there are no bad clusters or  $\text{cost}(X, S_i) \leq 2\gamma^2 \cdot \gamma_z \text{cost}(X, C_{OPT})$ . Both of these cases mean that we have  $\text{cost}(A_j, S) \leq 2\gamma^2 \cdot \gamma_z \cdot \text{cost}(A_j, C_{OPT})$   
 1202 for all  $j \in [k]$ , so  $S$  becomes a constant approximation to the optimal  $(k, z)$ -clustering.  $\square$   
 1203

1204

## 1206 D BLACKBOARD MODEL OF COMMUNICATION

### 1208 D.1 BICRITERIA APPROXIMATION IN THE BLACKBOARD MODEL

1209 We first give the initialization algorithm for  $(k, z)$ -clustering of blackboard model in [Algorithm 7](#),  
 1210 which generates a point  $x$  and uploads an 2-approximation  $\widetilde{D}_i$  of the total cost  $D_i$  of every site on  
 1211 the blackboard.

---

#### 1214 **Algorithm 7** INITIALIZATION

1216 **Input:** Dataset  $X_i$  given to each site  $i \in [s]$

1217 **Output:** A set  $S$  with one sampled point  $x$ , approximate cost  $\widetilde{D}_i$  for every site on blackboard

1218 1: The coordinator samples a point  $x$  and upload it on blackboard,  $S \leftarrow \{x\}$

1219 2: **for**  $i \leftarrow 1$  to  $s$  **do**

1220 3:    $r_i \leftarrow \text{POWERAPPROX}(\text{cost}(X_i, S), 2)$

1221 4:   Write  $r_i$  on the blackboard

1222 5: **end for**

1223 6:  $D_i \leftarrow 2^{r_i}$

1224 7: **return**  $S, \widetilde{D}_i$

1225

1226 After INITIALIZATION, we use LAZYSAMPLING to sample  $\mathcal{O}(k)$  points to get an  $(\mathcal{O}(1), \mathcal{O}(1))$ -  
 1227 bicriteria approximation for the  $(k, z)$ -clustering.

1228

---

#### 1229 **Algorithm 8** Lazy Adaptive $(k, z)$ -Sampling for $k = \mathcal{O}(\log n)$

1230 **Input:** The dataset  $X_i$  every site  $i$  owns,  $i \in [s]$

1231 **Output:** A set  $S$  that is an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation for the  $(k, z)$ -clustering

1232 1: Run INITIALIZATION to get  $S$  and  $\widetilde{D}_i$

1233 2:  $N \leftarrow \mathcal{O}(k)$

1234 3: **for**  $i \leftarrow 1$  to  $N$  **do**

1235 4:    $x \leftarrow \text{LAZYSAMPLING}(\{\text{cost}(x, S)\}, \{\widetilde{D}_j\})$ , and upload  $x$  on blackboard

1236 5:    $S \leftarrow S \cup \{x\}$

1237 6:   Every site  $j$  computes  $D_j = \text{cost}(X_j, S)$

1238 7:    $r_j \leftarrow \text{POWERAPPROX}(D_j, 2)$ , every site  $j$  update  $r_j$  on the blackboard if  $r_j \neq \log_2 \widetilde{D}_j$

1239 8:    $D_j \leftarrow 2^{r_j}$

1240 9: **end for**

1241 10: **return**  $S$

We first show that [Algorithm 8](#) returns an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation for the  $(k, z)$ -clustering with  $\mathcal{O}(sk \log \log n + kd \log n)$  bits of communication and  $\mathcal{O}(k)$  rounds of communication.

**Lemma D.1.** *[Algorithm 8](#) outputs a set  $S$  with size  $\mathcal{O}(k)$  using  $\tilde{\mathcal{O}}(sk + kd \log n)$  bits of communication and  $\mathcal{O}(k)$  rounds of communication, such that with probability at least 0.98,  $\text{cost}(S, X) \leq \mathcal{O}(1) \cdot \text{cost}(C_{\text{OPT}}, X)$ , where  $C_{\text{OPT}}$  is the optimal  $(k, z)$ -clustering of  $X$ .*

*Proof.* We show the correctness of the algorithm, i.e., that the clustering cost induced by  $S$  is a constant-factor approximation of the optimal  $(k, z)$ -clustering cost. We also upper bound the number of points in  $S$ .

**Bicriteria approximation guarantee.** Since we update  $\tilde{D}_j$  every round, we meet the condition  $D_j \leq \tilde{D}_j < 2D_j$  every time we run [LAZYSAMPLING](#)( $\{\text{cost}(x, S)\}, \{\tilde{D}_j\}$ ) in the algorithm. Thus, by [Lemma B.11](#), we sample a point  $x$  with probability  $\frac{\text{cost}(x, S)}{\text{cost}(X, S)}$  if [LAZYSAMPLING](#) does not return  $\perp$ , and we can get  $\mathcal{O}(k)$  points which are not  $\perp$  with probability at least 0.99 after  $N = \mathcal{O}(k)$  rounds of lazy sampling. Therefore, we will get  $\mathcal{O}(k)$  sampled points that satisfy the required conditions for adaptive sampling in [ADAPTIVESAMPLING](#) ([Algorithm 3](#)). Therefore, by [Lemma C.2](#), we reduce the size of  $\text{Bad}_i$  with some constant probability  $p$  every round. It follows from [Theorem B.10](#) that [Algorithm 8](#) produces a constant approximation for  $(k, z)$ -clustering with probability at least 0.99.

**Communication complexity of Algorithm 8.** [INITIALIZATION](#) uses  $\mathcal{O}(s \log \log n + d \log n)$  bits of communication. Uploading the location of a sampled point  $x$  requires  $\mathcal{O}(d \log n)$  bits because there are  $d$  coordinates that need to be uploaded, and each needs  $\mathcal{O}(\log n)$  bits. Every site needs  $\mathcal{O}(1)$  bits to return  $\perp$  to the coordinator. Then, all  $s$  sites need to write  $r_i$  on the blackboard. Since  $r_i = \mathcal{O}(\log n)$ , and encoding  $r_i$  requires  $\mathcal{O}(\log \log n)$  bits, it leads to  $\mathcal{O}(s \log \log n)$  bits of communication. Since we will repeat the iteration for  $\mathcal{O}(k)$  times, the total communication cost would be  $\mathcal{O}(sk \log \log n + kd \log n)$  bits. Finally, we remark that since we will use two rounds of communication for each iteration of the  $N = \mathcal{O}(k)$  rounds of sampling, the total number of rounds of communication is  $\mathcal{O}(k)$ .  $\square$

## D.2 $L_1$ SAMPLING SUBROUTINE

In this section, we introduce an  $L_1$  sampling algorithm in [Algorithm 9](#). The main purpose of the  $L_1$  sampling algorithm is to ultimately improve the communication complexity and the round complexity of the adaptive sampling approach for  $k = \Omega(\log n)$ , c.f., [Appendix D.3](#).

The sampling algorithm [L1SAMPLING](#) can detect whether  $\tilde{D}$  is a good approximation of  $D$ . When  $\mu^2 D \leq \tilde{D}$ , which means that  $\tilde{D}$  is far from a good approximation, [L1SAMPLING](#) will return True to notify us to update the value of  $\tilde{D}$ . On the other hand, when  $\mu D > \tilde{D}$ , which means  $\tilde{D}$  is a good enough approximation of  $D$ , [L1SAMPLING](#) will return False, so we can save communication cost by avoiding unnecessary updates.

We now justify the correctness of [Algorithm 9](#), i.e., we show that if the total mass has decreased significantly, then the algorithm will return True with probability  $1 - \delta$  and similarly if the total mass has not decreased significantly, then the algorithm will return False with probability  $1 - \delta$ .

**Lemma D.2.** *Suppose  $\tilde{D}_j \geq D_j$  for all  $j \in [s]$ , and  $\tilde{D} \geq D$ . If  $\mu^2 D \leq \tilde{D}$ , [Algorithm 9](#) [L1SAMPLING](#)( $\{\tilde{D}_j\}, \{D_j\}, \mu, \delta$ ) will return True with probability at least  $1 - \delta$ . If  $\mu D > \tilde{D}$ , [L1SAMPLING](#)( $\{\tilde{D}_j\}, \{D_j\}, \mu, \delta$ ) will return False with probability at least  $1 - \delta$ .*

*Proof.* Let  $\widehat{D}_j = \lambda^{r_j}$ . Since  $r_j$  is returned by [POWERAPPROX](#)( $D_j, \lambda$ ), then by [Theorem B.12](#), we have  $D_j \leq \lambda^{r_j} = \widehat{D}_j < \lambda D_j$ .

Define the random variable

$$Z_i = \frac{1}{p_j} \widehat{D}_j, \text{ with probability } p_j = \frac{\widehat{D}_j}{\tilde{D}}.$$

1296

1297

---

**Algorithm 9**  $L_1$ -Sampling: L1SAMPLING( $\{\tilde{D}_j\}$ ,  $\{D_j\}$ ,  $\mu$ ,  $\delta$ )

1298 **Input:**  $D_i = \text{cost}(X_i, S)$ , the cost of points in site  $i$ ; total cost  $D = \sum_{i=1}^s D_i$ ;  $\{\widetilde{D}_i\}_{i=1}^s$  and  $\widetilde{D}$   
 1299 written on blackboard, which are approximations to  $\{D_i\}_{i=1}^s$  and  $D$ ;  $\mu > 1$ , the distortion  
 1300 parameter;  $\delta > 0$ , the failure probability

**Output:** A boolean indicator for whether  $\tilde{D}$  is a  $\mu^2$ -approximation of  $D$ .

$$1302 \quad 1: N \leftarrow \mathcal{O}(\log \frac{1}{\delta}), T \leftarrow 0, \lambda \leftarrow \frac{\mu}{4}, \alpha \leftarrow 2\mu$$

2: **for**  $i \leftarrow 1$  to  $N$  **do**

1304 3: Sample a site  $j$  with probability  $\Pr[j] = \frac{D_j}{D}$ , update  $r_j \leftarrow \text{POWERAPPROX}(D_j, \lambda)$  on  
 1305 blackboard

$$4: \quad T \leftarrow T + \frac{1}{\Pr[j]} \cdot \lambda^{r_j}$$

## 1307 5: **end for**

6: **if**  $\alpha \cdot T$

1309 7: **return** True

8: **else**

1311 9: retu

1312

We will evaluate the range of  $Z_i$  and  $\mathbb{E}[Z_i]$  so that we can apply Hoeffding's inequality later. We have  $\widetilde{D}_j \geq D_j$  and  $\widetilde{D}_j < \lambda D_j$ . Hence,

$$Z_i = \frac{1}{p_j} \widehat{D}_j = \frac{\widetilde{D}}{\widetilde{D}_i} \widehat{D}_j \leq \frac{\widetilde{D}}{D_j} \lambda D_j = \lambda \widetilde{D}.$$

1320 On the other hand,  $Z_i$  must be non-negative by definition, so  $Z_i \in [0, \lambda \tilde{D}]$ .

1321 For the range of  $\mathbb{E}[Z]$ , we have

$$\mathbb{E}[Z_i] = \sum_{j=1}^s \frac{1}{p_j} \widehat{D}_j \cdot p_j = \sum_{j=1}^s \widehat{D}_j.$$

1326 Since  $\widehat{D}_j \in [D_j, \lambda D_j]$ , then we have  $\mathbb{E}[Z_i] \in [D, \lambda D]$ . Similarly by linearity of expectation, since  
 1327  $T = \sum_{i=1}^N Z_i$ , then we have  $\mathbb{E}[T] = N \cdot \mathbb{E}[Z_i] \in [ND, N\lambda D]$ . We perform casework on whether  
 1328  $\mu^2 D < \bar{D}$  or  $\mu^2 D > \bar{D}$ , corresponding to each of the two cases in the stated guarantee.

**Case 1:**  $\mu^2 D \leq \tilde{D}$ . We first analyze the case  $\mu^2 D \leq \tilde{D}$ . We define the event  $\mathcal{N}$  that L1SAMPLING( $\{\tilde{D}_j\}$ ,  $\{D_j\}$ ,  $\mu$ ,  $\delta$ ) returns False when  $\mu^2 D \leq \tilde{D}$  as the false negative event. Observe that this occurs in the algorithm if and only if  $\alpha T > N\tilde{D}$  when  $\mu^2 D \leq \tilde{D}$ .

1334 Notice that  $\alpha T > N\tilde{D}$  is impossible if  $|T - \mathbb{E}[T]| \leq \lambda ND$  since  $\mu^2 D \leq \tilde{D}$  and  $|T - \mathbb{E}[T]| \leq$   
 1335  $\lambda ND$  imply

$$\alpha T \leq \alpha(\mathbb{E}[T] + |T - \mathbb{E}[T]|) \leq \alpha(\lambda ND + \lambda ND) \leq \frac{2\alpha\lambda}{\mu^2} N \tilde{D} = \frac{1}{\mu} N \tilde{D} < N \tilde{D}.$$

1339 The first inequality is due to the triangle inequality. The second inequality comes from the fact  
 1340 that  $\mathbb{E}[T] \in [ND, N\lambda D]$  and  $|T - \mathbb{E}[T]| \leq \lambda ND$ . The third inequality is due to the condition  
 1341  $\mu^2 D < \tilde{D}$ . The equality is because of  $\alpha = 2\mu$  and  $\lambda = \frac{\mu}{2}$ . The last inequality is due to  $\mu > 1$ .

Hence under the condition that  $\mu^2 D \leq \tilde{D}$ , the false negative  $\mathcal{N}$  can only occur if  $|T - \mathbb{E}[T]| > \lambda ND$ . Therefore,

$$\Pr[\mathcal{N}] = \Pr\left[\alpha T > N\tilde{D} \mid \mu^2 D \leq \tilde{D}\right] \leq \Pr\left[|T - \mathbb{E}[T]| > \lambda ND \mid \mu^2 D \leq \tilde{D}\right].$$

<sup>1347</sup> Then by Hoeffding's inequality, c.f. Theorem B.6.

$$\Pr \left[ |T - \mathbb{E}[T]| > \lambda ND | \mu^2 D \leq \widetilde{D} \right] \leq 2 \exp \left( - \frac{(\lambda ND)^2}{\sum_{i=1}^N (\lambda D)^2} \right) = 2 \exp(-N) \leq \delta,$$

1350 because we have  $N_i \in [0, \lambda D]$  and  $N = \mathcal{O}(\log \frac{1}{\delta})$ . Since we have shown the probability of the  
 1351 event  $\mathcal{N}$  of getting a false negative is no more than  $\delta$ , then  $\text{L1SAMPLING}(\{\widetilde{D}_j\}, \{D_j\}, \mu, \delta)$  will  
 1352 return True when  $\mu^2 D \leq \widetilde{D}$  with probability at least  $1 - \delta$ .  
 1353

1354 We next analyze the case  $\mu D > \widetilde{D}$ .  
 1355

1356 **Case 2:  $\mu D > \widetilde{D}$ .** We define the event  $\mathcal{P}$  that  $\text{L1SAMPLING}(\{\widetilde{D}_j\}, \{D_j\}, \mu, \delta)$  returns True when  
 1357  $\mu D > \widetilde{D}$ , i.e., a false positive event. Observe that this can happen if and only if  $\alpha T \leq N \widetilde{D}$  when  
 1358  $\mu D > \widetilde{D}$ .  
 1359

1360 Notice that  $\alpha T \leq N \widetilde{D}$  is impossible if  $|T - \mathbb{E}[T]| \leq \frac{1}{2} ND$ . In fact, if  $\mu D > \widetilde{D}$  and  $|T - \mathbb{E}[T]| \leq$   
 1361  $\frac{1}{2} ND$ , then  
 1362

$$1363 \alpha T \geq \alpha(\mathbb{E}[T] - |T - \mathbb{E}[T]|) \geq \alpha(ND - \frac{1}{2} ND) = \frac{\alpha}{2} ND > \frac{\alpha}{2\mu} N \widetilde{D} = N \widetilde{D}.$$

1365 The first inequality is due to the triangle inequality. The second inequality comes from the fact that  
 1366  $\mathbb{E}[T] \in [ND, N\lambda D]$  and  $|T - \mathbb{E}[T]| \leq \frac{1}{2} ND$ . The third inequality is due to the condition  $\mu D > \widetilde{D}$ .  
 1367 The second equality is because of  $\alpha = 2\mu$ .  
 1368

1369 Hence under the condition that  $\mu D > \widetilde{D}$ , a false positive  $\mathcal{P}$  can only occur if  $|T - \mathbb{E}[T]| > \frac{1}{2} ND$ .  
 1370 Therefore,

$$1372 \Pr[\mathcal{P}] = \Pr\left[\alpha T \leq N \widetilde{D} \mid \mu D > \widetilde{D}\right] \leq \Pr\left[|T - \mathbb{E}[T]| > \frac{1}{2} ND \mid \mu D > \widetilde{D}\right].$$

1374 Then by Hoeffding's inequality, c.f., [Theorem B.6](#),

$$1376 \Pr\left[|T - \mathbb{E}[T]| > \frac{1}{2} ND \mid \mu D > \widetilde{D}\right] \leq 2 \exp\left(-\frac{(\frac{1}{2} ND)^2}{\sum_{i=1}^N (\lambda D)^2}\right) = 2 \exp\left(-\frac{N}{4\lambda^2}\right) \leq \delta,$$

1379 because  $N_i \in [0, \lambda D]$  and  $N = \mathcal{O}(\log \frac{1}{\delta})$ . Since we have shown the probability of the event  $\mathcal{P}$  is  
 1380 no more than  $\delta$ , then it follows that  $\text{L1SAMPLING}(\{\widetilde{D}_j\}, \{D_j\}, \mu, \delta)$  will return False if  $\mu D > \widetilde{D}$   
 1381 with probability at least  $1 - \delta$ .  $\square$   
 1382

### 1383 D.3 BICRITERIA APPROXIMATION WITH COMMUNICATION/ROUND REDUCTION

1385 In this section, we will introduce another protocol that uses  $\mathcal{O}(\log n \log k)$  rounds of communication  
 1386 and  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  bits of communication cost. The protocol will use fewer rounds of  
 1387 communication and total communication than [Algorithm 8](#) when  $k = \Omega(\log n)$ .  
 1388

1389 We first apply the  $L_1$  sampling subroutine  $\text{L1SAMPLING}$  to estimate  $D$ , which uses a low communica-  
 1390 tion cost. We then update  $\widetilde{D}$  if the estimation of  $D$  from the  $L_1$  sampling procedure has decreased  
 1391 significantly. Using such a strategy, we only need to update  $\widetilde{D}_j$  in very few rounds, saving the  
 1392 communication cost of updating  $\widetilde{D}_j$ .  
 1393

1394 However, the communication rounds are still  $\mathcal{O}(k)$ , which would be too expensive in some settings.  
 1395 To further decrease the rounds of communication, we use the batch sampling strategy. We double the  
 1396 number of points to be sampled in every round until the total cost  $D$  drops significantly.  $D$  can only  
 1397 decrease significantly for at most  $\mathcal{O}(\log n)$  rounds, so we can reduce the rounds of communication  
 1398 to  $\mathcal{O}(\log n \log k)$ .  
 1399

1400 Another issue that must be addressed is that we may sample some invalid points because the total  
 1401 cost  $D$  may decrease significantly during a round of batch sampling. A sampled point is valid only if  
 1402 it is sampled under the condition that  $\widetilde{D}$  is a  $\mathcal{O}(1)$ -approximation of  $D$ , so some sampled points may  
 1403 be invalid if  $D$  drops significantly during that sampling round. To ensure that we sample enough  
 1404 valid points, we need to count the number of valid samples during the sampling procedure. Rather  
 1405 than setting a fixed number  $N = \mathcal{O}(k)$  of points to sample, we count the number of valid samples  
 1406 and only terminate sampling after we get at least  $N$  valid samples. Fortunately, although we may

---

1404 sample more than  $N$  points in such a strategy, we can prove that the total number of points we  
1405 sample is still  $\mathcal{O}(k)$ .  
1406

1407 The algorithm appears in full in [Algorithm 10](#).

1408

---

1409 **Algorithm 10** Bicriteria approximation algorithm for  $(k, z)$ -clustering

1410

1411 **Input:** Dataset  $X_i$  for each site  $i \in [s]$

1412 **Output:** A set  $S$  that is an  $(\mathcal{O}(1), \tilde{\mathcal{O}}(1))$ -bicriteria approximation for the  $(k, z)$ -clustering

1413 1: Uniformly sample a point into  $S$  and compute constant-factor approximations  $\{\tilde{D}_j\}$  to  $D_j =$   
1414  $\text{cost}(X_j, S)$   
1415 2:  $N \leftarrow \mathcal{O}(k)$ ,  $M \leftarrow 0$ ,  $\mu \leftarrow 8$ ,  $\delta \leftarrow \mathcal{O}\left(\frac{1}{\log^2 n}\right)$   
1416 3: **while**  $M < N \setminus \setminus$  Sample roughly  $k$  points **do**  
1417 4:  $bool \leftarrow \text{False}$ ,  $i \leftarrow 1$   
1418 5: **while**  $bool = \text{False} \setminus \setminus$  Approximate distances  $\{\tilde{D}_j\}$  are accurate **do**  
1419 6: **for**  $l \leftarrow 1$  to  $2^i \setminus \setminus$  Attempt to sample points into  $S$  **do**  
1420 7:  $x_l \leftarrow \text{LAZYSAMPLING}(\{\text{cost}(x, S)\}, \{\tilde{D}_j\})$ , upload  $x_l$  on blackboard  
1421 8: **end for**  
1422 9:  $S \leftarrow S \cup (\cup_{l=1}^{2^i} \{x_l\})$ ,  $L \leftarrow \#\{x_l \neq \perp, 1 \leq l \leq 2^i\}$   $\triangleright$  Add successful samples to  $S$   
1423 10:  $M \leftarrow M + L$   
1424 11: Every site  $j$  computes  $D_j = \text{cost}(X_j, S)$   $\triangleright$  Update distances to closest center  
1425 12:  $bool \leftarrow \text{L1SAMPLING}(\{\tilde{D}_j\}, \{D_j\}, \mu, \delta)$   $\triangleright$  Check approximate distances  $\{\tilde{D}_j\}$   
1426 13: **if**  $bool = \text{False}$  **then**  
1427 14:  $i \leftarrow i + 1$   $\triangleright$  More aggressive number of samples to reduce number of rounds  
1428 15: **else**  
1429 16:  $r_j \leftarrow \text{POWERAPPROX}(D_j, 2)$ , every site  $j$  update  $r_j$  on the blackboard if  $r_j \neq \log_2 \tilde{D}_j$   
1430 17:  $\tilde{D}_j \leftarrow 2^{r_j}$   $\triangleright$  All sites update approximate distances  
1431 18: **end if**  
1432 19: **end while**  
20: **end while**  
21: **return**  $S$

---

1436

1437 [Algorithm 10](#) uses L1SAMPLING to detect whether  $D$  decreases significantly and needs to be up-  
1438 dated. Under the condition that  $\tilde{D}$  is a  $\mathcal{O}(1)$ -approximation of  $D$ , it uses batch sampling to sample  
1439 points, and verify whether  $\tilde{D}$  is still a  $\mathcal{O}(1)$ -approximation of  $D$  after batch sampling. If  $\tilde{D}$  is still a  
1440  $\mathcal{O}(1)$ -approximation of  $D$ , it means that all the sampled points are valid, and we count them. If  $\tilde{D}$   
1441 is no longer a  $\mathcal{O}(1)$ -approximation of  $D$ , it means the  $\mathcal{O}(1)$ -approximation condition breaks during  
1442 the batch sampling, which leads some sampled points invalid. However, the first sampled point  
1443 must be valid, since we sample it under the condition that  $\tilde{D}$  is an  $\mathcal{O}(1)$ -approximation. Hence,  
1444 we can add 1 to our valid sampled points count. We repeat the sampling until we have counted at  
1445 least  $N = \mathcal{O}(k)$  valid sampled points. Then the returned set  $S$  will have  $\mathcal{O}(k)$  points and is an  
1446  $\mathcal{O}(1)$ -approximation. Furthermore, by using L1SAMPLING and batch sampling, we can guarantee  
1447 a low round complexity and total communication for [Algorithm 10](#).

1448 We split the proof into three parts. First we will prove that  $S$  is an  $\mathcal{O}(1)$ -approximation and  $|S| =$   
1449  $\mathcal{O}(k)$ . Secondly, we will prove that the algorithm uses  $\mathcal{O}(\log n \log k)$  rounds of communication.  
1450 Lastly, we will prove that the algorithm uses  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  bits of communication with  
1451 probability at least 0.99.

1452 For the purposes of analysis, we define the concept of valid sample. Suppose  $x$  is a point sampled  
1453 by  $\text{LAZYSAMPLING}(\{\text{cost}(x, S)\}, \{\tilde{D}_i\})$ . Then we define  $x$  to be a valid sample if it is sampled  
1454 under the condition  $D \leq \tilde{D} < \gamma \cdot D_i$  for some constant  $\gamma$ .

1455 **Lemma D.3.** [Algorithm 10](#) will return  $S$  such that  $|S| = \mathcal{O}(k)$  and  $\text{cost}(S, X) \leq \mathcal{O}(1) \cdot$   
1456  $\text{cost}(C_{\text{OPT}}, X)$  with probability at least 0.98, where  $C_{\text{OPT}}$  is the optimal  $(k, z)$ -clustering of  $X$ .

---

1458 *Proof.* Suppose  $x$  is a point sampled by  $\text{LAZYSAMPLING}(\{\text{cost}(x, S)\}, \{\tilde{D}_i\})$ . If  $x$  is a valid  
1459 sample, it will reduce the size of  $\mathbf{Bad}_i$  with constant probability by [Lemma C.2](#). If we sample  
1460  $N = \mathcal{O}(k)$  valid points by our algorithm, by [Theorem B.10](#), we will get a constant approximation  
1461  $S$  with probability at least 0.99. We will show that our algorithm indeed samples at least  $N$  valid  
1462 samples for  $\gamma = \mu^2$ .

1463 According to the algorithm,  $\tilde{D}_j$  is always a 2-approximation of  $D_j$  that  $D_j \leq \tilde{D}_j < 2D_j$  after we  
1464 update  $\tilde{D}_j$ . Between two updates of  $\tilde{D}_j$ ,  $\tilde{D}_j$  is fixed, but  $D_j$  may only decrease. This means that  
1465  $D_j \leq \tilde{D}_j$  always holds in [Algorithm 10](#). Since  $D_j \leq \tilde{D}_j$  always holds,  $D \leq \tilde{D}$  always holds, too.

1466  
1467 If  $\tilde{D} < \mu^2 D$  after we sample  $L$  points, it means that all these  $L$  points are valid. We will either  
1468 add  $L$  to the counter  $M$  if  $\text{L1SAMPLING}$  returns False, or add 1 to the counter  $M$  if  $\text{L1SAMPLING}$   
1469 returns True. In either case, the number of valid points we sample is at least the number we add to  
1470 the counter  $M$ .

1471  
1472 If  $\tilde{D} \geq \mu^2 D$  after we sample  $L$  points, it means that some points we sample are invalid. How-  
1473 ever, the first point we sampled must be valid, as it is sampled under the condition  $\tilde{D} < \mu^2 D$ . If  
1474  $\text{L1SAMPLING}$  returns True, we will only add 1 to the counter  $M$ . Then the number of new valid  
1475 samples in this round is at least the number we add to the counter  $M$ .

1476 Therefore, the number of valid points we sample is at least the number we add to the counter  $M$   
1477 if  $\text{L1SAMPLING}$  returns True every time  $\tilde{D} \geq \mu^2 D$ . Since  $D = \text{poly}(n)$  when we have only one  
1478 point in  $S$ , and every time we update  $\tilde{D}$  in [Algorithm 10](#), it holds that  $D \leq \tilde{D} < 2D$ ,  $\tilde{D} \geq \mu^2 D$   
1479 can only occur at most  $\mathcal{O}(\log n)$  times. We know  $\text{L1SAMPLING}$  will return True if  $\tilde{D} \geq \mu^2 D$  with  
1480 probability at least  $1 - \delta$ . Since  $\delta = \mathcal{O}\left(\frac{1}{\log^2 n}\right)$ , then by a union bound,  $\text{L1SAMPLING}$  returns True  
1481 every time  $\tilde{D} \geq \mu^2 D$  with probability at least 0.99.

1482  
1483 Therefore, with probability at least 0.98, we will sample at least  $\mathcal{O}(k)$  valid points and they form an  
1484  $\mathcal{O}(1)$ -approximation for the optimal solution.

1485  
1486 **Upper bound on the size of  $S$ .** We will evaluate the number of sampled points that are not counted  
1487 as valid samples, and show the total number of such points is  $\mathcal{O}(k)$ . Suppose that we update  $\tilde{D}_j$  for  
1488  $m$  times for total.

1489  
1490 Let  $N_p$  be the total number of sampled points that are not counted as valid samples, and  $p_i$  as the  
1491 number of points that are not counted as valid samples between the  $(i - 1)$ -th and  $i$ -th update.  
1492 Then  $N_p = \sum_{i=1}^m p_i$ . Let  $N_q$  be the total number of sampled points that are counted as valid  
1493 samples, and  $q_i$  as the number of points that are counted as valid samples between the  $(i - 1)$ -th  
1494 and  $i$ -th update. Then  $N_q = \sum_{i=1}^m q_i$ . Since we will terminate the algorithm if we count for  
1495  $N = \mathcal{O}(k)$  valid samples,  $N_q = \mathcal{O}(k)$ . Since the number of points sampled before the  $i$ -th  
1496 update is  $p_i + 1$ , and the number of point sampled before these  $p_i + 1$  points is  $\frac{p_i + 1}{2}$ , therefore,  
1497  $p_i \leq 2q_i$ . Hence  $N_p \leq 2N_q = \mathcal{O}(k)$ . The points in  $S$  are either counted as valid samples or not, so  
1498  $|S| = N_p + N_q = \mathcal{O}(k)$ . □

1499  
1500 We next upper bound the round complexity of our algorithm.

1501  
1502 **Lemma D.4.** [Algorithm 10](#) uses  $\mathcal{O}(\log n \log k)$  rounds of communication with probability at least  
1503 0.99.

1504  
1505 *Proof.* We need  $\mathcal{O}(1)$  communication rounds in the initial stage [Algorithm 7](#). For the communica-  
1506 tion rounds in the remaining part of [Algorithm 10](#), we evaluate how many rounds of batch sampling  
1507 occurs between two update, and evaluate the times of update. Then the product is just the rounds of  
1508 total sampling.

1509  
1510 For the rounds of batch sampling that occurs between two update, it must be at most  $\mathcal{O}(\log k)$  rounds  
1511 since we double the number of points to be sampled for every other round if we do not update  $\tilde{D}$  and  
the total points to be sampled is  $\mathcal{O}(k)$ . For the times we update  $\tilde{D}$ , there are two cases: we update an

---

1512 ‘unnecessary update’ under the condition that  $\tilde{D} < \mu D$ , and we update a ‘necessary update’ under  
 1513 the condition that  $\tilde{D} \geq \mu D$ . By [Lemma D.2](#), L1SAMPLING will return True and we will have an  
 1514 ‘unnecessary update’ under the condition that  $\tilde{D} < \mu D$  with probability at most  $\delta$ . Since there are at  
 1515 most  $\mathcal{O}(\log k)$  successive batch sampling without an update, we will make an ‘unnecessary update’  
 1516 between two ‘necessary update’ with probability at most  $\mathcal{O}(\delta \log k)$ . For the ‘necessary update’,  
 1517 we will update  $\tilde{D}$  so that  $D \leq \tilde{D} < 2D$ . Since  $\mu = 8$ , it means every time we make a ‘necessary  
 1518 update’ under the condition that  $\tilde{D} \geq \mu D$ ,  $\tilde{D}$  decreases by at least a factor of 4. Hence, such update  
 1519 can occur at most  $\mathcal{O}(\log n)$  times.  
 1520

1521 Since the probability that we will make an ‘unnecessary update’ between two ‘necessary updates’ is  
 1522 at most  $\mathcal{O}(\delta \log k)$ , and the times of ‘necessary update’ is at most  $\mathcal{O}(\log n)$ , we will make an ‘un-  
 1523 necessary update’ in [Algorithm 10](#) with probability at most  $\mathcal{O}(\delta \log n \log k)$ . Since  $\delta = \mathcal{O}\left(\frac{1}{\log^2 n}\right)$ ,  
 1524 we will have no ‘unnecessary update’ in [Algorithm 10](#) with probability at least 0.99. Since ‘nec-  
 1525 essary update’ will occur at most  $\mathcal{O}(\log n)$  times, it means we will have at most  $\mathcal{O}(\log n)$  updates of  
 1526  $\tilde{D}$  with probability at least 0.99.  
 1527

1528 Therefore, with probability at least 0.99, [Algorithm 10](#) has  $\mathcal{O}(\log n \log k)$  rounds of batch sam-  
 1529 pling. Then we need to evaluate how many communication rounds we need for a round of batch  
 1530 sampling. For each round of batch sampling, the coordinator can send the request of  $2^i$  lazy sam-  
 1531 pling at the same time, and every site can respond afterwards. Hence there will be two rounds of  
 1532 communication for each iteration of sampling. To get the result returned by L1SAMPLING, the  
 1533  $\mathcal{O}(\log \frac{1}{\delta}) = \mathcal{O}(\log \log n)$  requests can be made simultaneously, so that there are  $\mathcal{O}(1)$  rounds of  
 1534 communication. Hence, the total rounds of communication in [Algorithm 10](#) is  $\mathcal{O}(\log n \log k)$ .  
 1535

1536 Finally, we upload  $r_j$  when we update  $\tilde{D}_j$ . All the sites can update the values of  $r_j$  in the same  
 1537 round. Since we will update at most  $\mathcal{O}(\log n)$  times, we needs  $\mathcal{O}(\log n)$  rounds of communication  
 1538 for total. Summing the rounds of communication across each part of our algorithm, it follows that  
 the total rounds of communication is  $\mathcal{O}(\log n \log k)$  with probability at least 0.99.  $\square$   
 1539

1540 Finally, we analyze the total communication of our algorithm.

1541 **Lemma D.5.** [Algorithm 10](#) uses  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  bits of communication with probability at  
 1542 least 0.99.  
 1543

1544 *Proof.* The [Algorithm 7](#) subroutine INITIALIZATION induces  $\mathcal{O}(d \log n + s \log \log n)$  bits of com-  
 1545 munication because we only write the sample point  $x$  and  $r_i$  on the blackboard, and  $r_i =$   
 1546  $\mathcal{O}(\log \log \text{cost}(X_i, S)) = \mathcal{O}(\log \log n)$ .  
 1547

1548 We will update  $\mathcal{O}(k)$  samples, which will use  $\mathcal{O}(kd \log n)$  total bits of communication.  
 1549

1550 Moreover, we will run L1SAMPLING at most  $\mathcal{O}(\log n \log k)$  times. For each time it runs, the  
 1551 coordinator will choose  $\mathcal{O}(\log \frac{1}{\delta}) = \mathcal{O}(\log \log n)$  sites. It uses  $\mathcal{O}(\log s)$  bits to represent a  
 1552 site on the blackboard, so that site knows it is chosen. Each chosen site needs  $\mathcal{O}(\log \log n)$   
 1553 bits to reply. Hence the total communication cost for running L1SAMPLING in [Algorithm 10](#) is  
 1554  $\mathcal{O}(\log n \log k(\log s + \log \log n))$  bits.  
 1555

1556 Updating the values  $\{r_j\}$  costs  $\mathcal{O}(s \log \log n)$  bits for each iteration. Since the value will be up-  
 1557 dated  $\mathcal{O}(\log n)$  times with probability at least 0.99, the total cost for updating the values  $\{r_j\}$  is  
 $\mathcal{O}(s \log n \log \log n)$  bits.  
 1558

1559 Therefore, by adding the communication cost for all parts of [Algorithm 10](#) and ignore all the poly-  
 1560 logarithm terms for  $s, k, \log n$ , the total communication is just  $\tilde{\mathcal{O}}(s \log n + kd \log n)$ .  $\square$   
 1561

1562 We have the following full guarantees for [Algorithm 10](#).

1563 **Lemma D.6.** [Algorithm 10](#) will return  $S$  such that  $|S| = \mathcal{O}(k)$  and  $\text{cost}(S, X) \leq \mathcal{O}(1) \cdot$   
 1564  $\text{cost}(C_{\text{OPT}}, X)$  with probability at least 0.98, where  $C_{\text{OPT}}$  is the optimal  $(k, z)$ -clustering of  $X$ . The  
 1565 algorithm uses  $\tilde{\mathcal{O}}(s \log n + kd \log n)$  bits of communication and  $\mathcal{O}(\log n \log k)$  rounds of commu-  
 1566 nication with probability at least 0.99.  
 1567

1566 *Proof.* The proof follows immediately from Lemma D.3, Lemma D.4, and Lemma D.5.  $\square$

#### 1568 D.4 $(1 + \varepsilon)$ -CORESET VIA SENSITIVITY SAMPLING IN THE BLACKBOARD MODEL

1570 To achieve a  $(1 + \varepsilon)$ -coreset for  $X$  with the  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation  $S$ , we use sensitivity sampling. The challenge lies in determining  $C_j$  and  $\text{cost}(C_j, S)$  to calculate  $\mu(x)$ , as communicating the exact values of  $C_j$  and  $\text{cost}(C_j, S)$  requires  $\mathcal{O}(sk \log n)$  bits, which is impractical for our purposes. However, a constant approximation of  $C_j$  and  $\text{cost}(C_j, S)$  suffices to approximate  $\mu(x)$  for the sensitivity sampling, and this can be achieved efficiently with minimal communication through Morris counters.

---

#### 1577 Algorithm 11 MORRIS( $r, n$ )

1578 **Input:** Initial count index  $r$ , and elements number  $n$   
 1579 **Output:** New count index  $m$   
 1580 1:  $m \leftarrow r$   
 1581 2: **for**  $i \leftarrow 1$  to  $n$  **do**  
 1582 3:   With probability  $\frac{1}{2^m}$ ,  $m \leftarrow m + 1$   
 1583 4: **end for**  
 1584 5: **return**  $r$

---

1585  
 1586 **Theorem D.7.** (Morris, 1978) Let  $X = \text{MORRIS}(0, n)$ . Then  $\mathbb{E}[2^X - 1] = n$ . Moreover, there  
 1587 exists a constant  $\gamma > 0$  such that if  $l \geq \frac{\gamma}{\varepsilon^2} \log \frac{1}{\delta}$  and  $Y = \frac{1}{l} \sum_{i=1}^l (2^{X_i} - 1)$ , where  $X_1, X_2, \dots, X_l$   
 1588 are  $l$  independent outputs of  $\text{MORRIS}(0, n)$ , then  
 1589

$$1590 \quad Y \in [n - \varepsilon n, n + \varepsilon n],$$

1591 with probability at least  $1 - \delta$ .  
 1592

1593 We can adapt MORRIS to a distributed version DISTMORRIS to reduce the communication cost.  
 1594

---

#### 1595 Algorithm 12 DISTMORRIS( $\varepsilon$ ):

1596  
 1597 **Input:** Precision parameter  $\varepsilon$ , every site  $S_i$  owns  $k$  numbers  $n_{i,j}$ ,  $j \in [k]$   
 1598 **Output:** Approximation  $(\tilde{N}_1, \tilde{N}_2, \dots, \tilde{N}_k)$  for sums  $N_j = \sum_{i=1}^s n_{i,j}$ ,  $j \in [k]$   
 1599 1:  $l \leftarrow \mathcal{O}(\frac{1}{\varepsilon^2} \log(100k))$   
 1600 2:  $m_{j,t} \leftarrow 0$  for all  $j \in [k]$  and  $t \in [l]$   
 1601 3: **for**  $i \leftarrow 1$  to  $s$  **do**  
 1602 4:    $m'_{j,t} \leftarrow \text{MORRIS}(m_{j,t}, n_{i,j})$  for all  $j \in [k]$  and  $t \in [l]$   
 1603 5:   **if**  $m'_{j,t} = m_{j,t}$  for all  $j \in [k]$  and  $t \in [l]$  **then**  
 1604 6:     Site  $i$  uploads  $\perp$  on blackboard  
 1605 7:   **else**  
 1606 8:     Site  $i$  uploads  $(m'_{j,t} - m_{j,t}, j, t)$  for all  $j \in [k]$  and  $t \in [l]$  that  $m'_{j,t} \neq m_{j,t}$   
 1607 9:   **end if**  
 1608 10:    $m_{j,t} \leftarrow m'_{j,t}$   
 1609 11: **end for**  
 1610 12:  $\tilde{N}_j \leftarrow \frac{1}{l} \sum_{t=1}^l (2^{m_{j,t}} - 1)$  for all  $j \in [k]$   
 1611 13: **return**  $(\tilde{N}_1, \tilde{N}_2, \dots, \tilde{N}_k)$

---

1612  
 1613 **Lemma D.8.** Let every site  $i$  own  $k$  numbers  $n_{i,j}$ , and  $|N_j| = \sum_{i=1}^s n_{i,j} = \text{poly}(n)$ . With prob-  
 1614 ability at least 0.99, DISTMORRIS( $\frac{1}{4}$ ) returns constant approximations  $\tilde{N}_j$  that  $\tilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$   
 1615 for all  $j \in [k]$  using  $\tilde{\mathcal{O}}(s + k \log n)$  bits of communication.  
 1616

1617 *Proof.* Although we split the process of MORRIS into a distributed version, its accuracy does not  
 1618 matter. This is because every step of MORRIS only needs to know the status of the previous step,  
 1619 and every time a site uses a step of MORRIS, it knows the status of the previous step either because  
 both of these steps occur at this site, or because the status of previous step is accurately uploaded by

1620 the previous site. Hence, the statement of the original MORRIS in [Theorem D.7](#) still holds for our  
1621 distributed version.

1622 By [Theorem D.7](#),  $\widetilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$  with probability at least  $1 - \frac{1}{100k}$  for any  $j \in [k]$  if we set  
1623  $l = \mathcal{O}(\log(100k))$ . Then, by a union bound,  $\widetilde{N}_j \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$  for all  $j \in [k]$  at the same time with  
1624 probability at least 0.99.

1625 To evaluate the total communication cost, we introduce  $M_i$  and  $M_{i,j}$  as follows to facilitate the  
1626 analysis. We set  $M_i$  as 0 if site  $i$  uploads  $\perp$  on the blackboard and  $M_i$  as the cost of communication  
1627 to update all  $(m'_{j,t} - m_{j,t}, j, t)$  for all  $j \in [k]$  and  $t \in [l]$  that  $m'_{j,t} \neq m_{j,t}$  otherwise. We set  $M_{i,j,t}$   
1628 to be the communication cost required to update  $(m'_{j,t} - m_{j,t}, j, t)$ .  
1629

1630 Every site with  $M_i = 0$  only needs  $\mathcal{O}(1)$  bits to upload  $\perp$ . Since there are  $s$  sites in total, the number  
1631 of sites with  $M_i = 0$  is at most  $s$ . Therefore, the total communication for these sites is  $\mathcal{O}(s)$ .  
1632

1633 For the sites with  $M_i \neq 0$ , the communication cost used is

$$\sum_{M_i \neq 0} M_i = \sum_{i=1}^s M_i = \sum_{j=1}^k \sum_{t=1}^l M_{i,j,t}.$$

1634 Note that the first equality holds because we only add more terms that all have value zero, whereas  
1635 the second equality is obtained by dividing  $M_i$  into  $M_{i,j,t}$ .  
1636

1637 Since  $\widetilde{N}_j = \frac{1}{l} \sum_{t=1}^l (2^{m_{j,t}} - 1) \in [\frac{3}{4}N_j, \frac{5}{4}N_j]$ , then  $2^{m_{j,t}}$  cannot be greater than  $2l \cdot N_j =$   
1638  $\mathcal{O}(\log k \text{poly}(n))$ . Therefore  $m_{j,t} \leq \mathcal{O}(\log n \log \log k)$ . Since  $M_{i,j,t}$  must be at least 1 if it is  
1639 nonzero, there are at most  $m_{j,t}$  nonzero  $M_{i,j,t}$  for given  $j, t$ . For every non-zero  $M_{i,j,t}$ , since  
1640 the site uploads  $m'_{j,t} - m_{j,t}$  which is at most the final  $m_{j,t}$ , then the site will use no more than  
1641  $\mathcal{O}(\log m_{j,t})$  bits to update  $m'_{j,t} - m_{j,t}$ , and  $\mathcal{O}(\log k + \log l)$  bits to express  $j$  and  $t$ . Thus,  
1642

$$\sum_{i=1} M_{i,j,t} = \sum_{i \in [s], M_{i,j,t} \neq 0} M_{i,j,t} \leq m_{j,t} \cdot \mathcal{O}(\log m_{j,t} + \log k + \log l).$$

1643 Since  $m_{j,t} \leq \mathcal{O}(\log n \log \log k)$  and  $l = \mathcal{O}(\log k)$ , then  $\sum_{i=1} M_{i,j,t} \leq \tilde{\mathcal{O}}(\log n)$ . Therefore,  
1644

$$\sum_{M_i \neq 0} M_i = \sum_{j=1}^k \sum_{t=1}^l M_{i,j,t} \leq kl \cdot \tilde{\mathcal{O}}(\log n) = \tilde{\mathcal{O}}(k \log n).$$

1645 By summing the communication cost of the sites that upload  $\perp$  and the communication cost of the  
1646 other sites, the total communication will be no more than  $\tilde{\mathcal{O}}(s + k \log n)$ .  $\square$   
1647

1648 Since  $|C_j| = \sum_{i=1}^s |C_j \cap X_i|$  and  $\text{cost}(C_j, S) = \sum_{i=1}^s \text{cost}(C_j \cap X_i, S)$  for any  $j \in [k]$ , we can use  
1649 DISTMORRIS to approximately evaluate these terms using low communication cost. Since both  $|C_j|$   
1650 and  $\text{cost}(C_j, S)$  are at most  $\text{poly}(n)$ , the total communication is at most  $\tilde{\mathcal{O}}(s + k \log n)$  bits. Every  
1651 site  $i$  can compute  $\text{cost}(x, S)$  for  $x \in X_i$  locally. Then with the constant-factor approximations for  
1652  $|C_j|$  and  $\text{cost}(C_j, S)$ , each site can evaluate a constant-factor approximation of the sensitivity  $\mu(x)$   
1653 locally.  
1654

1655 Finally, we give an algorithm producing a  $(1 + \varepsilon)$ -coreset for  $X$  with  $\mathcal{O}(\frac{dk}{\varepsilon^4}(\log k + \log \frac{1}{\varepsilon} \log \log n))$   
1656 bits of communication, given our  $\mathcal{O}(1)$ -approximation with  $\mathcal{O}(k)$  points and sensitivity  $\mu(x)$ .  
1657

1658 We now justify the correctness and complexity of the communication of [Algorithm 13](#).  
1659

1660 **Lemma D.9.** [Algorithm 13](#) returns  $A$  such that  $A$  is a  $(1 + \varepsilon)$ -coreset for  $X$  with probability at  
1661 least 0.98 if we already have an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation  $S$ . The algorithm uses  
1662  $\tilde{\mathcal{O}}(s + k \log n + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+\varepsilon})})$  bits of communication.  
1663

1664 *Proof.* By [Lemma D.8](#), we can get an  $\mathcal{O}(1)$ -approximation for  $|C_j|$  and  $\text{cost}(C_j, S)$  with probability  
1665 at least 0.99. Every site can also get the precise value of  $\text{cost}(x, S)$  locally. Since  
1666  $\mu(x) := \frac{1}{4} \cdot \left( \frac{1}{k|C_j|} + \frac{\text{cost}(x, S)}{k \text{cost}(C_j, S)} + \frac{\text{cost}(x, S)}{\text{cost}(X, S)} + \frac{\Delta_x}{\text{cost}(X, S)} \right)$  and  $\Delta_p := \text{cost}(C_j, S)/|C_j|$ , we can  
1667 evaluate an  $\mathcal{O}(1)$ -approximation  $\tilde{\mu}(x)$  of  $\mu(x)$ . Although [Bansal et al. \(2024\)](#) proves [Theorem B.7](#)

---

1674 **Algorithm 13**  $(1 + \varepsilon)$ -coreset for the blackboard model

1675

1676 **Input:** A constant approximation  $S$ ,  $|S| = \mathcal{O}(k)$

1677 **Output:** A  $(1 + \varepsilon)$ -coreset  $A$

1678 1: Use DISTMORRIS to get  $\mathcal{O}(1)$ -approximation for  $|C_j|$  and  $\text{cost}(C_j, S)$  for all  $j \in [k]$  on the blackboard

1679 2:  $m \leftarrow \tilde{\mathcal{O}}\left(\frac{k}{\varepsilon^2} \min\{\varepsilon^{-2}, \varepsilon^{-z}\}\right)$

1680 3: **for**  $i \leftarrow 1$  to  $s$  **do**

1681 4:    $A_i \leftarrow \emptyset$

1682 5:   Compute  $\tilde{\mu}(x)$  as an  $\mathcal{O}(1)$ -approximation of  $\mu(x)$  locally for all  $x \in X_i$

1683 6:   Upload  $\tilde{\mu}(X_i) = \sum_{x \in X_i} \tilde{\mu}(x)$

1684 7: **end for**

1685 8: Samples site  $i$  with probability  $\frac{\tilde{\mu}(X_i)}{\sum_{i=1}^s \tilde{\mu}(X_i)}$  independently for  $m$  times. Let  $m_i$  be the time site  $i$  are sampled. Write  $m_i$  on blackboard

1686

1687 9: **for**  $i \leftarrow 1$  to  $s$  **do**

1688 10:    $A_i \leftarrow \emptyset$

1689 11:   **for**  $j \in [m_i]$  **do**

1690 12:     Sample  $x$  with probability  $p_x = \frac{\tilde{\mu}(x)}{\tilde{\mu}(X_i)}$

1691 13:     **if**  $x$  is sampled **then**

1692 14:       Let  $x'$  be  $x$  efficiently encoded by  $S$  and accuracy  $\varepsilon' = \text{poly}(\varepsilon)$

1693 15:        $A \leftarrow A_i \cup \{(x', \frac{1}{m\tilde{\mu}(x)})\}$ , where  $\hat{\mu}(x)$  is a  $(1 + \frac{\varepsilon}{2})$ -approximation of  $\tilde{\mu}(x)$

1694 16:     **end if**

1695 17:   **end for**

1696 18:   Upload  $A_i$  to the blackboard

1697 19: **end for**

1698 20:  $A \leftarrow \cup_{i=1}^s A_i$

1699 21: **return**  $A$

---

1700

1701 in the setting that  $|S| = k$  and sample points with probability  $\mu(x)$ , they only need  $|S| = \mathcal{O}(k)$  and  $\frac{1}{\mu(x)} \leq \mathcal{O}(1) \cdot \max\{\frac{1}{k|C_j|}, \frac{\text{cost}(x, S)}{k \text{cost}(C_j, S)}, \frac{\text{cost}(x, S)}{\text{cost}(X, S)}, \frac{\Delta_x}{\text{cost}(X, S)}\}$  in their proof. Therefore, **Theorem B.7** is still valid under the condition that  $|S| = \mathcal{O}(k)$  and  $\tilde{\mu}(x) = \mathcal{O}(1) \cdot \mu(x)$ . Let  $B = \{(x, \frac{1}{m\tilde{\mu}(x)})\}$ , then  $B$  is an  $(1 + \frac{\varepsilon}{2})$ -coreset for  $X$  with probability at least 0.99 if we have  $\mathcal{O}(1)$ -approximation for  $|C_j|$  and  $\text{cost}(C_j, S)$ .

1702 We have  $A = \{(x', \frac{1}{m\tilde{\mu}(x)})\}$ . Since  $B$  is a  $(1 + \frac{\varepsilon}{2})$ -coreset for  $X$ , by **Lemma B.13**,  $A$  is a  $(1 + \varepsilon)$ -coreset for  $X$ . Therefore, **Algorithm 13** returns an  $(1 + \varepsilon)$ -coreset for  $X$  with probability at least 0.98.

1703 We need  $\mathcal{O}(s + k \log n)$  bits to get the  $\mathcal{O}(1)$ -approximation for  $|C_j|$  and  $\text{cost}(C_j, S)$ . We need  $\mathcal{O}(s \log m) = \tilde{\mathcal{O}}(s)$  bits to write  $m_i$  on the blackboard. We need  $\mathcal{O}(\log k + d \log \log n)$  bits to upload  $x'$ , because we need to point out which center in  $S$  is closest to  $x$ , which needs  $\mathcal{O}(\log k)$  bits, and we need to upload the power index, which needs  $\mathcal{O}(\log \frac{1}{\varepsilon} \log \log n)$  bits. We need  $\mathcal{O}(\frac{1}{\varepsilon} \log \log n)$  bits to upload  $\frac{1}{m\tilde{\mu}(x)}$  because we only need to upload  $\hat{\mu}(x)$ , which is a  $(1 + \frac{\varepsilon}{2})$ -approximation of  $\tilde{\mu}(x)$ . Since  $|A| = \mathcal{O}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$ , we need  $\tilde{\mathcal{O}}\left(s + k \log n + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  bits in total.  $\square$

1704

1705 We complete this subsection by claiming that we can get a  $(1 + \varepsilon)$ -strong coresset with communication cost no more than  $\tilde{\mathcal{O}}\left(s \log(n\Delta) + dk \log(n\Delta) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  bits.

1706 **Theorem D.10.** *There exists a protocol on  $n$  points distributed across  $s$  sites that produces a  $(1 + \varepsilon)$ -strong coresset for  $(k, z)$ -clustering with probability at least 0.97 that uses*

$$1707 \tilde{\mathcal{O}}\left(s \log(n) + dk \log(n) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$$

1708

1709 *total bits of communication in the blackboard model.*

---

1728 *Proof.* We can use [Algorithm 10](#) to generate an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation  $S$ , and then  
 1729 use [Algorithm 13](#) to get an  $(1 + \varepsilon)$ -strong coresset. By [Lemma D.6](#) and [Lemma D.9](#), the returned  
 1730 set  $A$  is a  $(1 + \varepsilon)$ -strong coresset with probability at least 0.97 and uses a communication cost of  
 1731  $\tilde{\mathcal{O}}\left(s \log(n) + dk \log(n) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$  bits for total.  $\square$   
 1732

1733  
 1734 **E COORDINATOR MODEL OF COMMUNICATION**  
 1735

1736 In this section, we discuss our distributed algorithms for the coordinator model of communication.  
 1737 Note that if all servers implement the distributed algorithms from the blackboard setting, the re-  
 1738 sulting communication would have  $\mathcal{O}(dsk \log n)$  terms simply from the  $\mathcal{O}(k)$  rounds of adaptive  
 1739 sampling across the  $s$  servers.  
 1740

1741  
 1742 **E.1 EFFICIENT SAMPLING IN THE COORDINATOR MODEL**  
 1743

1744 To avoid these  $\mathcal{O}(dsk \log n)$  terms in communication cost, we need a more communication ef-  
 1745 ficient method so that we can apply adaptive sampling and upload coresset with low communica-  
 1746 tion cost. We propose an algorithm called **EFFICIENTCOMMUNICATION**, which can send the lo-  
 1747 cation of a point with high accuracy and low cost. Suppose that a fixed site has a set of points  
 1748  $X = \{x_1, \dots, x_l\}$ , and another site has a point  $y$ . To simulate adaptive sampling while avoiding  
 1749 sending each point explicitly to all sites, our goal is to send a highly accurate approximate location  
 1750 of  $y$  to the first site with low cost. To that end, **EFFICIENTCOMMUNICATION** approximates and  
 1751 sends the coordinate of a point in every dimension. For the  $i$ -th dimension, we use a subroutine  
 1752 **HIGHPROBGREATERTHAN**, which can detect whether  $y^{(i)}$  is greater than  $x_j^{(i)}$  with high probabili-  
 1753 ty, where  $y^{(i)}$  is the coordinate of the  $i$ -th dimension of  $y$  and  $x_j^{(i)}$  is the coordinate of the  $i$ -th  
 1754 dimension of  $x_j \in X$ .  
 1755

---

1756 **Algorithm 14** **HIGHPROBGREATERTHAN** $(x, y, \delta)$   
 1757

1758 **Input:** Integer  $x, y$ , failure parameter  $\delta$   
 1759 **Output:** A boolean *bool* that shows whether  $x$  is greater than  $y$   
 1760 1:  $N \leftarrow \mathcal{O}(\log \frac{1}{\delta})$ ,  $r \leftarrow 0$   
 1761 2: **for**  $i \leftarrow 1$  to  $N$  **do**  
 1762 3:    $r_i \leftarrow 0$  if **GREATERTHAN** $(x, y)$  tells us  $x \leq y$ , and  $r_i \leftarrow 1$  otherwise  
 1763 4:    $r \leftarrow r + r_i$   
 1764 5: **end for**  
 1765 6: **if**  $\frac{r}{N} \leq \frac{1}{2}$  **then**  
 1766 7:   **return** False  
 1767 8: **else**  
 1768 9:   **return** True  
 10: **end if**  
 1769

---

1770 The subroutine **HIGHPROBGREATERTHAN** is an adaptation of **GREATERTHAN** by [\(Nisan, 1993\)](#).  
 1771 The main purpose of the subroutine is that by using a binary search, we can find the relative location  
 1772 of  $y^{(i)}$  to  $\{x_1^{(i)}, \dots, x_l^{(i)}\}$  within  $\log l$  comparisons. Then, comparing  $y^{(i)}$  with  $x_j^{(i)} + (1 + \varepsilon)^m$ , we  
 1773 can find the best  $x_j^{(i)}$  and  $m$  to approximate  $y^{(i)}$ . Our algorithm for the coordinator model appears  
 1774 in full in [Algorithm 15](#).  
 1775

1776 We first show correctness of **HIGHPROBGREATERTHAN**, running multiple times and taking the  
 1777 majority vote if necessary to boost the probability of correctness.  
 1778

1779 **Lemma E.1.** *If **HIGHPROBGREATERTHAN** $(x, y, \delta)$  returns False, then  $x \leq y$  with probability at  
 1780 least  $1 - \delta$ . If **HIGHPROBGREATERTHAN** $(x, y, \delta)$  returns True, then  $x > y$  with probability at  
 1781 least  $1 - \delta$ . Furthermore, the protocol uses  $\mathcal{O}(\log \log n \log \frac{1}{\delta})$  bits of communication provided that  
 $x, y \in [-\text{poly}(n), \text{poly}(n)]$ .*

---

1782 **Algorithm 15** EFFICIENTCOMMUNICATION( $X, y, \varepsilon, \delta$ )  
1783  
1784 **Input:** A set  $X = \{x_1, \dots, x_l\}$  owned by one site; a point  $y$  owned by another site;  $\varepsilon$ , accuracy  
1785 parameter;  $\delta$ , the failure probability  
1786 **Output:** The second site sends  $\tilde{y}$  to the first site, which will be an approximate location of  $y$  that  
1787  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  for any  $x \in X$ , with probability at least  $1 - \delta$   
1788 1: **for**  $i \leftarrow 1$  to  $d$  **do**  
1789 2: Sort  $X = \{x_{i_1}, \dots, x_{i_l}\}$  such that  $x_{i_1}^{(i)} \leq \dots \leq x_{i_l}^{(i)}$ , where  $x_{i_j}^{(i)}$  is the  $i$ -th coordinate of  $x_{i_j}$   
1790 3:  $x_{i_0}^{(i)} \leftarrow -\Delta, x_{i_{l+1}}^{(i)} \leftarrow \Delta, \delta' \leftarrow \mathcal{O}\left(\frac{\delta}{d(\log l + \log \log n + \log \frac{1}{\varepsilon})}\right)$   
1791 4: Use binary search and HIGHPROBGREATERTHAN( $y^{(i)}, x_{i_j}^{(i)}, \delta'$ ) to find  $x_{i_s}^{(i)}$  that  $|x_{i_s}^{(i)} - y^{(i)}| \leq |x_{i_j}^{(i)} - y^{(i)}|$  for any  $i_j \in \{0, 1, 2, \dots, l, l+1\}$   
1792 5: Use HIGHPROBGREATERTHAN test whether  $y^{(i)} = x_{i_s}^{(i)}$   
1793 6: **if**  $y^{(i)} = x_{i_s}^{(i)}$  **then**  
1794 7:  $\Delta y^{(i)} \leftarrow 0$   
1795 8: **else**  
1796 9:  $\gamma \leftarrow \text{sign}(y^{(i)} - x_{i_s}^{(i)})$   
1797 10: Use binary search and HIGHPROBGREATERTHAN( $y^{(i)}, x_{i_s}^{(i)} + \gamma \cdot (1 + \varepsilon)^t, \delta'$ ) to find  $m$   
1798 that  $|x_{i_s}^{(i)} + \gamma \cdot (1 + \varepsilon)^m - y^{(i)}| \leq |x_{i_s}^{(i)} + \gamma \cdot (1 + \varepsilon)^t - y^{(i)}|$  for any  $t \in \mathbb{N}$   
1799 11:  $\Delta y^{(i)} \leftarrow \gamma \cdot (1 + \varepsilon)^m$   
1800 12: **end if**  
1801 13:  $\tilde{y}^{(i)} \leftarrow x_{i_s}^{(i)} + \Delta y^{(i)}$   
1802 14: **end forreturn**  $\tilde{y} = (\tilde{y}^{(1)}, \tilde{y}^{(2)}, \dots, \tilde{y}^{(d)})$

---

1803  
1804  
1805  
1806  
1807  
1808 *Proof.* We define random variable  $E_i = r_i$ . By Theorem B.8, GREATERTHAN will give a wrong  
1809 answer with probability at most  $p < \frac{1}{2}$ , so  
1810

$$\begin{aligned}\mathbb{E}[E_i|x \leq y] &= 0 \cdot \Pr[E_i = 0|x \leq y] + 1 \cdot \Pr[E_i = 1|x \leq y] \leq p, \\ \mathbb{E}[E_i|x > y] &= 0 \cdot \Pr[E_i = 0|x > y] + 1 \cdot \Pr[E_i = 1|x > y] \geq 1 - p.\end{aligned}$$

1811 Since  $E_i \in [0, 1]$ , by Hoeffding's inequality, c.f., Theorem B.6,  
1812

$$\begin{aligned}\Pr\left[\sum_{i=1}^N E_i > \frac{N}{2} \middle| x \leq y\right] &= \Pr\left[\sum_{i=1}^N E_i > Np + \frac{N}{2} - Np \middle| x \leq y\right] \\ &\leq \Pr\left[\left|\sum_{i=1}^N E_i - N\mathbb{E}[E_i]\right| > \frac{N}{2} - Np \middle| x \leq y\right] \\ &\leq 2 \exp\left(-\frac{\left(\frac{N}{2} - Np\right)^2}{N \cdot 1^2}\right) \\ &= \delta,\end{aligned}$$

1813 and  
1814

$$\begin{aligned}\Pr\left[\sum_{i=1}^N E_i \leq \frac{N}{2} \middle| x > y\right] &= \Pr\left[\sum_{i=1}^N E_i \leq N(1-p) + \frac{N}{2} - N(1-p) \middle| x > y\right] \\ &\leq \Pr\left[\left|\sum_{i=1}^N E_i - N \cdot \mathbb{E}[E_i]\right| > \frac{N}{2} - Np \middle| x > y\right] \\ &\leq 2 \exp\left(-\frac{\left(\frac{N}{2} - Np\right)^2}{N \cdot 1^2}\right) \\ &= \delta.\end{aligned}$$

1815 Hence HIGHPROBGREATERTHAN( $x, y, \delta$ ) is correct with probability at least  $1 - \delta$ .  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835

---

1836 For the communication cost, since GREATERTHAN uses  $\mathcal{O}(\log \log n)$  bits of communication, pro-  
 1837 vided that  $x, y \in [-\text{poly}(n), \text{poly}(n)]$ , and we run GREATERTHAN for  $N = \mathcal{O}(\log \frac{1}{\delta})$  times, then  
 1838 the total communication cost is  $\mathcal{O}(\log \log n \log \frac{1}{\delta})$  bits in total.  $\square$   
 1839

1840 EFFICIENTCOMMUNICATION( $X, y, \varepsilon, \delta$ ) in [Algorithm 15](#) will send  $\tilde{y}$ , a good approximation loca-  
 1841 tion of  $y$  with probability at least  $1 - \delta$ . In addition, it only uses  $d \log l \text{polylog}(\log n, \log l, \frac{1}{\varepsilon}, \frac{1}{\delta})$   
 1842 bits. To formally prove these guarantees, we first show  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  with probability  $1 - \delta$   
 1843 and then upper bound the total communication of the protocol.

1844 **Lemma E.2.** EFFICIENTCOMMUNICATION( $X, y, \varepsilon, \delta$ ) will send  $\tilde{y}$  to the first site such that  $\|y -$   
 1845  $\tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  with probability at least  $1 - \delta$ .  
 1846

1847 *Proof.* We condition on the correctness of HIGHPROBGREATERTHAN. We will prove that  $\|y -$   
 1848  $\tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$ . First, we will prove  $|y^{(i)} - \tilde{y}^{(i)}| \leq \varepsilon |x_j^{(i)} - y^{(i)}|$  for any  $j \in [d]$  and  $x_j \in X$ .  
 1849

1850 Let  $X^{(i)} = \{-\Delta, x_1^{(i)}, x_2^{(i)}, \dots, x_l^{(i)}, \Delta\}$ . Assume  $x_s^{(i)} \in X^{(i)}$  that  $|x_s^{(i)} - y^{(i)}| \leq |x_j^{(i)} - y^{(i)}|$  for  
 1851 any  $x_j^{(i)} \in X^{(i)}$ . If  $y^{(i)} = x_s^{(i)}$ ,  $\tilde{y}^{(i)}$  is just  $y^{(i)}$ , which means  $|y^{(i)} - \tilde{y}^{(i)}| = |x_s^{(i)} - y^{(i)}| = 0$ .  
 1852

1853 If  $y^{(i)} \neq x_s^{(i)}$ , we have  $\tilde{y}^{(i)} = x_s^{(i)} + \Delta y^{(i)}$  and  $\Delta y^{(i)} = \gamma \cdot (1 + \varepsilon)^m$ , where  $\gamma = \text{sign}(y^{(i)} - x_s^{(i)})$ .  
 1854 Assume  $y^{(i)} - x_s^{(i)} = \gamma \cdot (1 + \varepsilon)^{m+m'}$ . Then  
 1855

$$|y^{(i)} - \tilde{y}^{(i)}| = |(y^{(i)} - x_s^{(i)}) - (\tilde{y}^{(i)} - x_s^{(i)})| = |(1 + \varepsilon)^{m+m'} - (1 + \varepsilon)^m|.$$

1856 Since  $|x_s^{(i)} + \gamma \cdot (1 + \varepsilon)^m - y^{(i)}| \leq |x_s^{(i)} + \gamma \cdot (1 + \varepsilon)^t - y^{(i)}|$  for any  $t \in \mathbb{N}$ ,  $|m'|$  must be less than  
 1857 1. Hence  
 1858

$$|y^{(i)} - \tilde{y}^{(i)}| = (1 + \varepsilon)^{m+m'} \cdot |1 - (1 + \varepsilon)^{-m'}| \leq \varepsilon (1 + \varepsilon)^{m+m'} = \varepsilon |x_s^{(i)} - y^{(i)}|.$$

1861 Since  $|x_s^{(i)} - y^{(i)}| \leq |x_j^{(i)} - y^{(i)}|$  for any  $j$ , and  $|y^{(i)} - \tilde{y}^{(i)}| \leq \varepsilon |x_s^{(i)} - y^{(i)}|$ , thus  $|y^{(i)} - \tilde{y}^{(i)}| \leq$   
 1862  $\varepsilon |x_j^{(i)} - y^{(i)}|$ . Therefore, for any  $x_j \in X$ ,  
 1863

$$\|y - \tilde{y}\|_2^2 = \sum_{i=1}^d |y^{(i)} - \tilde{y}^{(i)}|^2 \leq \sum_{i=1}^d \varepsilon^2 |x_j^{(i)} - y^{(i)}|^2 = \varepsilon^2 \cdot \|x_j - y\|_2.$$

1864 Hence  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  for any  $x \in X$ .  
 1865

1866 **Analysis of the failure probability.** It remains to upper bound the failure probability by counting  
 1867 how many times HIGHPROBGREATERTHAN( $\cdot, \cdot, \delta'$ ) is run in the algorithm. For every dimension  
 1868  $i$ , we run HIGHPROBGREATERTHAN( $\cdot, \cdot, \delta'$ ) to find the closest  $x_{i_s}^{(i)}$  to  $y^{(i)}$  and  $m$  to approximate  
 1869  $y^{(i)} - x_{i_s}^{(i)}$ . To find the closest  $x_{i_s}^{(i)}$  to  $y^{(i)}$ , we can use binary search to find  $x_{i_p}^{(i)} \leq y^{(i)} \leq x_{i_{p+1}}^{(i)}$ , and  
 1870 then compare their midpoint  $\frac{x_{i_p}^{(i)} + x_{i_{p+1}}^{(i)}}{2}$  with  $y^{(i)}$  to determine which one is closer to  $y^{(i)}$ . Since there  
 1871 are  $l + 2$  elements in  $\{x_{i_0}^{(i)}, x_{i_1}^{(i)}, \dots, x_{i_l}^{(i)}, x_{i_{l+1}}^{(i)}\}$ , we need to run HIGHPROBGREATERTHAN( $\cdot, \cdot, \delta'$ )  
 1872 for  $\log(l + 2) + 1$  times in this stage.  
 1873

1874 To find  $m$  that  $|x_{i_s}^{(i)} + \gamma \cdot (1 + \varepsilon)^m - y^{(i)}| \leq |x_{i_s}^{(i)} + \gamma \cdot (1 + \varepsilon)^t - y^{(i)}|$  for any  $t \in \mathbb{N}$ , we can use binary  
 1875 search to find  $(1 + \varepsilon)^q \leq |y^{(i)} - x_{i_s}^{(i)}| \leq (1 + \varepsilon)^{q+1}$ , and then compare their midpoint with  $|y^{(i)} - x_{i_s}^{(i)}|$   
 1876 to determine which one is closer to  $|y^{(i)} - x_{i_s}^{(i)}|$ . Since we already have  $x_{i_p}^{(i)} \leq y^{(i)} \leq x_{i_{p+1}}^{(i)}$  and  
 1877  $|x_{i_j}^{(i)}| \leq \Delta$  for all  $j \in \{0, 1, \dots, l + 1\}$ , we only need to search  $q$  among  $\{0, 1, \dots, \log_{1+\varepsilon} \Delta\}$ .  
 1878 Since  $\Delta = \text{poly}(n)$ , we can use binary search to find  $q$  in at most  $\mathcal{O}(\log \frac{\log n}{\varepsilon})$  rounds. Thus  
 1879 we need to run HIGHPROBGREATERTHAN( $\cdot, \cdot, \delta'$ ) for  $\mathcal{O}(\log \log n + \log \frac{1}{\varepsilon})$  times in this stage.  
 1880 Therefore, we will apply HIGHPROBGREATERTHAN for at most  $\mathcal{O}(d(\log l + \log \log n + \log \frac{1}{\varepsilon}))$   
 1881 times. Since HIGHPROBGREATERTHAN( $\cdot, \cdot, \delta'$ ) will return correct result with failure probability  
 1882 at most  $\delta'$  and  $\delta' = \mathcal{O}\left(\frac{\delta}{d(\log l + \log \log n + \log \frac{1}{\varepsilon})}\right)$ , EFFICIENTCOMMUNICATION( $X, y, \varepsilon, \delta$ ) has a  
 1883 failure probability at most  $\delta$ .  $\square$   
 1884

1890 Next, we analyze the communication complexity of our algorithm.

1891  
1892 **Lemma E.3.**  $\text{EFFICIENTCOMMUNICATION}(X, y, \varepsilon, \delta)$  uses  $d \log \ell \text{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits  
1893 of communication for total, where  $\ell = |X|$  is the number of points owned by the first site.

1894  
1895 *Proof.* We need to run  $\text{HIGHPROBGREATERTHAN}(\cdot, \cdot, \delta')$  for at most  
1896  $\mathcal{O}(d(\log \ell + \log \log n + \log \frac{1}{\varepsilon}))$  times. Since  $\text{HIGHPROBGREATERTHAN}(\cdot, \cdot, \delta')$  cost  
1897  $\mathcal{O}(\log \log n \log \frac{1}{\delta'})$  bits for a single running, it takes  $d \log \ell \text{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits to  
1898 run  $\text{HIGHPROBGREATERTHAN}(\cdot, \cdot, \delta')$  for total.

1899  
1900 We also need communication to send  $\tilde{y} = (\tilde{y}^{(1)}, \tilde{y}^{(2)}, \dots, \tilde{y}^{(d)})$  to the first site. However, we  
1901 only need to send  $x_{i_s}^{(i)}$  and  $\Delta y^{(i)}$  to the first site. Since the first site already has the location of  
1902 all  $x \in X$ , we only need to send  $i_s$  to identify  $x_{i_s}^{(i)}$ , which takes  $\mathcal{O}(\log \ell)$  bits. Since  $\Delta y^{(i)} =$   
1903  $\gamma \cdot (1 + \varepsilon)^m$ , we only need to send  $\gamma$  and  $m$ . Sending  $\gamma$  requires  $\mathcal{O}(1)$  bits since  $\gamma \in \{-1, 0, 1\}$ .  
1904 Sending  $m$  requires  $\mathcal{O}(\log \log n + \log \frac{1}{\varepsilon})$  bits since  $(1 + \varepsilon)^m = \text{poly}(n)$ . Therefore, it takes at  
1905 most  $d \log \ell \text{polylog}(\log n, \frac{1}{\varepsilon})$  bits to send  $\tilde{y} = (\tilde{y}^{(1)}, \tilde{y}^{(2)}, \dots, \tilde{y}^{(d)})$ .

1906 Hence [Algorithm 15](#) takes at most  $d \log \ell \text{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$  to send  $\tilde{y}$  to the first site.  $\square$

1908 Putting together [Lemma E.2](#) and [Lemma E.3](#), we have:

1910  
1911 **Lemma E.4.**  $\text{EFFICIENTCOMMUNICATION}(X, y, \varepsilon, \delta)$  will send  $\tilde{y}$  to the first site such that  
1912  $\|y - \tilde{y}\|_2 \leq \varepsilon \|x - y\|_2$  with probability at least  $1 - \delta$ . Furthermore, it only uses  
1913  $d \log \ell \text{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits of communication for total, where  $\ell = |X|$  is the number  
1914 of points owned by the first site.

## 1915 E.2 $(1 + \varepsilon)$ -CORESET VIA SENSITIVITY SAMPLING IN THE COORDINATOR MODEL

1916 Given the analysis in [Appendix E.1](#), it follows that we can effectively perform adaptive sampling to  
1917 achieve a bicriteria approximation at the coordinator. It remains to produce a  $(1 + \varepsilon)$ -coreset, for  
1918 which we again use sensitivity sampling.

1919  
1920  $\text{EFFICIENTCOMMUNICATION}(X, y, \varepsilon, \delta)$  can send the location of  $y$  using low communication cost.  
1921 However, its communication cost is  $d \log \ell \text{polylog}(\log n, \log \ell, \frac{1}{\varepsilon}, \frac{1}{\delta})$ , where  $\ell = |X|$ . For the  
1922 coordinator model, suppose that each site  $i \in [s]$  has a dataset  $X_i$ . Since  $|X_i| = \mathcal{O}(n)$ , and  
1923 the coordinator needs to send the approximate location of all  $\mathcal{O}(k)$  samples to each site to apply  
1924 adaptive sampling, which would still require  $dsk \log n \text{polylog}(\log n, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits to send the location  
1925 of samples using  $\text{EFFICIENTCOMMUNICATION}(X_i, y, \varepsilon, \delta)$ . Fortunately, we can generate a  $(1 + \frac{\varepsilon}{2})$ -  
1926 coresset  $P_i$  for every  $X_i$ , which has a size of  $\tilde{\mathcal{O}}\left(\frac{k}{\min\{\varepsilon^4, \varepsilon^{2+z}\}}\right)$ . Since adaptive sampling also works  
1927 for the weighted case, it is enough to generate an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation for the  
1928 weighted coresset  $P = \bigcup_{i \in [s]} P_i$ . Therefore, only  $dsk \text{polylog}(\log n, k, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits are necessary to  
1929 send the location of the samples to each site.

1930 To further eliminate the multiple dependency of  $d$ , we notice that only the distance between the data  
1931 point  $x \in P_i$  and the center generated by adaptive sampling  $s \in S$  is necessary to apply adaptive  
1932 sampling and sensitivity sampling. Hence, we can use the Johnson-Lindenstrauss transformation to  
1933 map  $P$  to  $\pi(P) \subset \mathbb{R}^{d'}$ , where  $d' = \mathcal{O}(\log(sk))$ . As a result of the JL transformation,  $\pi(P)$  is  
1934 located in a lower-dimensional space, but the pairwise distance is still preserved by the mapping.  
1935 Therefore, we can further reduce the communication cost needed to send the location of the samples  
1936 to each site, which is now  $sk \text{polylog}(\log n, s, k, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits. Hence, we can apply adaptive sampling  
1937 and send the exact location of the sampled point  $s$  to the coordinator. The coordinator can then  
1938 use  $\text{EFFICIENTCOMMUNICATION}(\pi(P_i), \pi(s), \varepsilon, \delta)$  to send the approximate location of  $\pi(s)$  to  
1939 every site, which is accurate enough for every site to update a constant approximation for the cost  
1940 of points. Thus, we can repeat adaptive sampling using a low cost of communication and get an  
1941  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation  $S$  for the optimal  $(k, z)$ -clustering.

1942 Since every site has an approximate copy of  $\pi(S)$ , they can send a constant approximation of  $|C_j \cap$   
1943  $P_i|$  and  $\text{cost}(C_j \cap P_i, S)$  to the coordinator. However, the site may assign a point to a center in  $S$  that  
1944 is not nearest to it. This is because the site only has an approximate location of  $\pi(S)$ , and therefore

---

1944 can assign  $x$  to another center  $s'$  if  $\text{cost}(x, s')$  is very close to  $\text{cost}(x, s)$ , where  $s$  is the center closest  
 1945 to  $x$ . Fortunately, the proof of [Theorem B.7](#) does not require that all points  $x$  be assigned to its  
 1946 closet point. [Theorem B.7](#) is still valid if  $\text{cost}(x, s') \leq \mathcal{O}(1) \cdot \text{cost}(x, s)$ . Hence, we can generate a  
 1947  $(1 + \frac{\varepsilon}{4})$ -coreset for  $P$  by sensitivity sampling.

1948 After applying sensitivity sampling to sample the points, each site can send the sampled points to  
 1949 the coordinator using [EFFICIENTCOMMUNICATION](#)( $S, x, \varepsilon', \delta$ ). By a similar discussion of efficient  
 1950 encoding in [Lemma B.13](#), we can prove that the sampled points form a  $(1 + \frac{\varepsilon}{2})$ -coreset  $A'$  for  
 1951  $P = \cup_{i \in [s]} P_i$ . Therefore,  $A'$  would be a  $(1 + \varepsilon)$ -coreset for  $X = \cup_{i \in [s]} X_i$ , and the coordinator  
 1952 can solve the  $(k, z)$ -clustering based on the coresets  $A'$ . Since  $|P_i| = \tilde{\mathcal{O}}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$ , we can  
 1953 run [EFFICIENTCOMMUNICATION](#)( $P_i, y, \varepsilon, \delta$ ) using  $\text{polylog}(\log n, k, \frac{1}{\varepsilon}, \frac{1}{\delta})$  bits of communication  
 1954 to send each sampled point. We give the algorithm in full in [Algorithm 16](#).

1956

---

1957 **Algorithm 16**  $(1 + \varepsilon)$ -coreset for the coordinator model

1958 **Input:** The dataset  $X_i$  every site  $i$  owns,  $i \in [s]$   
 1959 **Output:** A  $(1 + \varepsilon)$ -coreset  $A'$  sent to the coordinator

1960 1: Every site  $i$  generates a  $(1 + \frac{\varepsilon}{2})$ -coreset  $P_i$  of  $X_i$   
 1961 2: The coordinator send random seed to every site that generate Johnson-Lindenstrauss  $\pi$ , such  
 1962 that  $\|\pi(x) - \pi(y)\|_2 \in [\frac{1}{2}\|x - y\|_2, \frac{3}{2}\|x - y\|_2]$  for any  $x, y \in P = \cup_{i \in [s]} P_i$  and  $\pi(x) \in R^{d'}$   
 1963 where  $d' = \mathcal{O}(\log(\text{sk}))$   
 1964 3:  $S \leftarrow \{s_0\}$ , where  $s_0$  is a point sampled by the coordinator  
 1965 4:  $S_j \leftarrow \emptyset$  for  $j \in [s]$   
 1966 5:  $N \leftarrow \mathcal{O}(k), \varepsilon' \leftarrow \mathcal{O}(\varepsilon^z), m \leftarrow \tilde{\mathcal{O}}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right), \delta \leftarrow \mathcal{O}\left(\frac{1}{sk+m}\right)$   
 1967 6: **for**  $i \leftarrow 1$  to  $N$  **do**  
 1968 7: Use [EFFICIENTCOMMUNICATION](#)( $\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta$ ) to send  $\tilde{s}_{i-1}^{(j)}$ , an approximation of  
 1969  $\pi(s_{i-1})$  to every site  $j$ .  $S_j \leftarrow S_j \cup \{\tilde{s}_{i-1}^{(j)}\}$   
 1970 8: Every site updates the cost  $\text{cost}(\pi(P_j), S_j)$  and sends  $\tilde{D}_j$ , a constant approximation of  
 1971  $\text{cost}(\pi(P_j), S_j)$  to the coordinator  
 1972 9:  $s_i \leftarrow \text{LAZYSAMPLING}(\{\text{cost}(\pi(x), S_j)\}, \{\tilde{D}_j\})$   
 1973 10: **end for**  
 1974 11: Every site  $j$  computes  $|C_l^{(j)}|$  and  $\text{cost}(\pi(C_l^{(j)}), S_j)$  for  $l \leq |S|$ , where  $C_l^{(j)} = \{x \in P_j : \text{dist}(\pi(x), \tilde{s}_l^{(j)}) \leq \text{dist}(\pi(x), \tilde{s}_p^{(j)}), \forall p \leq |S|\}$   
 1975 12: Every site  $j$  sends a constant approximation of  $|C_l^{(j)}|$  and  $\text{cost}(\pi(C_l^{(j)}), S_j)$  to the coordinator  
 1976 13: The coordinator computes  $\sum_{l \leq |S|} |C_l^{(j)}|$  and  $\sum_{l \leq |S|} \text{cost}(\pi(C_l^{(j)}), S_j)$ , and sends constant  
 1977 approximation of them to every site  
 1978 14: Every site computes  $\tilde{\mu}(x)$  as an  $\mathcal{O}(1)$ -approximation of  $\mu(x)$  locally for all  $x \in P_j$ , and send  
 1979  $\tilde{\mu}(P_i)$ , a constant approximation of  $\tilde{\mu}(P_i) = \sum_{x \in P_i} \tilde{\mu}(x)$  to the coordinator  
 1980 15: The coordinator samples site  $j$  with probability  $\frac{\tilde{\mu}(P_j)}{\sum_{i=1}^s \tilde{\mu}(P_i)}$  independently for  $m$  times. Let  $m_j$   
 1981 be the time site  $j$  are sampled. Sends  $m_j$  to site  $j$   
 1982 16: **for**  $i \leftarrow 1$  to  $s$  **do**  
 1983 17:  $A_i \leftarrow \emptyset, A'_i \leftarrow \emptyset$   
 1984 18: **for**  $j \in [m_i]$  **do**  
 1985 19: Sample  $x$  with probability  $p_x = \frac{\tilde{\mu}(x)}{\tilde{\mu}(P_i)}$   
 1986 20: **if**  $x$  is sampled **then**  
 1987 21:  $A_i \leftarrow A_i \cup \{x, \frac{1}{m_j \tilde{\mu}(x)}\}, \tilde{x} \leftarrow \text{EFFICIENTCOMMUNICATION}(S, x, \varepsilon', \delta), A'_i \leftarrow A'_i \cup$   
 1988  $\{\tilde{x}, \frac{1}{m_j \tilde{\mu}(x)}\}$ , where  $\tilde{\mu}(x)$  is a  $(1 + \frac{\varepsilon}{2})$ -approximation of  $\tilde{\mu}(x)$   
 1989 22: **end if**  
 1990 23: **end for**  
 1991 24: **end for**  
 1992 25:  $A' \leftarrow \cup_{i=1}^s A'_i$   
 1993 26: **return**  $A'$

---

1998 We now show that [Algorithm 16](#) will return a  $(1 + \varepsilon)$ -coreset of  $X$  with constant probability and  
1999 uses low communication cost. We will first show that  $A'$  is a  $(1 + \varepsilon)$ -coreset of  $X$ .

2000 **Lemma E.5.** *Algorithm 16 returns a  $(1 + \varepsilon)$  coreset of  $X$  with probability at least 0.96 in the  
2001 coordinator model.*

2003 *Proof.* Since we use `EFFICIENTCOMMUNICATION`( $\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta$ ) to send  $\tilde{s}_{i-1}^{(j)}$ , by  
2004 [Lemma E.4](#),  $\|\pi(s_{i-1}) - \tilde{s}_{i-1}^{(j)}\|_2 \leq \varepsilon' \|\pi(x) - \pi(s_{i-1})\|_2$  for any  $x \in P_j$ . Hence  $\|\pi(x) - \tilde{s}_{i-1}^{(j)}\|_2$   
2005 would be a  $(1 + \varepsilon')$ -approximation of  $\|\pi(x) - \pi(s_{i-1})\|_2$ .

2007 Since  $\pi$  is a Johnson-Lindenstrauss mapping, by [Theorem B.9](#),  $\|\pi(x) - \pi(y)\|_2 \in [\frac{1}{2}\|x - y\|_2, \frac{3}{2}\|x - y\|_2]$  with probability at least  $1 - \frac{1}{\mathcal{O}(sk)}$ . Therefore,  $\|\pi(x) - \pi(y)\|_2 \in [\frac{1}{2}\|x - y\|_2, \frac{3}{2}\|x - y\|_2]$   
2008 holds for any  $x, y \in P = \cup_{i \in [s]} P_i$  with probability at least 0.99. Since  $\|\pi(x) - \pi(y)\|_2$  is a  
2009 constant approximation of any  $x, y \in P$ , and  $\|\pi(x) - \tilde{s}_{i-1}^{(j)}\|_2$  is a  $(1 + \varepsilon')$ -approximation of  $\|\pi(x) - \pi(s_{i-1})\|_2$ , thus  $\|\pi(x) - \tilde{s}_{i-1}^{(j)}\|_2$  would be a 4-approximation of  $\|x - s_{i-1}\|_2$ .

2013 By sending  $\tilde{D}_j$ , an  $\mathcal{O}(1)$ -approximation of  $\text{cost}(\pi(P_j), S_j)$ , the coordinator owns a  $\mathcal{O}(1)$ -  
2014 approximation of  $\text{cost}(\pi(P_j), S)$ . Thus, we can apply adaptive sampling, and by [Theorem B.10](#),  $S$   
2015 would be an  $(\mathcal{O}(1), \mathcal{O}(1))$ -bicriteria approximation of the optimal solution for  $(k, z)$ -clustering of  
2016  $P = \cup_{j=1}^s P_j$  with probability at least 0.99.

2018 Since  $C_l^{(j)} = \{x \in P_j : \text{dist}(\pi(x), \tilde{s}_l^{(j)}) \leq \text{dist}(\pi(x), \tilde{s}_p^{(j)}), \forall p \leq |S|\}$  and  $\frac{1}{4} \cdot \text{dist}(x, s_p) \leq$   
2019  $\text{dist}(\pi(x), \tilde{s}_p^{(j)}) \leq 4 \cdot \text{dist}(x, s_p)$ , therefore,  $\frac{1}{16} \cdot \text{dist}(x, S) \leq \text{dist}(\pi(x), \tilde{s}_l^{(j)}) \leq 16 \cdot \text{dist}(x, S)$  if  
2020  $x \in C_l^{(j)}$ . Hence, if we apply sensitivity sampling with

$$2022 \mu(x) = \frac{1}{4} \cdot \left( \frac{1}{k \left| \bigcup_{j=1}^s C_l^{(j)} \right|} + \frac{\text{cost}(\pi(x), S_j)}{k \sum_{l \leq |S|} D_l^{(j)}} + \frac{\text{cost}(\pi(x), S_j)}{\sum_{l \leq |S|, j \in [s]} D_l^{(j)}} + \frac{\Delta_x}{\sum_{l \leq |S|, j \in [s]} D_l^{(j)}} \right),$$

2026 where  $D_l^{(j)} = \text{cost}(\pi(C_l^{(j)}), S_j)$  and  $\Delta_p = \frac{\sum_{l \leq |S|} D_l^{(j)}}{\left| \bigcup_{j=1}^s C_l^{(j)} \right|}$ , it would return a  $(1 + \frac{\varepsilon}{4})$ -coreset for  
2027  $P = \cup_{j=1}^s P_j$  with probability at least 0.99.

2029 Since we send  $\tilde{x}$  to the coordinator by `EFFICIENTCOMMUNICATION`( $S, x, \varepsilon', \delta$ ), thus  $\|\tilde{x} - x\|_2 \leq$   
2030  $\varepsilon' \|s - x\|_2$ , where  $s$  is the point in  $S$  closest to  $x$ . Since  $\|\tilde{x} - x\|_2 \leq \varepsilon' \|s - x\|_2$ , by the proof of  
2031 [Lemma B.13](#),  $(1 - \frac{\varepsilon}{4}) \cdot \text{cost}(C, A) \leq \text{cost}(C, A') \leq (1 + \frac{\varepsilon}{4}) \cdot \text{cost}(C, A)$  for any solution  $|C| = k$ .

2032 Since  $P$  is a  $(1 + \frac{\varepsilon}{2})$ -coreset of  $X$ ,  $A$  is a  $(1 + \frac{\varepsilon}{4})$ -coreset of  $P$ , and  $A'$  is a  $(1 + \frac{\varepsilon}{4})$ -coreset of  $A$ ,  
2033 therefore,  $A'$  is a  $(1 + \varepsilon)$ -coreset for  $X$ .

2035 **Evaluation of the success probability.** To apply adaptive sampling, we need to apply  
2036 `EFFICIENTCOMMUNICATION`( $\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta$ ) for every  $P_j$  and  $s_{i-1}$ , which would be  $\mathcal{O}(sk)$   
2037 times of running in total. Since we need to apply `EFFICIENTCOMMUNICATION`( $S, x, \varepsilon', \delta$ ) for  
2038 every point sampled by sensitivity sampling, it would be  $m$  times of running in total. Since  
2039  $\delta \leftarrow \mathcal{O}(\frac{1}{sk+m})$ , the probability that all instances of `EFFICIENTCOMMUNICATION` returns an  
2040 approximate location accurate enough is at least 0.99.

2042 Since the Johnson-Lindenstrauss would preserve the pairwise distance up to a multiple constant  
2043 factor with probability at least 0.99, the probability that adaptive sampling returns an  $(\mathcal{O}(1), \mathcal{O}(1))$ -  
2044 bicriteria approximation is at least 0.99, and sensitivity sampling returns a coreset with probability  
2045 at least 0.99, therefore, the total probability that [Algorithm 16](#) returns a coreset for  $X$  is at least  
2046 0.96.  $\square$

2047 **Lemma E.6.** *Algorithm 16 uses  $\tilde{\mathcal{O}}(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log n)$  bits of communication.*

2049 *Proof.* Since we apply Johnson-Lindenstrauss to map  $P$  to  $\mathbb{R}^{d'}$  where  $d' = \mathcal{O}(\log |P|) =$   
2050  $\mathcal{O}(\log(sk))$ , by [Theorem B.9](#), the coordinator needs to send an  $\mathcal{O}(\log(sk \log(sk)) \cdot \log d)$  bits  
2051 random seed to each site so that every site can generate the map  $\pi$ . It uses  $s \text{polylog}(s, k, d)$  bits.

2052 By Lemma E.4, since  $|P_j| = \mathcal{O}(k)$  and  $d' = \mathcal{O}(\log(sk))$ , we need to use  $\text{polylog}(\log n, s, k, \frac{1}{\varepsilon'}, \frac{1}{\delta})$   
 2053 bits to apply EFFICIENTCOMMUNICATION( $\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta$ ). Since  $|S| = \mathcal{O}(k)$ , we need to  
 2054 use  $d \text{polylog}(\log n, s, k, \frac{1}{\varepsilon'}, \frac{1}{\delta})$  bits to apply EFFICIENTCOMMUNICATION( $S, x, \varepsilon', \delta$ ). Since we  
 2055 need to apply EFFICIENTCOMMUNICATION( $\pi(P_j), \pi(s_{i-1}), \varepsilon', \delta$ ) for  $\mathcal{O}(sk)$  times to send location  
 2056 of  $\pi(s_{i-1})$  to each site, and apply EFFICIENTCOMMUNICATION( $S, x, \varepsilon', \delta$ ) for  $\mathcal{O}\left(\frac{k}{\min(\varepsilon^4, \varepsilon^{2+z})}\right)$   
 2057 times to send location of points sampled by sensitivity sampling to the coordinator, then the total  
 2058 cost to apply EFFICIENTCOMMUNICATION, in bits, is  
 2059

$$2060 \quad 2061 \quad sk \text{polylog}\left(\log n, s, k, \frac{1}{\varepsilon}\right) + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} \text{polylog}\left(\log n, s, k, \frac{1}{\varepsilon}\right). \\ 2062$$

2063 We need to send the exact location of  $s_{i-1}$  to the coordinator, which takes  $\mathcal{O}(dk \log n)$   
 2064 bits. Every site  $j$  sends a constant approximation of  $|C_l^{(j)}|$  and cost  $\left(\pi\left(C_l^{(j)}\right), S_j\right)$  to the  
 2065 coordinator, and the coordinator needs to send constant approximation of  $\sum_{l \leq |S|} |C_l^{(j)}|$  and  
 2066  $\sum_{l \leq |S|} \text{cost}\left(\pi\left(C_l^{(j)}\right), S_j\right)$  to every site, which costs  $\mathcal{O}(sk \log \log n)$  bits for total. Every site  
 2067 needs to send  $\hat{\mu}(P_i)$  to the coordinator, which costs  $\mathcal{O}(s \log \log n)$  bits. The coordinator needs to  
 2068 send  $m_i$ , the number of points to be sampled to every site, which costs  $\mathcal{O}(s \log k)$  bits. We need  
 2069  $\mathcal{O}\left(\frac{1}{\varepsilon} \log \log n\right)$  bits to send the weight  $\hat{\mu}(x)$  to the coordinator.  
 2070

2071 Therefore, the total communication cost for Algorithm 16 is

$$2074 \quad \tilde{\mathcal{O}}\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log n\right) \\ 2075$$

2076 bits. □  
 2077

2078 We now give our full guarantees of our algorithm for the coordinator model.  
 2079

2080 **Theorem E.7.** *There exists an algorithm that returns a  $(1 + \varepsilon)$  coresset of  $X$  with probability at least  
 2081 0.96 in the coordinator model and uses  $\tilde{\mathcal{O}}\left(sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log n\right)$  bits of communication.*  
 2082

2083 *Proof.* The proof follows immediately from Lemma E.5 and Lemma E.6. □  
 2084

### 2085 E.3 CLUSTERING ON GENERAL TOPOLOGIES

2086 We provide a formal definition of distributed clustering under general topologies. Let  $V =$   
 2087  $\{v_1, \dots, v_n\}$  be a set of  $n$  nodes connected via an undirected graph  $G = (V, E)$ , where each edge  
 2088  $(v_i, v_j) \in E$  represents a direct communication link between sites  $v_i$  and  $v_j$ . Each node  $v_i$  holds a  
 2089 local dataset  $X_i$ , and the global dataset is  $X = \bigcup_{i=1}^n X_i$ . Communication is allowed only along the  
 2090 edges of  $G$ .  
 2091

2092 The objective is to compute a set of  $k$  centers  $\mathcal{C} = \{c_1, \dots, c_k\}$  that minimizes the global clustering  
 2093 cost for a given norm parameter  $z \geq 1$ :  
 2094

$$2095 \quad \text{cost}(X, \mathcal{C}) = \sum_{x \in X} d(x, \mathcal{C})^z, \\ 2096$$

2097 where  $d(x, \mathcal{C}) = \min_{c_j \in \mathcal{C}} d(x, c_j)$  is the distance from point  $x$  to its closest center in  $\mathcal{C}$ . The goal is  
 2098 to compute an approximate set of centers  $\hat{\mathcal{C}}$  while minimizing communication over the edges of  $G$ .  
 2099

2100 In Balcan et al. (2013), the clustering problem is studied under this general-topology setting.  
 2101 They assume  $s$  sites are connected by an undirected graph  $G = (V, E)$  with  $m = |E|$  edges.  
 2102 Their algorithm constructs a  $(1 + \varepsilon)$ -coreset, with communication  $\mathcal{O}(m(\frac{dk}{\varepsilon^4} + sk \log sk))$  words  
 2103 for  $k$ -means and  $\mathcal{O}(m(\frac{dk}{\varepsilon^2} + sk))$  words for  $k$ -median. Since a word requires  $\mathcal{O}(d \log n)$  bits  
 2104 in their model, the corresponding costs are  $\mathcal{O}(mdk \log n(\frac{d}{\varepsilon^4} + s \log sk))$  bits for  $k$ -means and  
 2105  $\mathcal{O}(mdk \log n(\frac{d}{\varepsilon^2} + s))$  bits for  $k$ -median.

Our coordinator-model algorithm can be naturally extended to general topologies: each message to the coordinator traverses at most  $m$  edges, multiplying total communication by  $m$ . The resulting communication cost, in bits, becomes

$$\tilde{\mathcal{O}} \left( m \left( sk + \frac{dk}{\min(\varepsilon^4, \varepsilon^{2+z})} + dk \log n \right) \right).$$

Compared to [Balcan et al. \(2013\)](#), our approach improves the dependency on  $n$  by replacing a multiplicative  $\log n$  factor with an additive term, while preserving the same approximation guarantees for general  $(k, z)$ -clustering.

## F ADDITIONAL EMPIRICAL EVALUATIONS

In this section, we perform additional empirical evaluations on both synthetic and real-world datasets to further support our theoretical guarantees. Unlike the experiments in the blackboard model presented in [Section 4](#), we now focus on assessing our algorithms in the *coordinator model*. As a baseline, we use the constant-factor approximation algorithm based on adaptive sampling, denoted **AS**. The second applies a standard dimensionality reduction approach to each of the points, using shared randomness, which can be acquired from public randomness, corresponding with the more communication-efficient variant **AS-JL**. In particular, for a dataset with  $d$  features, we generate a random matrix of size  $d' \times d$ , where each entry is drawn from the scaled normal distribution  $\frac{1}{\sqrt{d'}} \cdot \mathcal{N}(0, 1)$  and  $d'$  is set to be a constant factor smaller than  $d$ , i.e.,  $d' = \frac{d}{2}$  or  $d' = \frac{d}{4}$ . Finally, we apply our compact encoding scheme to the sampled points, centers produced by adaptive sampling, representing our coordinator model algorithm, denoted by **EAS-JL**.

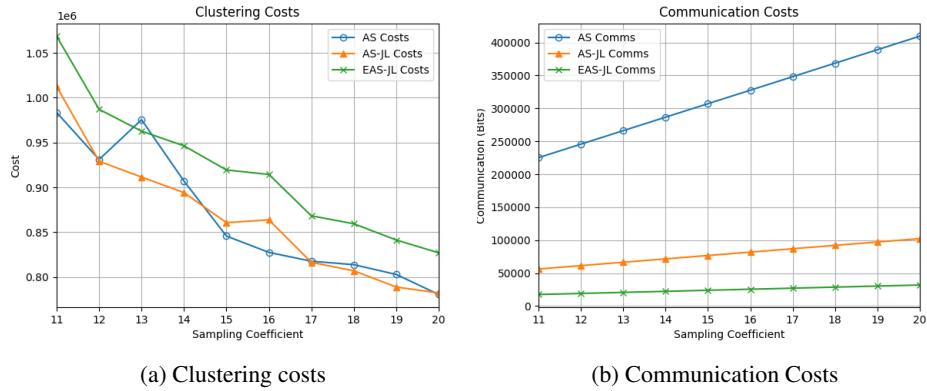


Fig. 7: Experiments for clustering costs and communication costs on DIGITS dataset

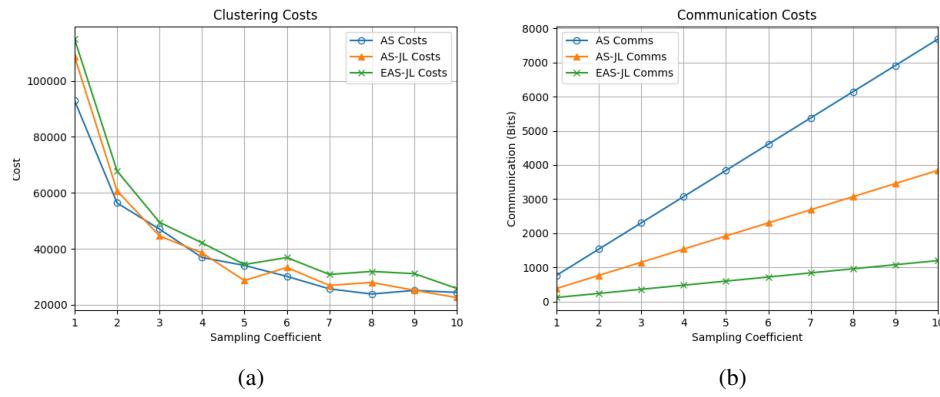
### F.1 REAL-WORLD DATASET

We first evaluated our algorithms on the DIGITS dataset ([Alpaydin & Kaynak, 1998](#)), as previously described in [Section 4](#). As before, we use the parameters  $n = 1797$ ,  $d = 64$ , and  $k = 10$ . We set  $d' = \frac{d}{4} = 16$  for **AS-JL** and **EAS-JL**. We compare the clustering costs in [Figure 5a](#) and communication costs in [Figure 5b](#) for  $c \in \{11, 12, \dots, 20\}$ . Our results show that the clustering costs of the centers returned by our algorithm **EAS-JL** is consistently competitive with the centers returned by the other algorithms **AS** and **AS-JL**, while using significantly less communication across all settings of  $c \in \{11, 12, \dots, 20\}$ . For example [Figure 7](#) shows that at  $c = 11$ , the clustering cost of **EAS-JL** is roughly 1.07 times the clustering cost of **AS** while using 9 $\times$  less communication. This trend seems to continue throughout the range of the sampling coefficient  $c$ , e.g., at  $c = 20$ , the clustering cost of **EAS-JL** remains roughly 1.07 times the clustering cost of **AS** while still using roughly 9 $\times$  less communication.

---

2160 F.2 SYNTHETIC DATASET  
2161

2162 We next evaluated our algorithms on synthetic datasets consisting of Gaussian mixtures, as in Section 4.  
2163 Specifically, we generated  $k = 3$  clusters, each containing 640 points for a total of 1920  
2164 points, each with 8 features, for a total dataset size of 15360. Each cluster was drawn from a distinct  
2165 Gaussian distribution whose mean was selected uniformly at random from the range  $[-10, 10]^8$ , and  
2166 whose covariance matrix was generated as a random positive-definite matrix, producing clusters of  
2167 varying orientations and shapes.

2168 We compare clustering costs and communication costs across a sampling coefficient of  $c \in$   
2169  $\{1, 2, \dots, 10\}$ , and find that EAS-JL consistently achieves clustering performance competitive with  
2170 both AS and AS-JL while using substantially less communication. For instance, at  $c = 5$ , the clus-  
2171 tering costs of EAS-JL and AS are almost equal, while the communication cost of EAS-JL is more  
2172 than a factor of  $4\times$  better. Similarly, at  $c = 10$ , the clustering costs of EAS-JL and AS are al-  
2173 most equal, while the communication cost of EAS-JL is a factor of almost  $8\times$  better. These results  
2174 parallel our findings on real-world data, demonstrating that our algorithm effectively preserves the  
2175 clustering quality while significantly reducing communication in the coordinator model.

2188 Fig. 8: Experiments for clustering costs and communication costs on synthetic dataset  
2189

2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213