On Sampling Based Algorithms for k-Means

Anup Bhattacharya
Indian Statistical Institute Kolkata, India
bhattacharya.anup@gmail.com

Dishant Goyal
Indian Institute of Technology Delhi, India
Dishant.Goyal@cse.iitd.ac.in

Ragesh Jaiswal[1]
Indian Institute of Technology Delhi, India
rjaiswal@cse.iitd.ac.in

Amit Kumar
Indian Institute of Technology Delhi, India
amitk@cse.iitd.ac.in

Abstract

We generalise the results of Bhattacharya et al.[9] for the *list-k-means* problem defined as – for a (unknown) partition $X_1, ..., X_k$ of the dataset $X \subseteq \mathbb{R}^d$, find a *list* of k-center-sets (each element in the list is a set of k centers) such that at least one of k-center-sets $\{c_1, ..., c_k\}$ in the list gives an $(1+\varepsilon)$-approximation with respect to the cost function $\min_{\text{permutation } \pi} \left[\sum_{i=1}^{k} \sum_{x \in X_i} ||x - c_{\pi(i)}||^2\right]$. The list-$k$-means problem is important for the constrained k-means problem since algorithms for the former can be converted to PTAS for various versions of the latter. The algorithm for the list-k-means problem by Bhattacharya et al.is a D^2-sampling based algorithm that runs in k iterations. Making use of a constant factor solution for the (classical or unconstrained) k-means problem, we generalise the algorithm of Bhattacharya et al.in two ways – (i) for any fixed set $X_{j_1}, ..., X_{j_t}$ of $t \leq k$ clusters, the algorithm produces a list of $(\frac{k}{\varepsilon})^{O(\frac{t}{\varepsilon})}$ t-center sets such that (w.h.p.) at least one of them is good for $X_{j_1}, ..., X_{j_t}$, and (ii) the algorithm runs in a single iteration. Following are the consequences of our generalisations:

1. *Faster PTAS under stability and a parameterised reduction*: Property (i) of our generalisation is useful in scenarios where finding good centers becomes easier once good centers for a few "bad" clusters have been chosen. One such case is clustering under stability of Awasthi et al.[5] where the number of such bad clusters is a constant. Using property (i), we significantly improve the running time of their algorithm from $O(dn^3)(k \log n)^{\mathtt{poly}(\frac{1}{\beta}, \frac{1}{\varepsilon})}$ to $O\left(dn^3 \left(\frac{k}{\varepsilon}\right)^{O(\frac{1}{\beta\varepsilon^2})}\right)$. Another application is a parameterised reduction from the *outlier* version of k-means to the classical one where the bad clusters are the outliers.
2. *Streaming algorithms*: The sampling algorithm running in a single iteration (i.e., property (ii)) allows us to design a constant-pass, logspace streaming algorithm for the list-k-means problem. This can be converted to a constant-pass, logspace streaming PTAS for various constrained versions of the k-means problem. In particular, this gives a 3-pass, polylog-space streaming PTAS for the constrained binary k-means problem which in turn gives a 4-pass, polylog-space streaming PTAS for the generalised binary ℓ_0-rank-r approximation problem. This is the first constant pass, polylog-space streaming algorithm for either of the two problems. *Coreset* based techniques, which is another approach for designing streaming algorithms in general, is not known to work for the constrained binary k-means problem to the best of our knowledge.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Facility location and clustering

Keywords and phrases k-means, low rank approximation

[1] Part of this work was done while the author was on a sabbatical from IIT Delhi and visiting UC San Diego.

40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2020).
Editors: Nitin Saxena and Sunil Simon; Article No. 13; pp. 13:1–13:17
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2020.13

Related Version A full version of the paper is available at https://arxiv.org/abs/1909.07511 and https://arxiv.org/abs/1909.11744.

Acknowledgements The authors would like to thank Sanjeev Khanna and Sepehr Assadi for allowing us to use their impossibility argument for the chromatic k-means problem. Anup Bhattacharya would like to thank SERB-National Post Doctoral Fellowship, India. Dishant Goyal would like to thank TCS Research Scholar Program.

1 Introduction

Clustering is one of the most important tools for data analysis and the k-means clustering problem is one of the most prominent mathematical formulations of clustering. The goal of clustering is to partition data objects into groups, called *clusters*, such that similar objects are in the same cluster and dissimilar ones are in different clusters. Defining the clustering problem formally requires us to quantify the notion of similarity/dissimilarity and there are various ways of doing this. Given that in most contexts data objects can be represented as vectors in $\mathbb{R}^d$, a natural notion of distance between data points is the squared Euclidean distance and this gives rise to the k-means problem.

> **k-means**: Given a dataset $X \subset \mathbb{R}^d$ and a positive integer k, find a set $C \subset \mathbb{R}^d$ of k points, called *centers*, such that the following cost function is minimised: $\Phi(C, X) \equiv \sum_{x \in X} \min_{c \in C} \|x - c\|^2$. [2]

The k-means problem has been widely studied by both theoreticians and practitioners and is quite uniquely placed in the computer science research literature. The theoretical worst-case analysis properties of the k-means problem are fairly well understood. The problem is known to be NP-hard [17, 36, 39] and APX-hard [6, 14]. A lot of work has been done on obtaining efficient constant approximation algorithms for this problem (e.g., [30, 2]). However, this is not the main focus of this work. In this work, we discuss approximation schemes for the k-means problem and its variants. Approximation schemes are a family of algorithms $\{A\}_\varepsilon$ that give $(1 + \varepsilon)$-approximation guarantee.

Given the hardness of approximation results, it is known that a *Polynomial Time Approximation Scheme (PTAS)* is not possible unless $\mathsf{P} = \mathsf{NP}$. However, there are efficient approximation schemes when at least one of k, d is not part of the input (and hence assumed to be a fixed constant). The work on approximation schemes for the k-means problem can be split into two categories where one consists of algorithms under the assumption that k is a constant while the other with d as a constant. Assuming k is a constant, there are various PTAS [32, 19, 28, 29] with running time $O(nd \cdot 2^{\tilde{O}(\frac{k}{\varepsilon})})$.[3] Note that the running time has a dependence on 2^k. This is nicely supported by a conditional lower bound result [3] that says that under the *Exponential Time Hypothesis (ETH)* any approximation algorithm (beyond a fixed approximation factor) that runs in time polynomial in n and d will have a running time dependence of at least 2^k. On the other hand, PTAS based on the assumption that d is a constant form another line of research culminating in the work of Addad et al. [15] and Friggstad et al. [23] who gave a local search based PTAS with running time dependence on d

[2] For a singleton set $C = \{c\}$, we will use $\Phi(c, X)$ and $\Phi(\{c\}, X)$ interchangeably.

[3] The multiplicative factor of nd can be changed to an additive factor using useful data analysis tools and techniques such as *coresets* [19] and *dimensionality reduction* [34].

of the form $(\frac{k}{\varepsilon})^{\zeta}$ where $\zeta = \frac{d^{O(d)}}{\varepsilon^{O(\frac{d}{\varepsilon})}}$. The work of Makarychev et al. [37] nicely consolidates the two lines of work by showing that the cost of the optimal k-means solution is preserved up to a factor of $(1+\varepsilon)$ under a projection onto a random $O\left(\frac{\log(k/\varepsilon)}{\varepsilon^2}\right)$-dimensional subspace.

The k-means problem nicely models the *locality* requirement of clustering. That is, similar (or closely located points) should be in the same cluster and dissimilar (or far-away points) should be in different clusters. However, in many different clustering contexts in machine learning and data mining, locality is not the only desired clustering property. There are other constraints in addition to the locality requirement. For example, one requirement is that the clusters should be balanced or in other words contain roughly equal number of points. Modelling such requirements within the framework of the k-means problem gives rise to the so-called *constrained k-means problem*. The constrained k-means problem can be modelled as follows: Let $\mathbb{C}$ denote the set of k-clusterings that satisfy the relevant constraint. Then the goal is to find a clustering $\mathcal{X} = \{X_1, ..., X_k\}$ of the dataset $X \subset \mathbb{R}^d$ such that the clustering $\mathcal{X}$ belongs to $\mathbb{C}$ and the following cost function is minimised:

$$\Delta(\mathcal{X}) \equiv \sum_{i=1}^{k} \Delta(X_i), \text{ where } \Delta(X_i) \equiv \Phi(\mu(X_i), X_i) \text{ and } \mu(X_i) \equiv \frac{\sum_{x \in X_i} x}{|X_i|}.$$

Note that $\mu(X_i)$ is the *centroid* of the data points X_i. It can be easily shown that the centroid gives the best 1-means cost for any dataset and so $\Delta(X_i)$ denotes the optimal 1-means cost of dataset X_i. The above formulation in terms of the feasible clusterings $\mathbb{C}$ is an attempt to give a unified framework for considering different variations of the constrained clustering problem. The issue with such an attempt is how to concisely represent the set of feasible clusterings $\mathbb{C}$. This issue was addressed in the nice work of Ding and Xu [18] who gave a unified framework for considering constrained versions of the k-means problem. For every constrained version, instead of defining $\mathbb{C}$ they define a *partition algorithm* $\mathcal{P}^{\mathbb{C}}$ which, when given a set of k centers $\{c_1, ..., c_k\}$, outputs a feasible clustering $\{X_1, ..., X_k\}$ (i.e., a clustering in $\mathbb{C}$) that minimises the cost $\sum_{i=1}^{k} \Phi(\{c_i\}, X_i)$. They give efficient partition algorithms for a variety of constrained k-means problems. These problems and their description are given in Table 1. Note that the partition algorithm for the k-means problem (i.e., the classical unconstrained version) is simply the *Voronoi partitioning* algorithm.

Efficient partition algorithms allow us to design PTAS in the following manner: Let $\mathcal{X} = \{X_1, ..., X_k\}$ be an optimal clustering for some constrained k-means problem with optimal cost $OPT = \Delta(\mathcal{X}) = \sum_{i=1}^{k} \Delta(X_i)$. Suppose in some way, we are able to find a k-center-set $\{c_1, ..., c_k\}$ such that $\min_{\text{permutation } \pi} \left[\sum_{i=1}^{k} \sum_{x \in X_i} ||x - c_{\pi(i)}||^2\right] \leq (1+\varepsilon) \cdot OPT$. Then we can use the partition algorithm to find a clustering $\bar{\mathcal{X}} = \{\bar{X}_1, ..., \bar{X}_k\}$ such that $\Delta(\bar{\mathcal{X}}) \leq (1+\varepsilon) \cdot OPT$. It turns out that even though producing a single such k-center-set may not be possible, producing a *list* of such k-center-sets is possible. Using the partition algorithm to find the clustering with least cost from the list will give us a $(1+\varepsilon)$-approximate solution. This is the main idea used for designing PTAS by Ding and Xu [18] and Bhattacharya et al. [9]. Bhattacharya et al. [9] gave quantitative improvements over the results of Ding and Xu in terms of the list size. They also formally defined the *list-k-means* problem that is a natural problem in the context of the above discussion.[4] One of the main focus of discussion of this paper will be the list-k-means problem. So, let us first define the problem formally.

[4] Note that Ding and Xu [18] implicitly gave an algorithm for list-k-means without naming it so.

Table 1 Constrained k-means problems with efficient partition algorithm (see Section 4 in [18]).

#	Problem	Description
1.	r-gather k-means clustering (r, k)-`GMeans`	Find clustering $\mathcal{X} = \{X_1, ..., X_k\}$ with minimum $\Delta(\mathcal{X})$ such that for all i, $\lvert X_i\rvert \geq r$
2.	r-Capacity k-means clustering (r, k)-`CaMeans`	Find clustering $\mathcal{X} = \{X_1, ..., X_k\}$ with minimum $\Delta(\mathcal{X})$ such that for all i, $\lvert X_i\rvert \leq r$
3.	l-Diversity k-means clustering (l, k)-`DMeans`	Given that every data point has an associated colour, find a clustering $\mathcal{X} = \{X_1, ..., X_k\}$ with minimum $\Delta(\mathcal{X})$ such that for all i, the fraction of points sharing the same colour inside X_i is $\leq \frac{1}{l}$
4.	Chromatic k-means clustering k-`ChMeans`	Given that every data point has an associated colour, find a clustering $\mathcal{X} = \{X_1, ..., X_k\}$ with minimum $\Delta(\mathcal{X})$ such that for all i, X_i should not have more than one point with the same colour.
5.	Fault tolerant k-means clustering (l, k)-`FMeans`	Find clustering $\mathcal{X} = \{X_1, ..., X_k\}$ such that the sum of squared distances of the points to the l nearest centers out of $\{\mu(X_1), ..., \mu(X_k)\}$, is minimised.
6.	Semi-supervised k-means clustering k-`SMeans`	Given a target clustering $\mathcal{X}' = \{X'_1, ..., X'_k\}$ and constant α find a clustering $\mathcal{X} = \{X_1, ..., X_k\}$ such that the cost $\alpha \cdot \Delta(\mathcal{X}) + (1 - \alpha) \cdot Dist(\mathcal{X}', \mathcal{X})$ is minimised. $Dist$ denotes the set-difference distance.

> **List-k-means**: Let $X \subset \mathbb{R}^d$ be the dataset and let $\mathcal{X} = \{X_1, ..., X_k\}$ be an arbitrary clustering of dataset X. Given X, positive integer k, and error parameter $\varepsilon > 0$, find a *list* of k-center-sets such that (w.h.p.[5]) at least one of the sets gives $(1 + \varepsilon)$-approximation with respect to the cost function: $\Psi(\{c_1, ..., c_k\}, \mathcal{X}) \equiv \min_{\text{permutation } \pi} \left[\sum_{i=1}^{k} \sum_{x \in X_i} ||x - c_{\pi(i)}||^2\right]$.

Bhattacharya et al. [9] gave a lower bound on the list size using a counting argument and a closely matching upper bound using a D^2*-sampling* based approach. D^2-sampling is a simple idea that is very useful in the context of the k-means/median clustering problems. Here, the centers are sampled from the given dataset in successive iterations where the probability of a point getting sampled as the center in an iteration is proportional to the squared distance of this point to the nearest center out of the centers already chosen in the previous iterations. Before discussing the algorithm for the list-k-means problem, let us first make sure that the relevance of this problem in the context of the constrained k-means problems is well understood. Indeed, given any constrained k-means clustering problem with feasible clusterings $\mathbb{C}$ and partition algorithm $\mathcal{P}^{\mathbb{C}}$, one can obtain a $(1 + \varepsilon)$-approximate solution by first running an algorithm for the list k-means problem (where the unknown clustering is any optimal clustering for the constrained k-means problem) to obtain a list $\mathcal{L}$ and then use the partition algorithm $\mathcal{P}^{\mathbb{C}}$ to pick the minimum cost clustering from $\mathcal{L}$. From the previous discussion, it should be clear that this will give us a $(1+\varepsilon)$-approximate solution (w.h.p.). Let us now discuss the D^2-sampling based algorithm for the list-k-means problem.

Bhattacharya et al. [9] gave an algorithm for the list-k-means problem with list size $|\mathcal{L}| = (\frac{k}{\varepsilon})^{O(\frac{k}{\varepsilon})}$ and running time $O(nd|\mathcal{L}|)$. Their algorithm explores a rooted tree of size $(\frac{k}{\varepsilon})^{O(\frac{k}{\varepsilon})}$ and depth k where the degree of every non-leaf vertex is $(\frac{k}{\varepsilon})^{O(\frac{1}{\varepsilon})}$. Every node in this tree has an associated center and the path from root to a leaf node gives one of the k-center-sets for the output list. Let v be an internal node at depth i. The path from root to v defines i centers C_v and their algorithm extends these i centers to $(i + 1)$ centers by D^2-sampling $\texttt{poly}(\frac{k}{\varepsilon})$ points w.r.t. C_v and considering the centroids of all possible subsets

[5] We use w.h.p. as an abbreviation for "with high probability".

of size $O(\frac{1}{\varepsilon})$ of the sampled points plus copies of centers in C_v.[6] This defines the $(\frac{k}{\varepsilon})^{O(\frac{1}{\varepsilon})}$ children of v that are further explored subsequently. In their analysis, they showed that for every node v, there is always (w.h.p.) a child of v that is a good center for one of the clusters for which none of the centers in C_v is good.

Note that the algorithm of Bhattacharya et al.[9] in the previous paragraph has an unavoidable iteration of depth k since their analysis works only when the centers are picked *one-by-one* in k iterations. We circumvent this inherent restriction by using a constant factor approximate solution C to the k-means problem (i.e., the unconstrained k-means problem) for the given dataset X. That is, $\Phi(C, X) \leq \alpha \cdot OPT^{\star}$, where $OPT^{\star}$ denotes the optimal k-means cost. Note that there are a number of constant factor approximation algorithms available for the k-means problem. So, this assumption is not restrictive at all. We can even further relax the assumption by noting that an $(O(1), O(1))$ bi-criteria approximate solution C is sufficient. This means that $|C| = O(k)$ and $\Phi(C, X) \leq \alpha \cdot OPT^{\star}$. There are bi-criteria approximation algorithms available for the k-means problem. For example, there is a simple $O(nkd)$ bi-criteria approximation algorithm based on D^2-sampling that just samples $O(k)$ points (using D^2-sampling) and it has been shown [1] that the set of centers obtained gives a constant approximation with high probability. Making use of a constant factor solution C, we generalise the D^2-sampling based algorithm of Bhattacharya et al. [9] in two ways:

1. We consider the case where we may not need to find good centers for *all* clusters but for $t \leq k$ clusters $X_{j_1}, ..., X_{j_t}$. For any fixed choice of t clusters $X_{j_1}, ..., X_{j_t}$, our algorithm returns a list of $(\frac{k}{\varepsilon})^{O(\frac{t}{\varepsilon})}$ t-center sets such that (w.h.p.) at least one of them is "good" for $X_{j_1}, ..., X_{j_t}$. Note that the list size is exponential in t but not in k. This can be useful in scenarios where finding good centers of most of the clusters becomes easier (or not even required) once good centers of a few $t << k$ clusters have been chosen.
2. The sampling algorithm runs in a *single* iteration where $\texttt{poly}(\frac{t}{\varepsilon})$ points from X are D^2-sampled w.r.t. C. We show that good centers for clusters $X_{j_1}, ..., X_{j_t}$ can simultaneously be found from the sampled points and points in the set C. (Note that there is an iteration for probability amplification in algorithm `GoodCenters` but since the 2^t rounds are independent, they can be executed independently.)

The formal description of the generalised algorithm is given in Algorithm 1. The algorithm takes as input dataset X, an α-approximate solution C, error parameter ε, and t and outputs a list $\mathcal{L}$ of t-center sets. Note that the list size produced by the above algorithm is $|\mathcal{L}| = (\frac{k}{\varepsilon})^{O(\frac{t}{\varepsilon})}$ and running time is $O(nd|\mathcal{L}|)$. We will show that the `GoodCenters` algorithm behaves well (w.h.p.) for **any** fixed set of t clusters $X_{j_1}, ..., X_{j_t}$ out of clusters $X_1, ..., X_k$. What this means is that for **any** fixed set of t clusters $X_{j_1}, ..., X_{j_t}$, the list $\mathcal{L}$ produced by the `GoodCenters` algorithm will (w.h.p.) contain a t-center set $\mathcal{C}$ that is good for these clusters $X_{j_1}, ..., X_{j_t}$. This is our main result on list-k-means and we formally state this as the next theorem.

▶ **Theorem 1** (Main Theorem). *Let $0 < \varepsilon \leq \frac{1}{2}$ and t be any positive integer. Let $X_{j_1}, ..., X_{j_t}$ denote an arbitrary set of t clusters out of k clusters $X_1, ..., X_k$ of the dataset X. Let $\mathcal{L}$ denote the list returned by the algorithm `GoodCenters`(X, C, ε, t). Then with probability at least $\frac{3}{4}$, $\mathcal{L}$ contains a center set $\mathcal{C}$ such that:*

$$\Psi\left(\mathcal{C}, \{X_{j_1}, ..., X_{j_t}\}\right) \leq \left(1 + \frac{\varepsilon}{2}\right) \cdot \sum_{i=1}^{t} \Delta(X_{j_i}) + \frac{\varepsilon}{2} \cdot OPT \leq (1+\varepsilon) \cdot OPT, \quad \textit{where } OPT = \sum_{i=1}^{k} \Delta(X_i).$$

[6] D^2-sampling w.r.t. a center set C implies sampling from the dataset X using a distribution where the probability of sampling point x is proportional to $\min_{c \in C} ||x - c||^2$. In the case $C = \emptyset$, D^2-sampling is the same as uniform sampling.

Algorithm 1 Algorithm for finding good centers.

```
GoodCenters$(X, C, \varepsilon, t)$
    **Inputs**: Dataset $X$, $\alpha$-approximate $C$, accuracy $\varepsilon$, and number of centers $t$
    **Output**: A list $\mathcal{L}$, each element in $\mathcal{L}$ being a $t$-center set
    **Constants**: $\eta = \frac{2^{16}\alpha t}{\varepsilon^4}$; $\tau = \frac{128}{\varepsilon}$
    (1) $\mathcal{L} \leftarrow \emptyset$
    (2) Repeat $2^t$ times:
    (3)     Sample a multi-set $M$ of $\eta t$ points from $X$ using $D^2$-sampling wrt center set $C$
    (4)     $M \leftarrow M \cup \{\frac{128t}{\varepsilon}$ copies of each element in $C\}$
    (5)     For all disjoint subsets $S_1, ..., S_t$ of $M$ such that $\forall i, |S_i| = \tau$:
    (6)         $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mu(S_1), ..., \mu(S_t))\}$
    (7) return($\mathcal{L}$)
```

Moreover, $|\mathcal{L}| = (\frac{k}{\varepsilon})^{O(\frac{t}{\varepsilon})}$ *and the running time of the algorithm is* $O(nd|\mathcal{L}|)$.

We shall formally prove the above theorem in the full version of the paper. We give a high-level discussion here. Without loss of generality, we will assume that $j_i = i$, that is the t clusters $X_{j_1}, ..., X_{j_t}$ are the first t clusters $X_1, ..., X_t$. Since, the input center set C is an (α, β)-approximate solution to the standard k-means problem on dataset X, we have

$$\Phi(C, X) \leq \alpha \cdot OPT^\star \quad \text{and} \quad |C| \leq \beta k \tag{1}$$

Note that the outer iteration (repeat 2^t times in line (2)) is to amplify the probability that the list $\mathcal{L}$ containing a good t-center set. We will show that the probability of finding a good t-center set in one iteration is at least $(3/4)^t$ and the theorem follows from simple probability calculation. So in the remaining discussion we will only discuss one iteration of the algorithm. Consider the multi-set M after line (3) of the algorithm. We will show that with probability at least $(3/4)^t$, there are disjoint (multi) subsets $T_1, ..., T_t$ each of size τ such that for every $j = 1, ..., t$,

$$\Phi(\mu(T_j), X_j) \leq \left(1 + \frac{\varepsilon}{2}\right) \cdot \Delta(X_j) + \frac{\varepsilon}{2t} \cdot OPT. \tag{2}$$

Since we try out all possible subsets in step (5), we will get the desired result. We will argue in the following manner: consider the multi-set $C' = \left\{\frac{16t}{\varepsilon} \text{ copies of each element in } C\right\}$. We can interpret C' as a union of multi-sets $C'_1, C'_2, ..., C'_t$, where $C'_j = \{\frac{16}{\varepsilon}$ copies of each element in $C\}$. Also, since M consists of ηt independently sampled points, we can interpret M as a union of multi-sets $M'_1, M'_2, ..., M'_t$ where M'_1 is the first η points sampled, M'_2 is the second η points and so on. For all $j = 1, ..., t$, let $M_j = C'_j \cup (M'_j \cap X_j)$.[7] We will show that for every $j \in \{1, ..., t\}$, with probability at least $(3/4)$, M_j contains a subset T_j of size τ that satisfies eqn. (2). Note that T_j's being disjoint follows from the definition of M_j. It will be sufficient to prove the following lemma.

▶ **Lemma 2.** *Consider the sets $M_1, ..., M_t$ as defined above. For any $j \in \{1, ..., t\}$,*

$$\mathbf{Pr}\left[\exists T_j \subseteq M_j \ s.t.\ |T_j| = \tau \ and\ \left(\Phi(\mu(T_j), X_j) \leq \left(1 + \frac{\varepsilon}{2}\right) \cdot \Delta(X_j) + \frac{\varepsilon}{2t} OPT\right)\right] \geq \frac{3}{4}.$$

[7] $M'_j \cap X_j$ in this case, denotes those points in the multi-set M'_j that belongs to X_j.

The formal proof of the above lemma is deferred to the full version of the paper. The proof is through a case analysis that is based on whether $\frac{\Phi(C,X_j)}{\Phi(C,X)}$ is large or small for a particular $j \in \{1, ..., t\}$.

- *Case-I*: $\left(\Phi(C, X_j) \leq \frac{\varepsilon}{6\alpha t} \cdot \Phi(C, X)\right)$
  The interpretation of this condition is that the points in X_j are close to centers in the center set C. This means that an appropriate convex combination of points in C will give a good center for X_j. More precisely, here we will show that there is a subset $T_j \subseteq C'_j \subseteq M_j$ that satisfies eqn. (2).
- *Case-II*: $\left(\Phi(C, X_j) > \frac{\varepsilon}{6\alpha t} \cdot \Phi(C, X)\right)$
  This is the case where all points in X_j do not have a close center in the center set C. If we can show that a D^2-sampled set with respect to center set C has a subset S that may be considered uniform sample from X_j, then we can use known results[8] to argue that M_j has a subset T_j such that $\mu(T_j)$ is a good center for X_j. Note that since $\frac{\Phi(C,X_j)}{\Phi(C,X)} > \frac{\varepsilon}{6\alpha t}$, we can argue that if we D^2-sample $poly(\frac{t}{\varepsilon})$ elements, then we will get a good representation from X_j. However, note that some of the points from X_j may have centers in C that are very close and hence will have a very small chance of being D^2-sampled. In such a case, no subset S of a D^2-sampled set will behave like a uniform sample from X_j. So, we need to argue more carefully taking into consideration the fact that there may be points in X_j for which the chance of being D^2-sampled is very small. Here is the high-level argument that we will make:
  - Consider the set X'_j which is same as X_j except that points in X_j that are very close to C have been "collapsed" to their closest center in C.
  - Argue that a good center for the set X'_j is a good center for X_j.
  - Show that a convex combination of copies of centers in C (i.e., C'_j) and D^2-sampled points from X_j gives a good center for the set X'_j.

  More precisely, in this case we will show that M_j contains a subset T_j such that $\Phi(\mu(T_j), X_j) \leq \left(1 + \frac{\varepsilon}{2}\right) \cdot \Delta(X_j)$ and hence T_j also satisfies eqn. (2).

In order to discuss the applications of the `GoodCenters` algorithm, let us note some of its interesting properties. Note that the algorithm essentially runs in a single iteration. The outer loop of size 2^t consists of independent iterations and can be executed independently. The rest of the algorithm clearly follows a single line of control and does not have dependencies. This allows us to design constant-pass streaming algorithms (using *reservoir sampling*) and parallel algorithms. The second useful property is that it finds a good list for *any* fixed set of $t \leq k$ clusters (w.h.p.). This allows us to exploit the algorithm in certain contexts where once good centers for a few clusters have been chosen, choosing good centers for the remaining clusters becomes easier. We discuss the applications of our algorithm in the subsequent subsections.

An interesting point to note about the `GoodCenters` algorithm is that the k-center-set C that it takes as input is only a constant factor approximate solution for the classical k-means problem (i.e., unconstrained version) and not some constrained version. Note that we will use the algorithm for designing PTAS for various constrained versions but constant factor solutions for those are not required. So in some sense, the `GoodCenters` algorithm can be seen as an effective way of converting a constant factor approximate solution for the k-means problem to PTAS for various constrained versions. Let us now discuss the applications.

[8] We use a result from Inaba et al. [27] which says that the centroid of $O(1/\varepsilon)$ uniformly sampled points from any dataset (w.h.p.) gives $(1+\varepsilon)$-approximation with respect to the 1-means cost for the dataset.

1.1 Clustering under stability/separation

The worst-case complexity of the k-means problem is well understood. As discussed earlier, the problem is NP-hard and APX-hard. Hence, various *beyond worst-case* type results have been explored in the context of the k-means problem and one such direction is clustering under some “clusterability” condition. That is, design algorithms for datasets that satisfy some condition that captures the fact that the data is clusterable or in other words the data has some meaningful clusters. Clusterability is captured in various ways using notions such as *separability* and *stability.* Separability means that the target clusters are separated in some geometrical sense and stability means that the target clustering does not change under small perturbations of the input points. Separability and stability are closely related and in various contexts one implies the other. A lot of work has been done in the area of algorithm design for the k-means problem under various clusterability conditions. We will discuss these stability properties and their relationship in detail in the full version of the paper. It can be argued that the β-distributed property of Awasthi et al. [5] given below is one of the weaker separation properties. Hence, any result for datasets satisfying the β-distributed condition will have consequences for datasets satisfying stronger conditions. So the relevant question is: *Are there good algorithms for datasets under this condition?*

▶ **Definition 3** (β-distributed)**.** *A k-means instance (X, k) is called β-distributed iff the following holds for any optimal clustering $\{X_1^\star, ..., X_k^\star\}$: $\forall i, \forall x \notin X_i^\star, ||x - \mu(X_i^\star)||^2 \geq \beta \cdot \frac{OPT^\star}{|X_i^\star|}$.*

Cohen-Addad and Schwiegelshohn [16] gave a local search based algorithm with neighbourhood size $O\left(\beta^{-1} \cdot poly(\varepsilon^{-3})\right)$ that translates to an approximation with running time $O\left(n^{\frac{1}{\beta} poly(\frac{1}{\varepsilon})}\right)$.[9] Awasthi et al. [5] gave a PTAS for the k-means/median problems on datasets that satisfy the β-distributed assumption. The running time has polynomial dependence on the input parameters n, k, d and exponential dependence on β^{-1} and ε^{-1} (ε is the accuracy parameter). Even though they showed that the super-polynomial dependence on β^{-1} and ε^{-1} cannot be improved, improving the dependence on other input parameters was left as an open question. In this work, we address this by giving a faster PTAS for the k-means problem under the β-distributed notion. The running time of the algorithm for the k-means problem by Awasthi et al. [5] is $O(dn^3)(k \log n)^{\mathtt{poly}(\frac{1}{\beta}, \frac{1}{\varepsilon})}$. We improve the running time to $O\left(dn^3 \left(\frac{k}{\varepsilon}\right)^{O(\frac{1}{\beta\varepsilon^2})}\right)$. Note that due to our improvement in running time, our algorithm is also a Fixed Parameter Tractable Approximation Scheme (FPT-AS) for the problem with parameters k and β. Moreover, the running time does not have an exponential dependence on k that is typically the case for such FPT approximation schemes for general datasets. We formally state our result as the following theorem. We shall discuss the proof of this theorem in the full version of the paper.

▶ **Theorem 4.** *Let $\varepsilon, \beta > 0$, k be a positive integer, and let $X \subset \mathbb{R}^d$ be a β-distributed dataset. There is an algorithm that takes as input $(X, k, \varepsilon, \beta)$ and outputs a k-center-set C such that $\Phi(C, X) \leq (1 + \varepsilon) \cdot OPT^\star$ and the algorithm runs in time $O\left(dn^3 \left(\frac{k}{\varepsilon}\right)^{O(\frac{1}{\beta\varepsilon^2})}\right)$.*

Our running time improvements over the algorithm of Awasthi et al. [5] comes from using a faster algorithm to find good centers for a few (constant) optimal clusters called “expensive clusters” in the terminology used by Awasthi et al.They had pointed out that if there was

[9] It may be tempting to think that using the local search algorithm on a coreset (instead of the dataset) will improve the running time to $O(nkd + k^{\frac{1}{\beta} poly(\frac{1}{\varepsilon})})$. However, it is important to realise that known coreset constructions that give coresets of size $poly(k, 1/\varepsilon)$ may not be stability/separation preserving.

a faster algorithm for finding good centers for these expensive clusters, then the overall running time of their algorithm could be significantly improved. This is precisely what our `GoodCenters` algorithm allows us to do. The `GoodCenters` algorithm creates a list such that at least one element in the list is a set of good centers for the expensive clusters. So, one can execute the algorithm of Awathi et al.repeatedly for every element of the list and then pick the best solution. The details are given in the full version of the paper.

1.2 Parameterised reduction from outlier k-means to k-means

The k-means problem models the clustering problem when the data is *noise-free*. That is, the data does not contain outlier points. Clustering algorithms designed for noise-free datasets may behave badly when used for datasets with outliers, where the objective is to cluster the non-outlier points. This is because clustering objective functions such as k-means/median may be sensitive to outliers. This motivates modelling noisy data clustering as a separate problem. One way to model noisy data clustering is through a problem known as *outlier* k-means or k-means with outliers problem. This problem has been studied in a number of previous works [11, 13, 12, 31, 22, 8, 25]. The problem is formally defined as:

> **Outlier k-means**: Given a set of n points $X \subset \mathbb{R}^d$ and positive integers k, m, find a set of k centers $C \subset \mathbb{R}^d$ such that the following cost function is minimised: $\Phi_o(C, X) \equiv \min_{Z \subseteq X, |Z|=m} \left(\sum_{x \in X \setminus Z} \min_{c \in C} ||x - c||^2 \right)$.

This is the same as optimising the k-means cost function on all but at most m points which can be interpreted as outliers. Note that once an optimal center-set C is obtained, the outliers can be located as the farthest m points from the centers in C. In the other direction, suppose we know the m outlier points $Z \subseteq X$, then the optimal center set C may be found by solving the k-means problem on the dataset $X \setminus Z$. The classical k-means problem can be considered a special case of this general problem where $m = 0$. So, the known hardness results for k-means naturally holds for outlier k-means as well. Given this, an interesting problem is to analyse the relative hardness of these problems. In other words, is the outlier k-means problem harder than the classical k-means problem in some sense? One way to formalise this question is to ask whether the outlier version becomes easier if there is an oracle for the k-means problem? In other words, is there an efficient reduction from the outlier-k-means problem to the k-means problem? One brute-force reduction is to consider all possible subsets of m outliers and then solve the k-means problem on the remaining points. However, the running time of this reduction is $\binom{n}{m} = O(n^m)$ which is prohibitively large.

The same question regarding the relative hardness of these problem can also be asked in the approximation setting. The known results on efficient approximation algorithms for these problems makes this question interesting even in the approximation setting. There is a gap in approximation guarantee between the best known efficient approximation algorithm for k-means and outlier k-means. The best known polynomial time approximation guarantee for the k-means problem is 6.358 [2] and for k-means with outliers is 53.003 [31]. So the relevant question is whether this gap can be be removed. We initiate the discussion by giving a parameterised reduction from the outlier k-means problem to the k-means problem. We give a parameterised reduction from the approximate k-means with outliers problems with parameters k, m, and $\frac{1}{\varepsilon}$ to the classical k-means problem.

▶ **Theorem 5.** *Let $0 < \varepsilon \leq \frac{1}{2}$. Let $\mathcal{M}$ be an oracle that returns an optimal solution for arbitrary instances of the k-means problem. Then there exists an algorithm* $\texttt{OutlierAlg}^{\mathcal{M}}(X, k, m, \varepsilon)$ *that returns a $(1 + \varepsilon)$-approximate solution to the outlier k-means problem with probability* *at least $\frac{3}{4}$, where $X \subset \mathbb{R}^d$ and k, m are positive integers. The number of calls made to the oracle $\mathcal{M}$ is bounded by $|\mathcal{L}| = \left(\frac{k+m}{\varepsilon}\right)^{O(\frac{m}{\varepsilon^2})}$ and the running time of the algorithm is bounded by $O(nd \cdot |\mathcal{L}|)$.*

The main idea is to consider the m outliers in an optimal solution as clusters of their own. We can then treat k optimal clusters along with these m *outlier clusters* as the partitioning $X_1, ..., X_{k+m}$ of the dataset. The `GoodCenters` algorithm, when executed with $t = m$, is guaranteed (w.h.p.) to output a list of m-center-sets such that at least one is good for the outlier-clusters. This means that at least one of the m-center-sets will be such that the m centers are close to the outliers. We can exploit this fact to locate good outliers for the dataset, remove them, and solve the k-means problem on the remaining instance. However, since we will need to try all m-center-sets in the list produced by the `GoodCenters` algorithm, we will pay in terms of the running time with a multiplicative factor proportional to the list size. Replacing the k-means oracle $\mathcal{M}$ with a more realistic constant c-approximation algorithm $\mathcal{A}$ for k-means with running time $t(n, k, d)$, we obtain a $(c + c\varepsilon)$-approximation algorithm $\texttt{OutlierAlg}^{\mathcal{A}}$ for outlier k-means with running time $O\left(t(n, k, d) \cdot \left(\frac{k+m}{\varepsilon}\right)^{O(\frac{m}{\varepsilon^2})}\right)$. The consequences of this is that it removes the approximation factor gap between the k-means and outlier k-means problem at the cost of increasing the running time by a factor of $\left(\frac{k+m}{\varepsilon}\right)^{O(\frac{m}{\varepsilon^2})}$. However, one should note that this factor is independent of the problem size and is small when compared to the brute-force reduction (considering all possible subsets of m outliers) with associated factor of $O(n^m)$.

Using the `GoodCenters` algorithm in a different manner in the outlier setting gives us another interesting consequence. The `GoodCenters` algorithm, when executed with $t = k$ is guaranteed (w.h.p.) to output a list of k-center-sets such that at least one is good for the k optimal clusters. This gives an FPT-approximation scheme (with parameters k and m) for the outlier k-means problem with running time $O(nd \cdot f(k, m, \varepsilon))$ and furthermore a 4-pass streaming algorithm that uses $O(f(k, m, \varepsilon) \cdot \log n)$-space, where $f(k, m, \varepsilon) = O\left(nd \left(\frac{m+k}{\varepsilon}\right)^{O(\frac{k}{\varepsilon})}\right)$. The details of this section are given in the full version of the paper.

1.3 Streaming algorithms for constrained versions of k-means

We discussed how an algorithm for the list-k-means problem can be converted to a PTAS for a constrained k-means problem given that there is a *partition algorithm* that finds a feasible clustering with the smallest k-means cost. Examining the `GoodCenters` algorithm closely, we realise that it can be implemented in 2-passes using small amount of space. This opens the door for designing streaming PTAS for the constrained versions of the k-means problem. If one can design a streaming version of the partition algorithm for some constrained k-means problem, then combining it with the streaming version of the `GoodCenters` algorithm will give us a streaming PTAS for the problem. So, let us first discuss how a streaming version of the `GoodCenters` algorithm can be designed.

The first bottleneck in designing a streaming version of `GoodCenters` is that we need a constant factor approximate solution C for the k-means problem (i.e., the unconstrained k-means problem). Fortunately, there exists a 1-pass, logspace streaming algorithm that gives a constant factor approximate solution to the k-means problem [10]. Given C, we need to show how to implement step (3) of the algorithm in a streaming manner (the 2^t repetitions can be performed independently, this appears as a multiplicative factor in the space used). The probability of sampling a point p is proportional to $\Phi(C, p)$, with the constant of proportionality being $\Phi(C, X)$. The sampling can be performed using the ideas of *reservoir sampling* (see e.g. [40]). Since we need to sample $\eta t \leq \texttt{poly}(\frac{k}{\varepsilon})$ points in step (3), reservoir sampling takes $O\left(\texttt{poly}(\frac{k}{\varepsilon}) \cdot \log n\right)$ space. Given a sample M, steps (5)-(6) can be implemented in $O(|M|^{k\tau})$ space, where $\tau = O(\frac{1}{\varepsilon})$. This can be summarised formally as the following useful lemma that we will prove in the full version of the paper (we assume that storing a point accounts for one unit of space).

▶ **Lemma 6.** *The algorithm* `GoodCenters` *can be implemented using* 2*-passes over the input data while maintaining space of* $O(f(k,\varepsilon) \cdot \log n)$*, where* $f(k,\varepsilon) = \left(\frac{k}{\varepsilon}\right)^{O(\frac{k}{\varepsilon})}$.

Let us now see how to design a streaming PTAS for a constrained k-means problem using the above lemma. Let $\mathcal{P}^{\mathbb{C}}$ denote the partition algorithm for this constrained problem and suppose there is a streaming version $\mathcal{SP}^{\mathbb{C}}$ of this partition algorithm. We will use the 2-pass streaming version of the `GoodCenters` algorithm to output the list $\mathcal{L}$. We will then use $\mathcal{SP}^{\mathbb{C}}$ on each element of $\mathcal{L}$ (independently) and pick the best solution. Since $|\mathcal{L}|$ is small, so is the space requirement. From the previous discussion, we know that (w.h.p.) we are guaranteed to obtain a $(1+\varepsilon)$-approximate solution. Hence we get a constant pass streaming PTAS. So, as long as there is a streaming partition algorithm for a constrained k-means problem, there is also a streaming PTAS. Now the question is whether there are constrained k-means problems for which such streaming partition algorithms can be designed. Interestingly, we can design such streaming partition algorithms for four out of the six constrained k-means problems in Table 1. Our results can be summarised as the following main theorem the proof of which is deferred to the full version of the paper. Here, Δ is the *aspect ratio*, i.e., $\Delta = \frac{\max_{p \in X, c \in C} ||p-c||}{\min_{p \in X \setminus C, c \in C} ||p-c||}$.

▶ **Theorem 7.** *There is a* $(1+\varepsilon)$*-approximate,* 4*-pass, streaming algorithm for the following constrained* k*-means clustering problems that uses* $O(f(k,\varepsilon) \cdot (\log \Delta + \log n))$*-space and* $O(d \cdot f(k,\varepsilon))$ *time per item, where* $f(k,\varepsilon) = (\frac{k}{\varepsilon})^{O(\frac{k}{\varepsilon})}$*:*

1. *k-means*
2. *r-gather k-means*
3. *r-capacity k-means*
4. *Fault tolerant k-means*
5. *Semi-supervised k-means*

Further, the space requirement can be improved to $O(f(k,\varepsilon) \cdot \log n)$ *using* 5*-passes.*

Note that the classical k-means problem can also be seen as a constrained k-means problem where there are no constraints. Also note that two constrained versions of constrained k-means problems from Table 1 are missing from the theorem above. These are the *chromatic k-means clustering* and the *l-diversity clustering*. We can show that deterministic logspace streaming algorithms for these problems are not possible. Due to space limitations, this is shown in the full version of the paper.

Comparison with Coreset based streaming algorithms

Streaming *coreset* constructions provide another approach to designing streaming algorithm for the k-means problem. An (ε, k) coreset of a dataset $X \subset \mathbb{R}^d$ is a weighted set $S \subset \mathbb{R}^d$ along with a weight function $w : S \to \mathbb{R}^+$ such that for any k-center-set C, we have: $|\sum_{s \in S} \min_{c \in C} w(s) \cdot \|s-c\|^2 - \sum_{x \in X} \min_{c \in C} \|x-c\|^2| \leq \varepsilon \cdot \sum_{x \in X} \min_{c \in C} \|x-c\|^2$ So, it is sufficient to find good k-center-set for a coreset S (instead of the dataset X). There exists one-pass streaming coreset construction [19] that uses $poly(k, \frac{1}{\varepsilon}, \log n)$ space and outputs a coreset of size $poly(k, \frac{1}{\varepsilon}, \log n)$. Using this, one can design a single-pass streaming algorithm for the k-means problem by first running the streaming algorithm to output a coreset and then finding a good k center set for the small coreset. If the output is supposed to be a clustering, then we will need to make another pass over the data. Note that the same idea of working on coreset does not trivially carry over to the constrained versions of k-means as there are additional constraints. However, there is a specific geometric coreset construction which works for constrained versions of k-means. This is one of the first coreset constructions for k-means by Har-Peled and Mazumdar [26] where the points in the coreset are such that the sum total of the distance of the data points to the nearest coreset point is small. The weight of a coreset point is simply the number of data points for which the coreset point is the closest. So, a coreset point *represents* a subset of data points. Schmidt et al. [38] used this construction for a contrained version called Fair k-means. This coreset construction can be performed in a single pass over the data. The coreset size is $O(k\varepsilon^{-d} \log n)$ and it can be computed in as much space using ideas developed later (e.g., [20]). Even though this gives a one-pass algorithm for producing a good center set (two passes for producing clustering), the space requirement is exponentially large in the dimension. Fortunately, in a more recent development by Makarychev et al. [37] showed that the k-means cost of *any* clustering is preserved up to a factor of $(1+\varepsilon)$ under a projection onto a random $O\left(\frac{\log(k/\varepsilon)}{\varepsilon^2}\right)$-dimensional subspace. This result when combined with the geometric coreset construction of Har-Peled and Mazumdar [26] gives a one-pass, $O\left(\left(\frac{k}{\varepsilon}\right)^{\frac{1}{\varepsilon^2}} \cdot \log n\right)$-space algorithm for producing a good k-center-set for *any* constrained version of the k-means problem. Even though the space bound has a slightly worse dependency on $1/\varepsilon$ than our list-k-means based idea, the dependency on k and number of passes is much better. Indeed, we overlooked this connection with coreset of Har-Peled and Mazumdar and dimension reduction of Makarychev et al.when we were designing our list-k-means based streaming algorithms and were made to realise this at a later stage of this work. At this point, all we can say is that designing streaming algorithm based on list-k-means is another way of approaching constrained k-means problem. Furthermore, we decided to include this section since some of the techniques developed here may have independent applications. We also note that coreset based technique does not seem to work for the constrained binary k-means which is also a problem does not fit into the unified framework of Ding and Xu [18]. This is because current known techniques for finding good centers for this problem requires uniform samples from the optimal clusters and it is not clear whether working with representative points (as in the coreset) will work. We discuss constrained binary k-means and a related problem next.

1.4 Streaming algorithms for binary-k-means and low rank approximation

Low rank approximation is a common data analysis task. The most general version of the problem, the ℓ_p-low rank approximation problem, is defined in the following manner:

> *ℓ_p-low rank approximation*: Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ (with $n \geq d$) and an integer r, find a rank-r matrix $\mathbf{B} \in \mathbb{R}^{n \times d}$ such that $\|\mathbf{A} - \mathbf{B}\|_p^p \equiv \sum_{i,j} |\mathbf{A}_{i,j} - \mathbf{B}_{i,j}|^p$ is minimised.

The above definition is for any positive value of p. When $p = 0$, the objective is to minimise $\|\mathbf{A} - \mathbf{B}\|_0$ which is defined to be the number of mis-matches in the matrices $\mathbf{A}$ and $\mathbf{B}$. The ℓ_p-low rank approximation problem is known to be NP-hard for $p \in \{0, 1\}$ while for $p = 2$ the problem can be solved using SVD (Singular Value Decomposition). The specific case of $p = 0$ is known as the *ℓ_0-low rank approximation problem.* The problem can alternatively be stated as: given an $n \times d$ matrix $\mathbf{A}$, find an $n \times r$ matrix $\mathbf{U}$ and a $r \times d$ matrix $\mathbf{V}$ such that $\|\mathbf{A} - \mathbf{U} \cdot \mathbf{V}\|_0$ is minimised. There is an interest in specific class of instances of the ℓ_0-low rank approximation problem where the matrices $\mathbf{A}, \mathbf{U}, \mathbf{V}$ are binary matrices. In fact, we can generalise even further by making the notion of $\mathbf{U} \cdot \mathbf{V}$ in the above definition more flexible in the following manner: If $\mathbf{A}' = \mathbf{U} \cdot \mathbf{V}$, then $\mathbf{A}'_{ij}$ is the inner product of the i^{th} row of $\mathbf{U}$ and the j^{th} column of $\mathbf{V}$. We can consider various fields for this inner product. The two popularly explored fields are: (i) $\mathbb{F}_2$ with inner product defined as $\langle x, y \rangle \equiv \oplus_i (x_i \cdot y_i)$, and (ii) Boolean semiring $\{0, 1, \wedge, \vee\}$ with inner product defined as $\langle x, y \rangle \equiv \vee_i (x_i \wedge y_i) = 1 - \prod_i (1 - x_i \cdot y_i)$. We can generalise the problem (using the formulation in terms of $\mathbf{U}$ and $\mathbf{V}$) so that the above versions become special cases. This was done by Ban et al. [7] and they called this problem *generalised binary ℓ_0-rank-r problem* that is defined below.

> *Generalised binary ℓ_0-rank-r approximation*: Given a matrix $\mathbf{A} \in \{0,1\}^{n \times d}$ with $n \geq d$, an integer r, and an inner product function $\langle ., . \rangle : \{0,1\}^r \times \{0,1\}^r \to \{0,1\}$, find matrices $\mathbf{U} \in \{0,1\}^{n \times r}$ and $\mathbf{V} \in \{0,1\}^{r \times d}$ that minimises $\|\mathbf{A} - \mathbf{U} \cdot \mathbf{V}\|_0$, where $\mathbf{U} \cdot \mathbf{V}$ is computed using the inner product function. That is $[\mathbf{U} \cdot \mathbf{V}]_{ij}$ is the inner product of the i^{th} row of $\mathbf{U}$ with the j^{th} column of $\mathbf{V}$.

Ban et al. [7] showed that there is no approximation algorithm for the generalised binary ℓ_0-rank-r problem running in time $2^{2^{\delta r}}$ for a constant $\delta > 0$ even though faster algorithms are known for certain specific versions [33]. The work of Ban et al. [7] and Fomin et al. [21] addressed one of the main open questions for generalised binary ℓ_0 rank-r problem – whether a PTAS for constant r is possible. They give such a PTAS using very similar set of ideas (even though they were obtained independently). We extend the previous work of Ban et al.and Fomin et al.to the streaming setting by using the connection of this problem to the *constrained binary k-means problem* which we discuss next. This connection was given and used by both Ban et al. [7] and by Fomin et al. [21]. We will work with the definition of the constrained binary k-means problem given by Fomin et al. [21]. For this, we first need to define the concept of a set of k centers $C \subseteq \{0,1\}^d$ satisfying a set of k-ary relations. Given a set $\mathcal{R} = \{R_1, ..., R_d\}$ of d, k-ary binary relations (i.e., $R_i \subseteq \{0,1\}^k$ for every i), a set $C = \{c_1, ..., c_k\} \subseteq \{0,1\}^d$ of k centers is said to satisfy $\mathcal{R}$ iff $(c_1[i], ..., c_k[i]) \in R_i$ for every $i = 1, ..., d$. Here, $c_j \in \{0,1\}^d$ is thought of as a d-dimensional vector and $c_j[i]$ denotes the i^{th} coordinate of this vector. We can now define the constrained binary k-means problem.

> *Constrained binary k-means*: Given a set of n points $X \subseteq \{0,1\}^d$, a positive integer k, and a set of k-ary relations $\mathcal{R} = \{R_1, ..., R_d\}$, find a set of k centers $C \subseteq \{0,1\}^d$ satisfying $\mathcal{R}$ such that the cost function $\Phi(C, X) \equiv \sum_{x \in X} \min_{c \in C} \|x - c\|_2^2 = \sum_{x \in X} \min_{c \in C} \mathcal{H}(x, c)$ is minimised. Here $\mathcal{H}(., .)$ denotes the Hamming distance.

It is important to distinguish between the definition of constrained binary k-means problem given above with the constrained k-means problem discussed earlier. The relevant question to ask is: *Does the constrained binary k-means problem fit into the unified framework of Ding and Xu [18]?* If the answer to the above question were yes, then a streaming PTAS for the constrained binary k-means problem would trivially follow from the earlier discussion on constrained k-means. Unfortunately, this is not true. Note that the framework of Ding and Xu [18] defines the constraints on the clusters while the definition of constrained binary k-means problem defines constraints on the centers. However, we note that the D^2-sampling based techniques can be extended to this setting. Below, we formally state our main results for the constrained binary-k-means problem.

▶ **Theorem 8.** *Let $0 < \varepsilon \leq 1/2$. There is a 3-pass streaming algorithm that outputs a $(1+\varepsilon)$-approximate solution for any instance of the constrained binary k-means problem. The space and per-item processing time of our algorithm is $O\left(d \cdot (\log n)^k \cdot 2^{\tilde{O}(\frac{k^2}{\varepsilon^2})}\right)$.*

Note that as per the formulation of the constrained binary k-means problem, the output is supposed to be a set of k centers. The above 3-pass algorithm outputs such a k-center-set. However, if the objective is to output the clustering of the data points X, then one more pass over the data will be required and the resulting algorithm will be a 4-pass algorithm. This is relevant for the generalised binary ℓ_0-rank-r approximation problem that we discuss next. We obtain a result for the generalised binary ℓ_0-rank-r problem that is similar to the above result, using a simple reduction. This reduction is used by both Fomin et al. [21] and Ban et al. [7]. We restate the result of Fomin et al. [21] for clarity.

▶ **Lemma 9** (Lemma 1 and 2 of [21]). *For any instance $(\boldsymbol{A}, r)$ of the generalised binary ℓ_0-rank-r approximation problem, one can construct in time $O(n + d + 2^{2r})$ an instance $(X, k = 2^r, \mathcal{R})$ of constrained binary k-means problem with the following property: Given any α-approximate solution C of $(X, k, \mathcal{R})$, an α-approximate solution $\boldsymbol{B}$ of $(\boldsymbol{A}, r)$ can be constructed in time $O(rnd)$.*

The dataset X corresponding to matrix $\mathbf{A}$, in the above reduction, is essentially the rows of the matrix $\mathbf{A}$ and $\forall i, R_i = \{(\langle x, \lambda_1 \rangle, ..., \langle x, \lambda_k \rangle) : x \in \{0,1\}^r\}$ and λ_i's are pairwise distinct vectors in $\{0,1\}^r$. The above reduction and Theorem 8 gives the following main result for the generalised binary ℓ_0-rank-r approximation problem. Note that since we need to output a matrix $\mathbf{B}$, we will need the clustering of the rows of $\mathbf{A}$ and as per previous discussion this will require one more pass than that in Theorem 8.

▶ **Theorem 10.** *Let $0 < \varepsilon \leq 1/2$. There is a 4-pass streaming algorithm that makes row-wise passes over the input matrix and outputs a $(1+\varepsilon)$-approximate solution for any instance of the generalised binary ℓ_0-rank-r problem. The space and per-item processing time of our algorithm is $O\left(d \cdot (\log n)^{2^r} \cdot 2^{\tilde{O}(\frac{2^{2r}}{\varepsilon^2})}\right)$.*

The details of this section are given in the full version of the paper.

1.5 Conclusion and open problems

Our results demonstrate the versatility of the sampling based approach for k-means. This has also been demonstrated in some of the past works. The effectiveness of k-means++ (which is basically D^2-sampling in k rounds) is well known [4]. The D^2-sampling technique has been used to give simple PTAS for versions of the k-means/median problems with various metric-like distance measures [28] and also various constrained variations of k-means [9]. It has also been used to give efficient algorithms in the semi-supervised setting [3, 24] and coreset construction [35]. In this work, we see its use in the streaming, outlier, and clustering-under-stability settings. The nice property of the sampling based approach is that we have a uniform template of the algorithm that is simple and that works in various different settings. This essentially means that the algorithm remains the same while the analysis changes.

This work raises many interesting questions. Our main result on list-k-means is a sampling algorithm that helps us find good centers for *any* subset of t clusters. We made use of this property in clustering-under-stability and outlier settings. There may be other such settings where the clustering problem may become easier once good centers for a few clusters have been chosen. Our discussion on outlier k-means raises an interesting question related to the relative hardness of the k-means and the outlier k-means problem. In the streaming setting for the constrained k-means, we give a generic algorithm within the unified framework of Ding and Xu [18]. The advantage of working in this unified framework is that we get streaming algorithms for various constrained versions of the k-means problem. However, it may be possible to obtain better streaming algorithms (in terms of space, time, and number of passes) for the constrained problems when considered separately as is the case for the classical k-means problem [10]. It may be worthwhile exploring these problems.

References

1 Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Proceedings of the 12th International Workshop and 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, APPROX '09 / RANDOM '09, pages 15–28, Berlin, Heidelberg, 2009. Springer-Verlag. doi:10.1007/978-3-642-03685-9_2.

2 S. Ahmadian, A. Norouzi-Fard, O. Svensson, and J. Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 61–72, October 2017. doi:10.1109/FOCS.2017.15.

3 Nir Ailon, Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Approximate Clustering with Same-Cluster Queries. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 40:1–40:21, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ITCS.2018.40.

4 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 '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1283383.1283494.

5 Pranjal Awasthi, Avrim Blum, and Or Sheffet. Stability yields a PTAS for k-median and k-means clustering. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 309–318, Washington, DC, USA, 2010. IEEE Computer Society. doi:10.1109/FOCS.2010.36.

6 Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The Hardness of Approximation of Euclidean k-Means. In Lars Arge and János Pach, editors, *31st International Symposium on Computational Geometry (SoCG 2015)*, volume 34 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 754–767, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SOCG.2015.754.

7 Frank Ban, Vijay Bhattiprolu, Karl Bringmann, Pavel Kolev, Euiwoong Lee, and David P. Woodruff. A PTAS for ℓ_p-low rank approximation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, pages 747–766, Philadelphia, PA, USA, 2019. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=3310435.3310482.

8 Aditya Bhaskara, Sharvaree Vadgama, and Hong Xu. Greedy sampling for approximate clustering in the presence of outliers. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 11146–11155. Curran Associates, Inc., 2019. URL: http://papers.nips.cc/paper/9294-greedy-sampling-for-approximate-clustering-in-the-presence-of-outliers.pdf.

9 Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Faster algorithms for the constrained k-means problem. *Theory of Computing Systems*, 62(1):93–115, January 2018.

10 Vladimir Braverman, Adam Meyerson, Rafail Ostrovsky, Alan Roytman, Michael Shindler, and Brian Tagiku. Streaming k-means on well-clusterable data. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 26–40, Philadelphia, PA, USA, 2011. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2133036.2133039.

11 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, page 642–651, USA, 2001. Society for Industrial and Applied Mathematics.

12 Sanjay Chawla and Aristides Gionis. *k-means: A unified approach to clustering and outlier detection*, pages 189–197. Society for Industrial and Applied Mathematics, 2013. doi:10.1137/1.9781611972832.21.

13 Ke Chen. A constant factor approximation algorithm for k-median clustering with outliers. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, page 826–835, USA, 2008. Society for Industrial and Applied Mathematics.

14 V. Cohen-Addad and Karthik C.S. Inapproximability of clustering in lp metrics. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 519–539, November 2019. doi:10.1109/FOCS.2019.00040.

15 Vincent Cohen-Addad, Philip N. Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in euclidean and minor-free metrics. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 00:353–364, 2016. doi:10.1109/FOCS.2016.46.

16 Vincent Cohen-Addad and Chris Schwiegelshohn. On the local structure of stable clustering instances. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 49–60, 2017. doi:10.1109/FOCS.2017.14.

17 Sanjoy Dasgupta. The hardness of k-means clustering. Technical Report CS2008-0916, Department of Computer Science and Engineering, University of California San Diego, 2008.

18 Hu Ding and Jinhui Xu. A unified framework for clustering constrained data without locality property. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 1471–1490, 2015. doi:10.1137/1.9781611973730.97.

19 Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the twenty-third annual symposium on Computational geometry*, SCG '07, pages 11–18, New York, NY, USA, 2007. ACM. doi:10.1145/1247069.1247072.

20 Hendrik Fichtenberger, Marc Gillé, Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Bico: Birch meets coresets for k-means clustering. In Hans L. Bodlaender and Giuseppe F. Italiano, editors, *Algorithms – ESA 2013*, pages 481–492, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

21 Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Approximation schemes for low-rank binary matrix approximation problems. *CoRR*, abs/1807.07156, 2018. arXiv:1807.07156.

22 Zachary Friggstad, Kamyar Khodamoradi, Mohsen Rezapour, and Mohammad R. Salavatipour. Approximation schemes for clustering with outliers. *ACM Trans. Algorithms*, 15(2), February 2019. doi:10.1145/3301446.

23 Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a PTAS for k-means in doubling metrics. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 00:365–374, 2016. doi:10.1109/FOCS.2016.47.

24 Buddhima Gamlath, Sangxia Huang, and Ola Svensson. Semi-Supervised Algorithms for Approximately Optimal and Accurate Clustering. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 57:1–57:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2018.57.

25 Shalmoli Gupta, Ravi Kumar, Kefu Lu, Benjamin Moseley, and Sergei Vassilvitskii. Local search methods for k-means with outliers. *Proc. VLDB Endow.*, 10(7):757?768, March 2017. doi:10.14778/3067421.3067425.

26 Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 291–300, New York, NY, USA, 2004. ACM. doi:10.1145/1007352.1007400.

27 Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *Proceedings of the tenth annual symposium on Computational geometry*, SCG '94, pages 332–339, New York, NY, USA, 1994. ACM. doi:10.1145/177424.178042.

28 Ragesh Jaiswal, Amit Kumar, and Sandeep Sen. A simple D^2-sampling based PTAS for k-means and other clustering problems. *Algorithmica*, 70(1):22–46, 2014. doi:10.1007/s00453-013-9833-9.

29 Ragesh Jaiswal, Mehul Kumar, and Pulkit Yadav. Improved analysis of D^2-sampling based PTAS for k-means and other clustering problems. *Information Processing Letters*, 115(2):100–103, 2015. doi:10.1016/j.ipl.2014.07.009.

30 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Proc. 18th Annual Symposium on Computational Geometry*, pages 10–18, 2002. doi:10.1145/513400.513402.

31 Ravishankar Krishnaswamy, Shi Li, and Sai Sandeep. Constant approximation for k-median and k-means with outliers via iterative rounding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 646–659, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188882.

32 Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2):5:1–5:32, February 2010. doi:10.1145/1667053.1667054.

33 Ravi Kumar, Rina Panigrahy, Ali Rahimi, and David Woodruff. Faster algorithms for binary matrix factorization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3551–3559, Long Beach, California, USA, 09–15 June 2019. PMLR. URL: http://proceedings.mlr.press/v97/kumar19a.html.

34 Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, June 1995. doi:10.1007/BF01200757.

35 Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training gaussian mixture models at scale via coresets. *J. Mach. Learn. Res.*, 18(1):5885–5909, January 2017. URL: http://dl.acm.org/citation.cfm?id=3122009.3242017.

36 Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is NP-hard. *Theor. Comput. Sci.*, 442:13–21, July 2012. doi:10.1016/j.tcs.2010.05.034.

37 Konstantin Makarychev, Yury Makarychev, and Ilya P. Razenshteyn. Performance of johnson-lindenstrauss transform for k-means and k-medians clustering. *CoRR*, abs/1811.03195, 2018. arXiv:1811.03195.

38 Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In Evripidis Bampis and Nicole Megow, editors, *Approximation and Online Algorithms*, pages 232–251, Cham, 2020. Springer International Publishing.

39 Andrea Vattani. The hardness of k-means clustering in the plane. Technical report, Department of Computer Science and Engineering, University of California San Diego, 2009.

40 J S Vitter. Random sampling with a reservoir. *ACM Trans. Math. Software*, 11(1):37–57, 1985.