K-Means Clustering with Distributed Dimensions

Hu Ding HUDING@MSU.EDU
Computer Science and Engineering, Michigan State University, East Lansing, MI, USA

Yu Liu LIUYUJYYZ@126.COM
Lingxiao Huang HUANGLINGXIAO1990@126.COM
Jian Li LAPORDGE@GMAIL.COM
Institute for Interdisciplinary Information Science, Tsinghua University, Beijing, China

Abstract

Distributed clustering has attracted significant attention in recent years. In this paper, we study the k-means problem in the distributed dimension setting, where the dimensions of the data are partitioned across multiple machines. We provide new approximation algorithms, which incur low communication costs and achieve constant approximation ratios. The communication complexity of our algorithms significantly improve on existing algorithms. We also provide the first communication lower bound, which nearly matches our upper bound in a certain range of parameter setting. Our experimental results show that our algorithms outperform existing algorithms on real data-sets in the distributed dimension setting.

1. Introduction

Clustering is a central problem in unsupervised learning and data analytics. Among hundreds of clustering problems and algorithms, the k-means problem is arguably the most popular one, due to its simplicity and effectiveness in numerous applications. Specifically, given n points in $\mathbb{R}^d$, k-means seeks k points (cluster centers) to induce a partition of the given point-set, so that the average squared distance from each data point to its closest cluster center is minimized.

Due to the sheer increase in volume and dimensionality of data, clustering (in particular k-means) in the distributed setting has attracted significant attention from various research communities in recent years. While much prior work has focused on the model where the data records (points, items) are distributed in different locations, we study the k-means problem in the *distributed dimension* model, in which the dimensions of the data points are stored across multiple machines.

Suppose the input data points are vectors in $\mathbb{R}^d$. In the *distributed dimension* model, each party is assigned a subset of the d dimensions, and holds the projection of all the data points onto these dimensions. If the whole input point-set is represented by an $n \times d$ matrix, where each row indicates one point and each column indicates one dimension, each party holds a column partition in our model, while the prior work focused on row partition. The distributed dimension (or column partition) model arise naturally in a variety of applications and data management platforms, some of which we briefly mention below.

(1). Suppose that we want to cluster online users based on several features such as online shopping records, online social relationships, and job hunting records, which are collected by different companies (Amazon, Facebook, and Linkedin) (assuming that there is a unique id, e.g., email address, which can be used to identify the same user). It is also quite common that different teams in the same company develop apps which automatically store different features in different machines. **(2).** Several widely deployed distributed DBMS systems, such as MonetDB, C-Store, VectorWise, HBASE, are *column-oriented* by design (Stonebraker et al., 2005; Abadi et al., 2009; 2013). These systems partition and store the data based on the columns.[1] **(3).** Another motivation comes from the proliferation of networked sensing systems, where we need to monitor several attributes of the objects using multiple sensors of different types. For example, an environment monitoring system (Gu et al., 2005; Su et al., 2011) may have video cameras, and different sensors recording audio, temperature, il-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

[1] Such systems typically perform compression of the columns, particularly for sparse data. We ignore such issue and assume the raw columns are stored and readily available in each server.

lumination, *etc.,* separately.

1.1. Formulation

We formally introduce some notations and our communication model. We use $[n]$ as a short-hand notation for $\{1, 2, \cdots, n\}$. Let $P = \{p_1, p_2, \cdots, p_n\} \subset \mathbb{R}^d$ be the set of points we want to cluster. There are T parties. The l-th party holds a subset $\mathcal{D}_l$ of d_l dimensions. Assume that $\cup_{l=1}^{T} \mathcal{D}_l = [d]$ and $\sum_{l=1}^{T} d_l = d$ (i.e., $\{\mathcal{D}_l\}_{l=1,\ldots,T}$ is a partition of $[d]$). For each point $p_i \in \mathbb{R}^d$, we can write $p_i = (p_i^1, p_i^2, \cdots, p_i^T)$, where $p_i^l \in \mathbb{R}^{d_l}$ $(1 \leq l \leq T)$. Let $P_l = \{p_i^l\}_{i \in [n]}$ be the data stored in the l-th party. With a slight abuse of notation, we also use P and P_l to denote their induced $n \times d$ and $n \times d_l$ matrices. Given a small integer $k \geq 1$, our goal is to find k cluster centers and a partition of P in the central server, such that the corresponding k-means clustering cost is minimized, and the communication complexity is as small as possible.

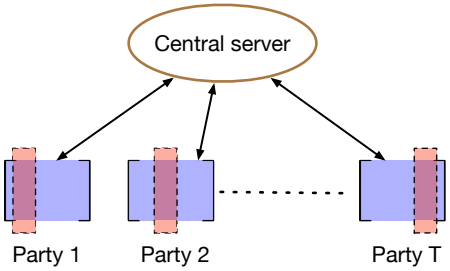


Figure 1. Our communication model. Suppose the input $n \times d$ matrix P is divided to T mutually disjoint subsets of columns, and each of the T parties reserves one individual subset; the communication is allowed only between the server and each party.

Communication model: We assume that there is a central server (or server in short) and only allow the communication between the central server and each of the T parties. We only require the central server to know the final solution. See Fig. 1 for an illustration. For simplicity, the communication cost are measured in *words*, i.e., sending one single number costs 1. The communication complexity/cost of an algorithm is the total number of words transmitted between the server and the parties. We also briefly discuss the communication complexity measured in bits in Section 7.

To capture the high volume and dimensionality of the data, we should think that n and d are very large, while k and T are much smaller in typical applications and treated as constants.

Note: To measure the quality of the clustering result, we compute the ratio between the obtained and minimum clustering costs, which is called *approximation ratio* (the closer to 1, the better).

1.2. Prior Work

We are not aware of any prior work explicitly focusing on clustering in distributed dimension setting (most existing algorithms for distributed clustering are for the row partition model, or some other purposes such as privacy preserving or making a consensus from different subspaces; the reader is referred to the end of this subsection for the details of these results), however, some can be easily adopted, which we briefly sketch below.

Random projection is a powerful technique for handling high dimensional data, and has been studied by (Boutsidis et al., 2015; Cohen et al., 2015) in the distributed clustering context. We can use random projection in the distributed dimension setting as follows: We multiply the sub-matrix P_l by a sub-matrix of the unified random projection matrix in the l-th party (see Fig. 2). Once the central server obtains $P \times R$ by receiving $P_l \times R_l$ from each party, it can perform any existing k-means clustering algorithm on it. The random projection ensures that the cost of k-means clustering is approximately preserved so that the performance guarantee of the distributed algorithm is almost the same as the centralized one.

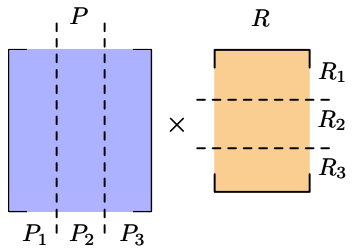


Figure 2. Matrices $P \in \mathbb{R}^{n \times d}$ and $R \in \mathbb{R}^{d \times t}$ represent the input data points and random projection matrix, respectively, where t indicates the reduced dimensionality depending on each individual method. In our setting of distributed dimensions, each party reserves a subset of columns of P, say P_l. The unified random projection matrix R can be independently generated in each party as long as they all use the same random seed, and the rows are divided into T groups with the indices consistent to the column partition on P. Then, each party computes the corresponding multiplication $P_l \times R_l$, and sends it to the central server. Finally, the central server obtains the summation $\sum_{l=1}^{T} P_l \times R_l$ which is equal to $P \times R$.

SVD can also be applied to k-means with distributed dimensions. There are two different ways to use SVD. **(1).** In the first approach, the server computes the orthonormal basis of a low dimensional subspace for approximating P (Feldman et al., 2013; Liang et al., 2014; Boutsidis et al., 2015; Cohen et al., 2015) (i.e., approximate PCA) by computing the SVD of P in the distributed setting[2], and

[2] Existing distributed SVD algorithms work for both the row partition and column partition models. We exchange the roles of n and d here due to column partition. So the basis now is a $n \times t$ matrix where t depends on each individual method.

Reference	Approximation ratio	Communication complexity
Johnson-Lindenstrauss (JL) transform$*$	$1+\epsilon$	$O(T\frac{\log n}{\epsilon^2}n)+kd$
(Boutsidis et al., 2015)†	$2+\epsilon$	$O(\frac{k}{\epsilon})(2Tn+d)$
(Boutsidis et al., 2015)‡	$3+\epsilon$	$O(\frac{k\log k}{\epsilon^2}n+\frac{k}{\epsilon}Tn+\frac{k}{\epsilon}d)$
(Boutsidis et al., 2015)$*$	$2+\epsilon$	$O(T\frac{k}{\epsilon^2}n)+kd$
(Liang et al., 2014; Cohen et al., 2015)†	$1+\epsilon$	$O(\frac{k}{\epsilon})(2Tn+d)$
(Cohen et al., 2015)‡	$1+\epsilon$	$O(\frac{k\log k}{\epsilon^2}n)+Tkn+kd$
(Cohen et al., 2015)$*$	$1+\epsilon$	$O(T\frac{k}{\epsilon^2}n)+kd$
(Cohen et al., 2015)$*$	$9+\epsilon$	$O(T\frac{\log k}{\epsilon^2}n)+kd$
Our results I	$9+\epsilon$	$Tn+kd$
Our results II	$5+\epsilon$	$2Tn+k^T\times d$ (or $3Tn+kd+k^T\times O(T\frac{k}{\epsilon^2})$ if $d\gg T\frac{k}{\epsilon^2}$).

Table 1. Existing and our new results for k-means clustering with distributed dimensions ($*$: random projection, †: PCA, ‡: feature selection), where $\epsilon>0$. The communication complexities for random projection and feature selection all contain an additive term kd. In fact, this is because once the central server obtains the clustering result in the space with reduced dimensionality, in order to map the k cluster centers back to the original space $\mathbb{R}^d$, each party should send the coordinate of each cluster center in its reserved subspace to the server, incurring an extra communication cost of $\sum_{l=1}^{T}kd_l=kd$.

sends the basis to all the parties. Then each party computes the (projected) coordinates of its own data in such a basis, which is a $t\times d_l$ matrix, and sends this matrix back to the server. Finally, the server reconstructs a rank-t approximation of P, and runs any existing k-means clustering algorithm on it (similar to the random projection based approaches). **(2).** The other way is called the *feature selection* method (Boutsidis et al., 2015; Cohen et al., 2015). The main difference from the first is that once obtaining the approximate SVD of P, the server generates a probability distribution over the d columns of P. Using the distribution, the servers selects t columns and inform all parties of the indices of the selected columns. Finally, each party sends the actual data of the selected columns to the server.

See Table 1 for the communication complexities of the above approaches. By closely examining the analysis of the previous dimension reduction approaches (including both random projection and SVD), we can see the approximation ratios are the same as those in the row partition models, but the communication complexities are different. The communication complexities listed in Table 1 can be easily obtained by adapting the previous algorithms to column partitions (as we sketched above), and we omit the details which are tedious and not particularly interesting.

Other Related Work. A number of results for clustering under centralized setting have been obtained in the past (Aggarwal & Reddy, 2013; Awasthi & Balcan, 2014). Particularly, k-means++ (Arthur & Vassilvitskii, 2007) and some of its variants are very efficient in practice and achieve the expected approximation ratio $O(\log k)$. Recently, (Chawla & Gionis, 2013) develops an algorithm for k-means with outliers. (Feldman et al., 2012; Zhang et al., 2008) apply *coresets* (Feldman & Langberg, 2011) to handle the problem of clustering with distributed data points (row partition). Following their works, (Balcan et al., 2013) develops the algorithm applicable to any connecting topology of communication networks, and also improves the communication cost. Several other algorithms for distributed data points are also proposed (Datta et al., 2006; Forman & Zhang, 2000; Bahmani et al., 2012). Some clustering results for distributed dimensions (Vaidya & Clifton, 2003; Jagannathan & Wright, 2005) mainly focus on privacy preserving rather than communication cost. Another similar problem is called *multi-view clustering* (Bickel & Scheffer, 2004) which focuses more on how to achieve a consensus of the multiple views, and the computation is usually executed in a single machine without concerning communication cost. Besides distributed clustering, many other machine learning problems are recently studied under distributed setting, such as PCA (Liang et al., 2014; Kannan et al., 2014), classification (Daumé III et al., 2012), PAC (Balcan et al., 2012), and sparse learning (Bellet et al., 2015).

1.3. Our Contributions

In this paper, we provide communication-efficient algorithms for k-means in the distributed dimension setting, improving upon the previous methods. In the random projection method, each party multiplies a sub-matrix R_l to its own sub-matrix P_l (as Fig. 2). R_l is chosen in a *data oblivious* way, without utilizing the property of each P_l. However, our method examines the distribution of P_l in each party, which allows us to spend less communication in our setting. In a high level, our algorithm works as follows. First, the l-th party computes a k-means clustering for P_l in the corresponding subspace $\mathbb{R}^{d_l}$, and sends to the server the obtained k cluster centers, as well as the *clustering memberships* of the n points.[3] Based on such information, the central server can approximate the point-set P by constructing a weighted grid in the whole space $\mathbb{R}^d$. Finally, one can apply any existing k-means clustering algorithm to the grid. When $k=O(1)$, we can achieve an approximation ratio of $9+\epsilon$, for any constant $\epsilon>0$, match-

[3] The clustering membership of a point p is the number in $[k]$ indexing the nearest neighbor of p among the k cluster centers.

ing those obtained using the random projection technique developed in (Cohen et al., 2015), but with a much lower communication complexity (Section 2). Another advantage of our algorithm is that it is very simple to implement and the computation effort for each party is also quite low.

By a more careful analysis of the distribution of each P_l, we show it is possible to achieve an improved $5+\epsilon$ approximation ratio with a slightly larger communication complexity (Section 3).

We show the comparison between the existing and our results for k-means clustering with distributed dimensions in Table 1. Note that the communication complexities from random projection and feature selection all have a hidden constant (usually bigger than 1) which depends on the success probability of the algorithms (due to their randomness). On the other hand, our communication protocol is deterministic, except that the centralized k-means algorithm (which is used as a black-box in each individual machine) may be randomized.

Furthermore, we point out another advantage of our method here. We can readily extend the method to some other related problems, such as *k-means with outliers* (Chawla & Gionis, 2013) and *constrained k-means clustering* (which has the same objective function as k-means but has extra combinatorial constraint) (Wagstaff et al., 2001; Ding & Xu, 2015; Bhattacharya et al., 2016), in the distributed dimension setting. We can achieve similar approximation ratios and communication complexities. The basic idea is that the built grid can be easily modified to adapt these problems, e.g., slightly increasing the size of the grid or sending more information besides the clustering memberships to the server. However, it is not clear to us how to adopt the existing approaches mentioned in Section 1.2 to these problems.

2. $(9+\epsilon)$-Approximation

In this section, we present a simple algorithm yielding a $(9+\epsilon)$-approximation for k-means clustering with distributed dimensions. The key idea is building a *weighted grid* in the original space $\mathbb{R}^d$ to approximately reconstruct the point-set P.[4] The reason for using a grid is that it can incur significant saving in communication complexity under the problem setting, *i.e.,* each party only needs to send a small amount of information to the central server for building such a grid. Moreover, in order to have a more accurate reconstruction, we assign a weight to each individual grid point. See Algorithm 1 for the details. In the algorithm, we denote $\underbrace{[k]\times[k]\times\cdots\times[k]}_{T}$ as $[k]^T$. We also denote the centralized algorithm for solving the k-means problem by $\mathcal{A}$, and assume that $\mathcal{A}$ can achieve an approximation ratio of λ (>1). For $k=O(1)$, it is known that there is a *PTAS* (i.e., polynomial-time approximation scheme) for k-means (Kumar et al., 2010; Feldman & Langberg, 2011) which indicates that the approximation ratio λ can be taken to be $1+\epsilon$ for any constant $\epsilon>0$. A toy example of the grid construction is shown in Fig. 3, and the theoretical guarantee is presented in the following Theorem 1.

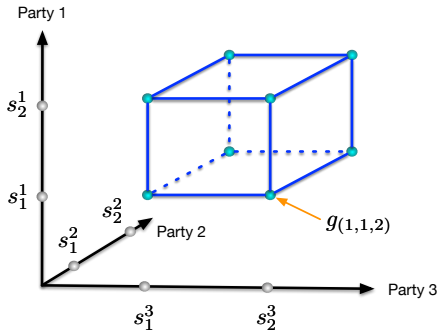


Figure 3. Let $n=6$, $k=2$, and $T=3$. Suppose the clustering results $\mathcal{M}^l$ for $l=1,2,3$ in the three parties are $\{\{p_1,p_2,p_3\},\{p_4,p_5,p_6\}\}$, $\{\{p_1,p_3\},\{p_2,p_4,p_5,p_6\}\}$, and $\{\{p_1,p_2,p_4,p_5\},\{p_3,p_6\}\}$ respectively. The cluster centers in each party are $\{s_1^l,s_2^l\}$. Then the grid built by Algorithm GRID is shown in the figure. For example, the grid point $g_{(1,1,2)}=(s_1^1,s_1^2,s_2^3)$. And since $\mathcal{M}_1^1\cap\mathcal{M}_1^2\cap\mathcal{M}_2^3=\{p_3\}$, the weight of $g_{(1,1,2)}$ is 1.

Theorem 1. *If we have a centralized algorithm $\mathcal{A}$ for k-means clustering which can achieve an approximation ratio λ,* DISTDIM-K-MEANS *(Algorithm 1) yields a $(2\lambda^2+3\lambda+2\lambda\sqrt{2(\lambda+1)})$-approximation with communication cost $Tn+kd$. In particular, the approximation ratio is $9+\epsilon$ if $\mathcal{A}$ is a PTAS.*

Proof. In DISTDIM-K-MEANS, only step 2 needs communication, and each party costs $n+kd_l$. Thus, the total communication cost is $Tn+k\sum_l d_l=Tn+kd$.

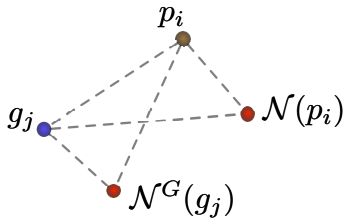


Figure 4. Illustration for the utilizations of triangle inequality in (1) and (4).

Below, we focus on the proof of the approximation guarantee. For simplicity, the grid G is rewritten as $\{g_1,\cdots,g_m\}$

[4] Our construction of the grid may seem to be similar to the *product quantization* (Jegou et al., 2011) which is used for nearest neighbor search in image retrieval. However, our approach is in fact fundamentally different, in that product quantization aims to find a partition on the d dimensions of the space such that the distortion of the built grid to the original point-set is minimized, while in our problem the partition is fixed as the input.

Algorithm 1 DISTDIM-K-MEANS

1. For each $l \in [T]$, the l-th party does the following: runs the centralized approximation algorithm $\mathcal{A}$ for k-means on P_l, and obtains the clustering memberships $\mathcal{M}^l = \{\mathcal{M}^l_1, \cdots, \mathcal{M}^l_k\}$ on $[n]$ (*i.e.,* each $\mathcal{M}^l_i \subset [n]$ and $\mathcal{M}^l_i \cap \mathcal{M}^l_{i'} = \emptyset$ if $i \neq i'$) and a set of corresponding cluster centers $\{s^l_1, \cdots, s^l_k\} \subset \mathbb{R}^{d_l}$.
2. For each $l \in [T]$, the l-th party sends the memberships $\mathcal{M}^l$ and the k cluster centers to the server.
3. The server builds a weighted grid G in $\mathbb{R}^d$ of size k^T, using Algorithm 2.
4. The server runs algorithm $\mathcal{A}$ on G, and obtains the k cluster centers $\{c^G_1, \cdots, c^G_k\}$ and the clustering memberships $\mathcal{M}^G = \{\mathcal{M}^G_1, \cdots, \mathcal{M}^G_k\}$ on $[k]^T$.
5. The server obtains the final clustering memberships $\mathcal{M} = \{\mathcal{M}_1, \cdots, \mathcal{M}_k\}$ on $[n]$, where each $\mathcal{M}_j = \cup\{i \in \bigcap_{l=1}^T \mathcal{M}^l_{i_l} \mid (i_1, \cdots, i_T) \in \mathcal{M}^G_j\}$.
6. Now, the server has the k cluster centers $\{c^G_1, \cdots, c^G_k\}$ and the clustering memberships $\mathcal{M} = \{\mathcal{M}_1, \cdots, \mathcal{M}_k\}$.

Algorithm 2 GRID

1. For each index tuple $(i_1, \cdots, i_T) \in [k]^T$, construct a point $g_{(i_1,\cdots,i_T)} = (s^1_{i_1}, s^2_{i_2}, \cdots, s^T_{i_T}) \in \mathbb{R}^d$.
2. Output $G = \{g_{(i_1,\cdots,i_T)} \mid (i_1, \cdots, i_T) \in [k]^T\}$, and each $g_{(i_1,\cdots,i_T)}$ is associated with the weight $|\bigcap_{l=1}^T \mathcal{M}^l_{i_l}|$.

where $m = k^T$, and for each g_j, its corresponding intersection $\bigcap_{l=1}^T \mathcal{M}^l_{i_l}$ is rewritten as S_j. Suppose the k cluster centers in an optimal solution are $\{c^\star_1, \cdots, c^\star_k\}$, and for each $p_i \in P$, we denote its nearest neighbor from $\{c^\star_1, \cdots, c^\star_k\}$ as $\mathcal{N}(p_i)$. Similarly, we denote the nearest neighbor of each $g_j \in G$ from $\{c^G_1, \cdots, c^G_k\}$ as $\mathcal{N}^G(g_j)$. Then, for each index $i \in S_j$,

$$\begin{aligned}\|p_i - \mathcal{N}^G(g_j)\|^2 &= \|p_i - g_j + g_j - \mathcal{N}^G(g_j)\|^2 \\ &\leq \|p_i - g_j\|^2 + \|g_j - \mathcal{N}^G(g_j)\|^2 \\ &\quad +2\|p_i - g_j\| \cdot \|g_j - \mathcal{N}^G(g_j)\|. \quad (1)\end{aligned}$$

See Fig. 4. For ease of analysis, we denote the cost of our solution from Algorithm 1, $\sum_{j=1}^{k^T} \sum_{i \in S_j} \|p_i - \mathcal{N}^G(g_j)\|^2$, by Γ, and the cost of the optimal solution, $\sum_{p_i} \|p_i - \mathcal{N}(p_i)\|^2$, by Γ_{opt}, respectively. Furthermore, $\sum_{j=1}^{k^T} \sum_{i \in S_j} \|p_i - g_j\|^2$ and $\sum_{j=1}^{k^T} |S_j| \|g_j - \mathcal{N}^G(g_j)\|^2$ are denoted by Γ_a and Γ_b, respectively. Summing both sides of (1), we have

$$\begin{aligned}\Gamma =& \sum_{j=1}^{k^T} \sum_{i \in S_j} \|p_i - \mathcal{N}^G(g_j)\|^2 \\ \leq& \Gamma_a + \Gamma_b + 2\sum_{j=1}^{k^T} \sum_{i \in S_j} \|p_i - g_j\| \cdot \|g_j - \mathcal{N}^G(g_j)\| \\ \leq& \Gamma_a + \Gamma_b + 2\sqrt{\Gamma_a \Gamma_b}, \quad (2)\end{aligned}$$

where the last inequality follows from *Cauchy-Schwarz inequality*. Note that $\{c^G_1, \cdots, c^G_k\}$ is a λ-approximation for the weighted point-set G, and each g_j has weight $|S_j|$. Actually each g_j can be viewed as $|S_j|$ unweighted points collocated at the same place. Then G can be viewed as a set of $\sum_j |S_j| = n$ unweighted points. Thus, we have

$$\begin{aligned}\Gamma_b &= \sum_{j=1}^{k^T} |S_j| \|g_j - \mathcal{N}^G(g_j)\|^2 \\ &\leq \lambda \sum_{j=1}^{k^T} \sum_{i \in S_j} \|g_j - \mathcal{N}(p_i)\|^2, \quad (3)\end{aligned}$$

where the left hand term Γ_b and the right hand term $\sum_{j=1}^{k^T} \sum_{i \in S_j} \|g_j - \mathcal{N}(p_i)\|^2$ indicate the costs of assigning the n points locating in G to the k clustering centers $\{c^G_1, \cdots, c^G_k\}$ and $\{c^\star_1, \cdots, c^\star_k\}$ respectively (note $\mathcal{N}^G(g_j) \in \{c^G_1, \cdots, c^G_k\}$ and $\mathcal{N}(p_i) \in \{c^\star_1, \cdots, c^\star_k\}$). Also, each

$$\|g_j - \mathcal{N}(p_i)\|^2 \leq 2\|g_j - p_i\|^2 + 2\|p_i - \mathcal{N}(p_i)\|^2 \quad (4)$$

by the relaxed triangle inequality (see Fig. 4). (3) and (4) together imply that

$$\Gamma_b \leq 2\lambda(\Gamma_a + \Gamma_{opt}). \quad (5)$$

Combining (2) and (5), we have

$$\Gamma \leq (2\lambda + 1)\Gamma_a + 2\lambda\Gamma_{opt} + 2\sqrt{2\lambda\Gamma_a(\Gamma_a + \Gamma_{opt})}. \quad (6)$$

Meanwhile, since the T subspaces among different parties are mutually orthogonal and disjoint (except the origin), the cost Γ_{opt} is the sum of the costs of their projections in the T subspaces. Also, in each subspace we run algorithm $\mathcal{A}$ which can achieve a factor of λ. Consequently, we have that

$$\begin{aligned}\Gamma_a &= \sum_{j=1}^{k^T} \sum_{i \in S_j} \|p_i - g_j\|^2 \\ &\leq \lambda \sum_{p_i} \|p_i - \mathcal{N}(p_i)\|^2 = \lambda\Gamma_{opt}. \quad (7)\end{aligned}$$

Plugging (7) into (6), we have

$$\Gamma \leq (2\lambda^2 + 3\lambda + 2\lambda\sqrt{2(\lambda + 1)})\Gamma_{opt}. \quad (8)$$

Since there is a PTAS for k-means when $k = O(1)$, we can take $\lambda = 1 + O(\epsilon^2)$, and obtain the approximation ratio of $9 + \epsilon$ for our problem. □

3. $(5+\epsilon)$-Approximation

Here, we follow the idea of DISTDIM-K-MEANS but modify the construction of G in GRID (Algorithm 2), to improve the approximation ratio from $9+\epsilon$ to $5+\epsilon$. We also show how to reduce the following increased communication complexity via the idea of random projection by (Cohen et al., 2015) in Section 3.1.

DISTDIM-K-MEANS-IMPROVED: In step 2 of DISTDIM-K-MEANS, we do not need to send the coordinates of the k centers from each party to the server; instead, in the GRID algorithm, we update each $g_{(i_1,\cdots,i_T)}$ by the mean point of $\{p_i \mid i \in \bigcap_{l=1}^{T} \mathcal{M}_{i_l}^l\}$. To achieve that, the server broadcasts the index-set $S_j = \bigcap_{l=1}^{T} \mathcal{M}_{i_l}^l$ to each party, and then each party sends $\frac{1}{|S_j|}\sum_{i\in S_j} p_i^l$, *i.e.,* the projection of the mean point on its subspace back to the server. Finally, through simple concatenating the server can obtain $\frac{1}{|S_j|}\sum_{i\in S_j} p_i$, *i.e.,* the mean point.

We consider the extra communication cost caused by this modification. Since each index from $[n]$ belongs one and only one of the k^T index-sets, broadcasting all these index-sets to the T parties cost a total communication complexity of Tn. Secondly, it costs d for constructing each individual mean point of $\{p_i \mid i \in \bigcap_{l=1}^{T} \mathcal{M}_{i_l}^l\}$, and thus $k^T d$ for all the k^T mean points. Note that our method only benefits the case of $n > k^T$, otherwise, we just simply send all the data to the server which costs a communication complexity of nd. Overall, the modification needs an extra communication cost of $Tn + k^T d$.

Theorem 2. *If we have a centralized algorithm $\mathcal{A}$ for k-means clustering which can achieve an approximation ratio λ,* DISTDIM-K-MEANS-IMPROVED *yields a $(2\lambda^2+3\lambda)$-approximation with communication cost $2Tn + k^T d$. The approximation ratio can be $5+\epsilon$ if $\mathcal{A}$ is a PTAS.*

Proof. From the above analysis, we can easily obtain the communication complexity of $2Tn + k^T d$. For the approximation ratio, we recall the proof of Theorem 1. At first, we need the well known fact below.

Fact 1. *Let q^* be the mean of a point-set $Q \subset \mathbb{R}^d$, then for any arbitrary point $q \in \mathbb{R}^d$, we have*

$$\sum_{q_i\in Q} \|q - q_i\|^2 = \sum_{q_i\in Q} \|q^* - q_i\|^2 + |Q|\|q - q^*\|^2. \quad (9)$$

Based on the above (9) and the fact that g_j is the mean of $\{p_i \mid i \in S_j\}$, we rewrite the cost of our solution

$$\begin{aligned}\Gamma &= \sum_{j=1}^{k^T}\sum_{i\in S_j} \|p_i - \mathcal{N}^G(g_j)\|^2 \\ &= \sum_{j=1}^{k^T}(\sum_{i\in S_j} \|p_i - g_j\|^2 + |S_j|\|g_j - \mathcal{N}^G(g_j)\|^2) \\ &= \Gamma_a + \Gamma_b \quad (10)\end{aligned}$$

Also, with respect to the value of Γ_a, since we change the position of each g_j to the mean point of $\{p_i \mid j \in S_j\}$, the value of Γ_a cannot increase. Furthermore, following the same manner of (3)-(7), we have that $\Gamma \le (2\lambda^2+3\lambda)\Gamma_{opt}$. Finally, the approximation ratio can reach $5+\epsilon$ if $\lambda = 1 + O(\epsilon)$. □

3.1. Improvement via Random Projection

(Cohen et al., 2015) shows that given an instance of k-means clustering, one can project all the points to a randomly selected $O(k/\epsilon^2)$-dimensional space, such that when recovering the solution in the original space, the approximation ratio increases by a factor at most $(1+\epsilon)$. In this section, we show how to apply this result to improve the communication complexity of the $(5+\epsilon)$-approximation.

One expensive step of DISTDIM-K-MEANS-IMPROVED is computing the exact coordinates of the grid points $\{g_1, \cdots, g_{k^T}\}$ in $\mathbb{R}^d$, which costs a communication complexity of $k^T d$. From (Cohen et al., 2015), we know that it is not necessary to do that. Actually, it is sufficient to only know their coordinates in $\mathbb{R}^{O(k/\epsilon^2)}$ after random projection. To realize that, we can use exactly the same idea showed in Fig. 2. As a consequence, each party sends a $k^T \times O(k/\epsilon^2)$ matrix to the server, and the communication complexity of this step is reduced from $k^T d$ to $O(T\frac{k^{T+1}}{\epsilon^2})$ (suppose $d \gg Tk/\epsilon^2$). In addition, after finding the clustering memberships of P in $\mathbb{R}^{O(k/\epsilon^2)}$, we still need to compute the coordinates of the k cluster centers in $\mathbb{R}^d$, which costs an extra $Tn + kd$ in communication complexity.

For the final approximation ratio, we can slightly reduce the value of ϵ used in the random projection and PTAS of centralized k-means clustering by a constant factor, such that the ratio of $5+\epsilon$ in Theorem 2 can still be guaranteed.

Theorem 3. *Through random projection, the communication complexity of* DISTDIM-K-MEANS-IMPROVED *is reduced to $3Tn + kd + O(T\frac{k^{T+1}}{\epsilon^2})$ and the same approximation ratio is preserved.*

4. Other Results

We extend our DISTDIM-K-MEANS algorithm to some other related clustering problems with distributed dimensions. In general, we consider two problems motivated by the particular real-world applications, k-means clustering with outliers and constrained k-means clustering. To the best of our knowledge, there is no obvious way to extend the previous mentioned approaches, such as random projection or SVD, to these problems.

4.1. k-Means Clustering with Outliers

Comparing to the ordinary k-means clustering, its variant of that considering outliers is more general and robust in practice. Using the same notations defined before and an

additional integer parameter $z < n$, the object is to kick out z points from P such that the objective value of k-means clustering for the remaining points is as small as possible. For this problem, we still follow the idea and main structure of DISTDIM-K-MEANS. Below is our modification.

DISTDIM-K-MEANS-OUTLIERS. We only need to modify a few places in DISTDIM-K-MEANS. At first, similar to k-means clustering, we assume that $\mathcal{B}$ is the centralized algorithm solving k-means with outliers and achieving an approximation of $\lambda > 1$. In step 1, we run $\mathcal{B}$. Moreover, the corresponding memberships $\mathcal{M}^l$ become $\{\mathcal{M}^l_1, \cdots, \mathcal{M}^l_{k+z}\}$ where the last z items indicates the indices of the z outliers. Also, in step 2 each party sends the corresponding memberships and coordinates of not only the k centers but also the z outliers. Consequently, the weighted grid G has a size of $(k+z)^T$. Similarly, we run $\mathcal{B}$ and compute the memberships in step 4 & 5. Note that when running $\mathcal{B}$ in step 4, we view each $g_{(i_1,\cdots,i_T)}$ as a multi-set of $|\bigcap_{l=1}^T \mathcal{M}^l_{i_l}|$ individual unweighted points. Finally, in step 5 we obtain the clustering memberships $\mathcal{M}$ on the n points except z outliers.

The theoretical guarantee is presented in the following Theorem 4. The proof is similar to that of Theorem 1, and due to space limit we put it in our supplement.

Theorem 4. *If we have a centralized algorithm $\mathcal{B}$ for k-means clustering with outliers which can achieve an approximation ratio λ,* DISTDIM-K-MEANS-OUTLIERS *yields a $(2\lambda^2 + 3\lambda + 2\lambda\sqrt{2(\lambda+1)})$-approximation with communication cost $Tn + (k+z)d$. In particular, the approximation ratio is $9+\epsilon$ if $\mathcal{B}$ is a PTAS.*

Remark 1. *Different from k-means, the only known PTAS for k-means with outliers (for the centralized setting) (Feldman & Schulman, 2012) requires the number of outliers to be a constant. For those applications where we need to exclude more outliers, we can instead run some heuristic algorithms such as (Chawla & Gionis, 2013).*

4.2. Constrained k-Means Clustering

Given an instance of k-means clustering, many real-world applications do not allow the set of points to be arbitrarily assigned to the clusters. The constraints, such as upper or lower bound on the size of each cluster and chromatic conflict (e.g., the points sharing the same color cannot be clustered into the same cluster), are motivated from data mining (Wagstaff et al., 2001), resource assignment (Khuller & Sussmann, 1996), and privacy preserving (Sweeney, 2002; Aggarwal et al., 2010; Li et al., 2010). Recently, (Ding & Xu, 2015; Bhattacharya et al., 2016) provide PTAS for a large class of constrained k-means clustering when $k = O(1)$.

Actually, some of the constrained k-means clustering problems under the setting of distributed dimensions can be easily solved by the framework of our DISTDIM-K-MEANS in Section 2 (just replace the algorithm $\mathcal{A}$ by the corresponding algorithms handling the constraints, such as the PTAS by (Ding & Xu, 2015; Bhattacharya et al., 2016), in the server and each party). To shed some light on this, consider k-means clustering with a lower bound $r > 0$ on the size of each cluster. Once building the grid G, we can still bound the distortion between G and the original point-set P by the similar way of (7), because the cost of the objective function (even after adding lower bound constraint) is always equal to the sum of the costs in the mutually orthogonal and disjoint subspaces due to the nature of k-means. Then through the manner of (2)-(6), we can obtain the same approximation ratio and communication complexity. Moreover, for item-level constraints such as chromatic conflict, we can view each grid point of G as a set of individual colored points locating at the same location, and consequently G is in fact a new instance of k-means clustering with chromatic conflict but having a bounded distortion to the original instance P; finally the theoretical results can be similarly guaranteed as well.

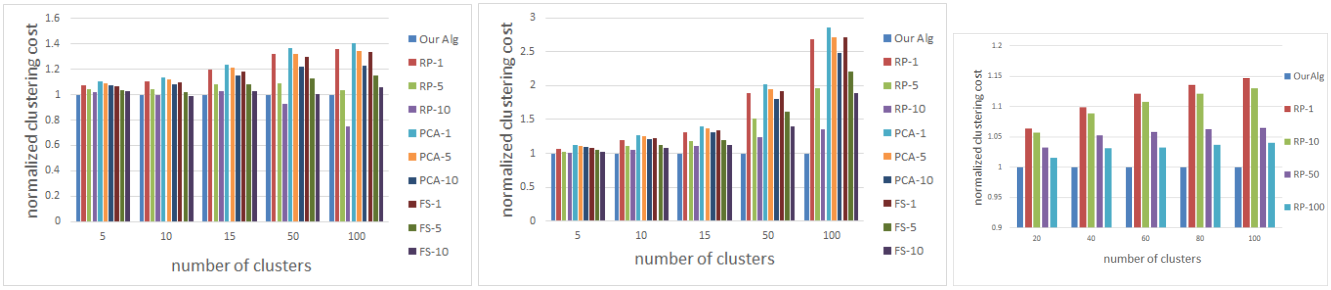
5. Lower Bound

In this section, we provide a lower bound of the communication cost for k-means problem with distributed dimensions. In fact, the lower bound even holds for the special case where the l-th party holds the l-th column. We denote by $k\text{-}Means_{n,T}$ the problem where there are T parties and n points in $\mathbb{R}^T$, and we want to compute k-means in the server. We prove a lower bound of $\Omega(n \cdot T)$ for $k\text{-}Means_{n,T}$ (for achieving any finite approximation ratio) by a reduction from the set disjointness problem (Chattopadhyay & Pitassi, 2010). Due to space limit, we put the details in our supplement.

6. Experiments

We compare our algorithms of DISTDIM-K-MEANS and DISTDIM-K-MEANS-OUTLIERS to the following three types of methods as mentioned in Section 1.2: (1) the random projection based approach RP-t where t is the dimensionality of the random subspace, and the two SVD based approaches (2) PCA-t where t is the rank of the reconstructed matrix approximating P and (3) FS-t where t is the number of selected features (dimensions).

Data-sets & Experimental methodology. We first choose a real-world data-set from (Bache & Lichman, 2013), *YearPredictionMSD* which contains 10^5 points in $\mathbb{R}^{90}$. We randomly divide the data-set into 3 parties with each having 30 attributes (i.e., $T = 3$), and set $k =$ 5-100. Also, for k-means clustering with outliers we set the number of outliers $z = 500$. In the server and each party, we use the algorithms from (Arthur & Vassilvitskii, 2007) and (Chawla & Gionis, 2013) as the centralized algorithms $\mathcal{A}$ and $\mathcal{B}$, i.e., the subroutines in DISTDIM-K-MEANS and DISTDIM-K-



(a) (b) (c)

Figure 5. The normalized costs of k-means clustering on (a) YearPredicitonMSD, (b)YearPredicitonMSD considering outliers, and (c) Bag of Words(NYTimes).

	Our Alg	RP			PCA			FS		
		1	5	10	1	5	10	1	5	10
KMeans	6.02	8.87	32.30	61.59	14.74	61.62	120.20	3.99	7.89	12.78
Outlier	7.32	9.76	33.19	62.48	15.63	62.51	121.09	4.88	8.78	13.67

Table 2. Average communication costs for YearPredictionMSD data-set (in100KB) over the values of k.

MEANS-OUTLIERS, respectively. Furthermore, in order to show the advantage of our approach in high dimension, we also implement our algorithm DISTDIM-K-MEANS on another data-set *Bag of Words(NYTimes)* from (Bache & Lichman, 2013) which contains 10^5 points in $\mathbb{R}^{102660}$. We randomly divide it into 100 parties with each having about 1000 attributes (i.e., $T = 100$), set $k =$ 20-100, and compare the results to the random projection based approach.[5]

Results & Analysis. Firstly, comparing to the clustering costs obtained by the centralized algorithms $\mathcal{A}$ and $\mathcal{B}$ (directly run on the whole data-sets without considering the distributed setting) for different values of k, our results obtained in the distributed setting are over within a factor of no more than 1.5 for most of the cases, which is much better than the theoretical ratio $9 + \epsilon$ by Theorem 1&4.

Secondly, we show the comparison on clustering cost between our and other algorithms in Fig. 5, where we normalized all the costs by the corresponding costs obtained by our approach (hence the ratios of our results are always 1). We also list the corresponding communication costs for the two data-sets in Table 2&3 separately (due to space limit, we only show the average costs over k in Table 2). For the first data-set YearPredictionMSD without considering outliers (Fig. 5(a)), the clustering costs are roughly the same when k is small, but our approach outperforms most of them when k increases to 50 and 100 (the only one beats ours is RP-10 which has about 10 times communication cost of ours as shown in Table 2). Further, for the same data-set but considering outliers (Fig. 5(b)), the advantage of our approach is much more significant which indicates that our algorithm is more efficient for handling outliers. For the second data-set Bag of Words(NYTimes) with a much higher dimension (Fig. 5(c)), our algorithm also outperforms the random projection based approach on both clustering cost and communication cost in most cases.

[5]Considering the size of Bag of Words(NYTimes), the SVD based approaches are much more impractical than the random projection based approach via current techniques in reality.

K	Our Alg	RP			
		1	10	50	100
20	27.54	33.30	209.08	990.33	1966.89
40	35.55	37.30	213.09	994.34	1970.90
60	43.55	41.31	217.09	998.34	1974.90
80	51.56	45.31	221.09	1002.35	1978.91
100	59.57	49.32	225.10	1006.36	1982.91

Table 3. Communication costs for NYTimes data-set (in MB).

7. The bit complexity

As mentioned in Section 1.1, we assume that sending one single number costs a communication complexity of 1. In terms of the bit complexity, encoding the membership for each point needs $\log k$ bits. Thus, the bit communication complexities of the two $(9 + \epsilon)$-approximation algorithms by (Cohen et al., 2015) and us should be $O(T\frac{\log k}{\epsilon^2})nL + kdL$ and $Tn \log k + kdL$, respectively, where L is the word length in the machine. We note that even in this case, our improvement is significant.

8. Future Work

Following our approach, several interesting problems deserve to be explored. For example, how to achieve a similar approximation ratio and communication complexity for k-median/center clustering problems. To generalize our method to the setting of arbitrary partition (such as (Kannan et al., 2014) for PCA) is also challenging but useful in some scenarios.

Acknowledgements

Ding was supported by a start-up fund from Michigan State University. Part of the work was done when Ding was in IIIS, Tsinghua University and Simons Institute, UC Berkeley. Liu, Huang, and Li were supported in part by the National Basic Research Program of China grants 2015CB358700, 2011CBA00300, 2011CBA00301, and the National NSFC grants 61033001, 61361136003.

References

Abadi, Daniel, Boncz, Peter, and Harizopoulos, Stavros. The design and implementation of modern column-oriented database systems. 2013.

Abadi, Daniel J, Boncz, Peter A, and Harizopoulos, Stavros. Column-oriented database systems. *Proceedings of the VLDB Endowment*, 2(2):1664–1665, 2009.

Aggarwal, Charu C and Reddy, Chandan K. *Data clustering: algorithms and applications*. CRC Press, 2013.

Aggarwal, Gagan, Panigrahy, Rina, Feder, Tomás, Thomas, Dilys, Kenthapadi, Krishnaram, Khuller, Samir, and Zhu, An. Achieving anonymity via clustering. *ACM Transactions on Algorithms (TALG)*, 6(3):49, 2010.

Arthur, David and Vassilvitskii, Sergei. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pp. 1027–1035, 2007.

Awasthi, Pranjal and Balcan, Maria-Florina. Center based clustering: A foundational perspective. 2014.

Bache, K. and Lichman, M. Uci machine learning repository. 2013.

Bahmani, Bahman, Moseley, Benjamin, Vattani, Andrea, Kumar, Ravi, and Vassilvitskii, Sergei. Scalable k-means++. *Proceedings of the VLDB Endowment*, 5(7): 622–633, 2012.

Balcan, Maria-Florina, Blum, Avrim, Fine, Shai, and Mansour, Yishay. Distributed learning, communication complexity and privacy. In *COLT 2012 - The 25th Annual Conference on Learning Theory, 2012, Edinburgh, Scotland*, pp. 26.1–26.22, 2012.

Balcan, Maria-Florina F, Ehrlich, Steven, and Liang, Yingyu. Distributed k-means and k-median clustering on general topologies. In *Advances in Neural Information Processing Systems 26*, pp. 1995–2003. Curran Associates, Inc., 2013.

Bellet, Aurélien, Liang, Yingyu, Garakani, Alireza Bagheri, Balcan, Maria-Florina, and Sha, Fei. A distributed frank-wolfe algorithm for communication-efficient sparse learning. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Canada, 2015*, pp. 478–486, 2015.

Bhattacharya, Anup, Jaiswal, Ragesh, and Kumar, Amit. Faster algorithms for the constrained k-means problem. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, pp. 16:1–16:13, 2016.

Bickel, Steffen and Scheffer, Tobias. Multi-view clustering. In *Data Mining, 2004. ICDM'04. Fourth IEEE International Conference on*, pp. 19–26. IEEE, 2004.

Boutsidis, Christos, Zouzias, Anastasios, Mahoney, Michael W, and Drineas, Petros. Randomized dimensionality reduction for-means clustering. *Information Theory, IEEE Transactions on*, 61(2):1045–1062, 2015.

Chattopadhyay, Arkadev and Pitassi, Toniann. The story of set disjointness. *ACM SIGACT News*, 41(3):59–85, 2010.

Chawla, Sanjay and Gionis, Aristides. k-means-: A unified approach to clustering and outlier detection. In *Proceedings of the 13th SIAM International Conference on Data Mining, 2013.USA.*, pp. 189–197, 2013.

Cohen, Michael B., Elder, Sam, Musco, Cameron, Musco, Christopher, and Persu, Madalina. Dimensionality reduction for k-means clustering and low rank approximation. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, STOC '15, pp. 163–172, 2015.

Datta, Souptik, Giannella, Chris, and Kargupta, Hillol. K-means clustering over a large, dynamic network*. In *Society for Industrial and Applied Mathematics. Proceedings of the SIAM International Conference on Data Mining*, pp. 153, 2006.

Daumé III, Hal, Phillips, Jeff M, Saha, Avishek, and Venkatasubramanian, Suresh. Efficient protocols for distributed classification and optimization. In *Algorithmic Learning Theory*, pp. 154–168. Springer, 2012.

Ding, Hu and Xu, Jinhui. A unified framework for clustering constrained data without locality property. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, USA, 2015*, pp. 1471–1490, 2015.

Feldman, Dan and Langberg, Michael. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pp. 569–578, 2011.

Feldman, Dan and Schulman, Leonard J. Data reduction for weighted and outlier-resistant clustering. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pp. 1343–1354, 2012.

Feldman, Dan, Sugaya, Andrew, and Rus, Daniela. An effective coreset compression algorithm for large scale sensor networks. In *Information Processing in Sensor Networks (IPSN), 2012 ACM/IEEE 11th International Conference on*, pp. 257–268, 2012.

Feldman, Dan, Schmidt, Melanie, and Sohler, Christian. Turning big data into tiny data: Constant-size coresets for *k*-means, PCA and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013,USA*, pp. 1434–1453, 2013.

Forman, George and Zhang, Bin. Distributed data clustering can be efficient and exact. *ACM SIGKDD explorations newsletter*, 2(2):34–38, 2000.

Gu, Lin, Jia, Dong, Vicaire, Pascal, Yan, Ting, Luo, Liqian, Tirumala, Ajay, Cao, Qing, He, Tian, Stankovic, John A., Abdelzaher, Tarek F., and Krogh, Bruce H. Lightweight detection and classification for wireless sensor networks in realistic environments. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, SenSys 2005, USA*, pp. 205–217, 2005.

Jagannathan, Geetha and Wright, Rebecca N. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pp. 593–599, 2005.

Jegou, Herve, Douze, Matthijs, and Schmid, Cordelia. Product quantization for nearest neighbor search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1):117–128, 2011.

Kannan, Ravi, Vempala, Santosh, and Woodruff, David P. Principal component analysis and higher correlations for distributed data. In *Proceedings of The 27th Conference on Learning Theory, COLT 2014, Barcelona, Spain, June 13-15, 2014*, pp. 1040–1057, 2014.

Khuller, Samir and Sussmann, Yoram J. The capacitated k-center problem. In *AlgorithmsESA'96*, pp. 152–166. Springer, 1996.

Kumar, Amit, Sabharwal, Yogish, and Sen, Sandeep. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2), 2010.

Li, Jian, Yi, Ke, and Zhang, Qin. Clustering with diversity. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010*, pp. 188–200, 2010.

Liang, Yingyu, Balcan, Maria-Florina F, Kanchanapally, Vandana, and Woodruff, David. Improved distributed principal component analysis. In *Advances in Neural Information Processing Systems*, pp. 3113–3121, 2014.

Stonebraker, Mike, Abadi, Daniel J, Batkin, Adam, Chen, Xuedong, Cherniack, Mitch, Ferreira, Miguel, Lau, Edmond, Lin, Amerson, Madden, Sam, O'Neil, Elizabeth, et al. C-store: a column-oriented dbms. In *Proceedings of the 31st international conference on Very large data bases*, pp. 553–564. VLDB Endowment, 2005.

Su, Lu, Gao, Jing, Yang, Yong, Abdelzaher, Tarek F., Ding, Bolin, and Han, Jiawei. Hierarchical aggregate classification with limited supervision for data reduction in wireless sensor networks. In *Proceedings of the 9th International Conference on Embedded Networked Sensor Systems, SenSys 2011, USA*, pp. 40–53, 2011.

Sweeney, Latanya. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

Vaidya, Jaideep and Clifton, Chris. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pp. 206–215, 2003.

Wagstaff, Kiri, Cardie, Claire, Rogers, Seth, and Schroedl, Stefan. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 577–584, 2001.

Zhang, Qi, Liu, Jinze, and Wang, Wei. Approximate clustering on distributed data streams. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pp. 1131–1139, 2008.