Non-uniform Sampling in Clustering and Streaming

Dissertation

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
der Technischen Universität Dortmund
an der Fakultät für Informatik

von

Morteza Monemizadeh

Dortmund

2010

Tag der mündlichen Prüfung:	29. September 2010
Dekan:	Prof. Dr. Peter Buchholz
Gutachter:	Prof. Dr. Christian Sohler
	Prof. Dr. Nicole Schweikardt

Abstract

Approximating a sum without computing the summands is a classic problem in statistics and machine learning. The problem is defined as follows: Assume Z is the sum of n numbers, $Z_1, \cdots, Z_n$ i.e., $Z = Z_1 + \cdots + Z_n$. The goal is to estimate Z without computing all the n summands but few.

According to the uniform sampling we choose a number Z_i with probability $\frac{1}{n}$ and assign the weight n to Z_i. The number nZ_i will be our estimation for Z. We see that the expectation of the random variable nZ_i is Z but the variance of nZ_i can be large. The reason is if the number of large numbers is few, then the probability that the random sample does not take one of them will be high and if this happens the variance of nZ_i will be large.

Using non-uniform sampling we can bound the variance in terms of the expectation and therefore estimate Z within factor $(1 \pm \epsilon)$ as follows: Having n probabilities $r_i \geq \frac{1}{\gamma}\frac{Z_i}{Z}$ for $1 \leq i \leq n$ corresponding to the numbers $Z_1, \cdots, Z_n$ we take a sample set $A = \{a_1, \cdots, a_j, \cdots, a_s\} \subseteq [n]$ of indices according to the probabilities r_i and assign a weight of $w(Z_{a_j}) = \frac{1}{s \cdot r_{a_j}}$ to a sampled number Z_{a_j} for $1 \leq j \leq s$. We then use the concentration bounds to show that for $s = O(\gamma^2 \epsilon^{-2} \log(1/\delta))$ the probability that the estimator $X = \sum_{a_j \in A} w(Z_{a_j}) \cdot Z_{a_j}$ deviates from Z by more than ϵZ is at most δ.

In this thesis we study applications of this estimator in high dimensional clustering and streaming. In particular, for the k-means and the j-subspace problems we get unbiased estimators that can $(1 \pm \epsilon)$-approximate the cost of the point set to an arbitrary center set. We then use these estimators to get coresets, linear time $(1 + \epsilon)$-approximation and insertion only streaming algorithms.

In the turnstile streaming model we are given a vector a of length n where the i-th coordinate is represented by a_i and a stream $\mathcal{S}$ as $m = \text{poly}(n, M)$ updates of the form (i, x), where $i \in [n]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$, indicating that the i-th coordinate a_i of a should be incremented by x. Let $Z_i = |a_i|^p$ for $p \in [0, 2], 1 \leq i \leq n$ and $Z = F_p(a) = \sum_{i=1}^{n} |a_i|^p$. In this model finding n probabilities $r_i \geq \frac{1}{\gamma}\frac{Z_i}{Z}$ using one pass and polylog space was known to be an open problem in the streaming community [CMI05]. We give a 1-pass $\text{poly}(\epsilon^{-1} \log n)$-space algorithm called L_p-*sampler* that samples according to probabilities r_i for $\gamma = (1 \pm \epsilon), p \in [0, 2]$. We show that the L_p-sampler leads to many improvements and a unification of well-studied streaming problems, including cascaded norms, heavy hitters, and moment estimation. In particular, as for the moment estimation using $O(n^{1-2/k}\epsilon^{-2})$ L_2-samplers in parallel for $k > 2$ we can $(1 \pm \epsilon)$-estimate $F_k(a) = \sum_{i=1}^{n} |a_i|^k$ using optimal space $n^{1-2/k} \cdot \text{poly}(\epsilon^{-1} \log n)$. This algorithm is the first that does not use Nisan's pseudorandom generator as a subroutine, potentially making it more practical.

Credit

The results of this thesis which were obtained in cooperation with my coauthors were published in

- Dan Feldman, Morteza Monemizadeh, and Christian Sohler, A PTAS for k-Means Clustering Based on Weak Coresets, Proceedings of the 23rd Annual Symposium on Computational Geometry (SoCG), pages: 11-18, 2007.
- Dan Feldman, Morteza Monemizadeh, Christian Sohler, and David Woodruff, Coresets and Sketches for High Dimensional Subspace Approximation Problems, Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA) , pages: 630-649, 2010.
- Morteza Monemizadeh and David Woodruff, 1-Pass Relative-Error L_p-Sampling with Applications, Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages: 1143-1160, 2010.

We should mention that the authors are listed lexicographically as is usual in computer science.

Acknowledgements

When I was applying for phd, my MS advisor told me that doing PhD in Germany might not be a good idea. But I told him that I read Christian Sohler's papers on coresets and I enjoyed. I would like to move there and I will try the best. After 4 years and 4 months working with Christian I am happy to say that I didn't make a wrong decision. Working with Christian on problems is fun, especially problems in high-dimensional geometry where for more than 1000 times you think your idea should work now, but since your imagination has been adapted for two or three dimensions, you'd be better take Christian's advice to sit down for a while and go through all proofs one by one to see whether your idea is working or like almost always wrong. He gave me this opportunity to invite people from all around the world and work with them.

Christian, I learned a lot from you, thank you!

I would like to thank my co-authors, Dan Feldman, David Woodruff, Christiane Lammersen, Krzysztof Onak, and Mohammad Abam. My special thanks to David Woodruff. In Dagstuhl I met him for the first time. On that time he was working on the problem of L_p-sampler. I asked him to let me work on this problem and he accepted that. By working on this problem I learned a lot about turnstile streaming model. David is always eager to work on new problems even at a very noisy Coffee shop at mid night or at a luxury restaurant where you pay more than 40 euros but you both still think you can have two more dishes and go to a Donner kebab to eat a lot for just 3 euros and think that Donner was better ;-)

During my phd in Germany I moved to three different universities in Paderborn, Bonn, and Dortmund. It was impossible to figure out how to go to different amts and offices in new cities to renew your visa, to enroll at a university without help of friends, colleagues and office-mates. My special thanks to my best friend in Germany, Christiane. I can not imagine to live in Germany without her help to translate more than 100 enrollment forms. I would like to thank Gereon, Bastian, Thomas, Alex, Florian, Melanie, Frank, Mohammad, Christine, and Chirstian, and Renate.

My last and the best thanks to my Mom and Dad for all the things that they gave me in my life. I owe you more than I can imagine.

To my wife Samira
for giving me the opportunity to believe.

Contents

Chapter 1

Introduction

Uniform sampling is a classic method in computer science to reduce the complexity of the input data of an algorithm. It is usually used when the size of the input of our algorithm is large and the time complexity of the algorithm is polynomially dependent on this size. In this setting running the algorithm on the input will be very time consuming and we need to figure out how to reduce the running time of the algorithm which might be a very challenging task.

Uniform sampling is a way to detour this problem by reducing the size of the input. First we sample a small subset of the input uniformly at random. Then we run our algorithm on this small subset. At the end we prove that the solution for the sampled set is a fairly good approximation for the entire input.

Let us consider the k-median problem in the 2-dimensional Euclidean plane as an example. In this problem, we are given a set P of n points in the plane where the diameter of P is upper bounded by some function of $\sqrt{n}$. Such an upper bound on the diameter of P is necessary in our setting. The goal is to report k points so-called centers in this plane such that the sum of distances of points in P to their nearest centers is minimized. The fastest approximation algorithm for the k-median due to Jain and Vazirani [JV01] has running time $O(n^2 \log n)$. However even this fast algorithm for point sets of size n when n is very large is very slow and remains only theoretically interesting not practically.

Ideally we would like to have an algorithm with running time linear in n. This is due to the fact that any algorithm for this problem needs to read the whole point set and bring it to the memory which takes $\Omega(n)$. Uniform sampling is a way to achieve a running time very near to this time. We take a sample set $S \subseteq P$ of size $|S| = O(\sqrt{n})$ u.a.r and run Jain and Vazirani's algorithm on S to report a set C of k centers. We then show that C is a fairly good solution for the point set P. Since $|S| = O(\sqrt{n})$ and the running time of the approximation algorithm is $O(n^2 \log n)$, the total running time will be $O(n \log n)$.

One issue with considering uniform random sampling as a ubiquitous tool for massive data sets is that in some scenarios a small subset of the data set carries more information than the rest. Since this subset is small if we take a uniform sampling over the data set with high probability we do not hit this subset. Therefore the solution of the sampled set is not a good solution for the entire data set.

In ***this thesis*** we look at *non-uniform sampling* as our candidate methodology for such scenarios. First, we assign higher probabilities to the more important elements (i.e., those which carry more information) and lower probabilities to the rest and then sample the data according to these probabilities. In this way, we can prove that with a reasonable probability there are enough samples from the more important elements inside the sample set; therefore the solution of the sample set is a good solution for the entire data set.

Next we explore a typical example of massive data set scenarios where we need to take non-uniform sampling.

1.1 A massive data set scenario

Massive data set is the broad class of applications where as input we have a huge amount of data or objects each one being characterized by many features. Some applications where we see massive data sets are:

- *WWW* [PRTV00], where objects are web documents and features are terms in the web documents,
- *Click-stream network* [FKK+00], where objects are web users and features are URLs that one user follows by clicking hypertexts,
- *Citation networks* [New04], where objects are authors and features are typical keywords in their publications,
- *Ecological networks* [AB02], where objects are species and features are predator-prey relationships,
- *Biological networks* [BO04], where objects are people and features are genes,
- *Recommendation Systems* [DKR02], where objects are customers and features are goods.

According to classical information retrieval we can represent each one of these examples as an *object-feature matrix*, where rows correspond to objects and columns correspond to features. In Figure 1.1, left picture we see a document-term matrix built upon a small part of WWW crawled and studied in [CKVW06]. As we observe this matrix is very sparse and seems to be meaningless, a phenomenon that happens very often in massive data set scenarios. By looking at such huge object-feature matrices the following natural questions come into our mind:

(1) How can we extract some well-structured patterns from an object-feature matrix? And how can we cluster objects into subsets (clusters) according to these extracted features?

(2) If we do not have enough space to store a whole object-feature matrix, can we find a compact representation of the matrix and then extract patterns by just seeing the entries of the matrix as a stream?

In order to formulate these questions we focus on the WWW application and its object-feature matrix known as the *document-term matrix* as our test case.

1.1.1 Formulation of the first question: Clustering problems

Latent semantic indexing (LSI) [DDL$^+$90] is the known model in information retrieval to formulate the first question. According to LSI, we think of k ideal documents $c_1, \cdots, c_i, \cdots, c_k$ each one describes one *topic* or *pattern*. A cluster C_i then would be all documents which are more *related* to the topic or better to say more *similar* to the pattern of the topic described by the ideal document c_i than the other topics. In this way documents in the same cluster are similar to each other with respect to a given similarity measure, whereas documents in different clusters are dissimilar.

Geometrically, each document including ideal ones is a vector in d-dimensional Euclidean space $\mathbb{R}^d$ where d is the number of terms or columns of the document-term matrix. We assume that the vectors corresponding to the ideal documents $c_1, \cdots, c_k$ are orthonormal basis of a k-dimensional linear subspace F, i.e., they are orthonormal and span F. This means that topics are independent of each other. Although this assumption is unrealistic, we make it to here to simplify the discussion. Note that the end point of the vector which corresponds to an ideal document c_i is usually called the *center* c_i.

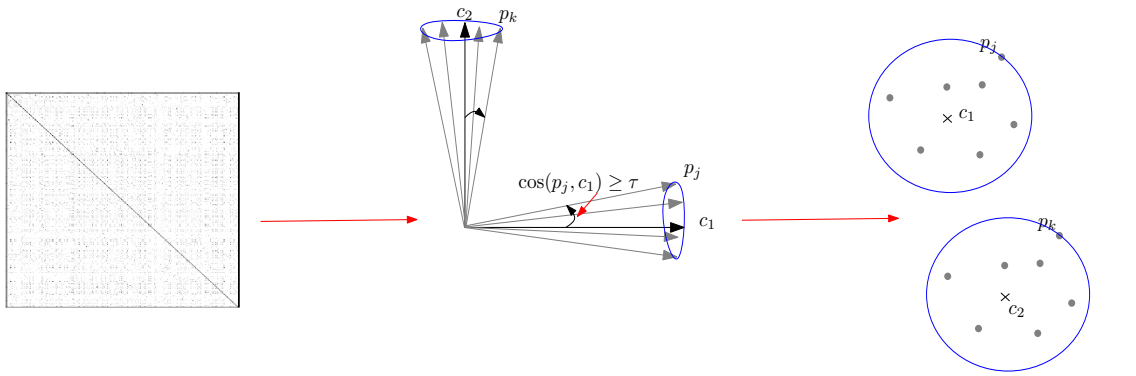
The similarity of two documents is measured using *cosine similarity measure* where two documents are similar if the cosine of the angle between their vectors is greater than some threshold τ. Using the cosine similarity measure, clustering web documents is defined as follows: all documents for which the cosine of their angles with c_i is greater than the cosine of their angles with $c_{j\neq i}$ are assigned to the cluster C_i. See the middle picture in Figure 1.1.

The cosine similarity measure is close to the Euclidean distance in $\mathbb{R}^d$. In fact, if we assume a vector x is normalized so that $x^T x = 1$ we get that $\|x - \text{proj}(x, c_i)\|_2^2 = 1 - \cos^2(\angle(x, c_i))$ where $\text{proj}(x, c_i)$ is the projection of the end point of the vector x onto c_i and $\|\cdot\|_2$ is the Euclidean distance. So we can reformulate the clustering web documents as assigning a vector to an ideal document c_i when $\|x - \text{proj}(x, c_i)\|_2^2$ is smaller than $\|x - \text{proj}(x, c_{j\neq i})\|_2^2$ for any j.

However, since we do not know the ideal documents c_i (which are vectors in $\mathbb{R}^d$) in advance, we define a global objective function to minimize the cost of clustering web documents where the minimization will be over all subsets of k vectors in $\mathbb{R}^d$ such that each two of these vectors are orthogonal.

Thus the first formulation of the first question would be: Given a set of vectors in $\mathbb{R}^d$ and k, find k orthogonal vectors $\{c_1, \cdots, c_k\} \subseteq \mathbb{R}^d$ subject to the constraint that the sum of squared distances of each vector to its nearest vector in $\{c_1, \cdots, c_k\}$ is minimum, where the distance of two vectors is the Euclidean distance between their end points, i.e., the length of the vector difference of the two vectors. We call this formulation k-*orthogonal topics*. Note that we put the restriction of finding k orthogonal vectors corresponding to the ideal documents $c_1, \cdots, c_k$ mainly because for the simplicity of exposition we assume that the topics are independent of each other.

A relaxation of this problem where the k vectors corresponding to the ideal documents do not need to be orthogonal is known as the k-*means* problem. In Chapter 5 we give an algorithm for the k-means problem such that in time $O(ndk + d \cdot (k/\epsilon)^{O(1)} + 2^{\tilde{O}(k/\epsilon)})$ returns a set $C \subseteq \mathbb{R}^d$ of k centers which is $(1+\epsilon)$-approximation to the optimal k-means cluster centers. The right picture in Figure 1.1 depicts clustering web documents according to the k-means clustering.



Raw document-term matrix
Topics (=patterns) are hidden

Topics are considered as an orthonormal basis of k-dimensional linear subspace

Topics are considered as centers in k-means

Fig. 1.1: Extracting well-structured patterns from a document-term matrix.

Although the k-means clustering is a good starting point to approximately extract patterns and cluster web documents, it can easily run into a problem known as the *curse of dimensionality*. In particular since the term-document matrix is a sparse matrix, i.e., not each document contains all the terms, we are expecting that all distances in full dimensional spaces become the same (i.e., tend to zero) and therefore clustering in full dimension would be meaningless.

Principal component analysis (PCA) [Pea01] is a basic technique in machine learning that addresses this problem. In PCA and in order to shave meaningless dimensions or terms we project orthogonally the rows of the term-document matrix onto a lower dimensional linear subspace of dimension k so-called *principal subspace* such that the variance of the projected vectors (i.e., rows) is maximized. In this way we are hoping that each cluster is totally separated from the rest of the clusters and can be extracted easily.

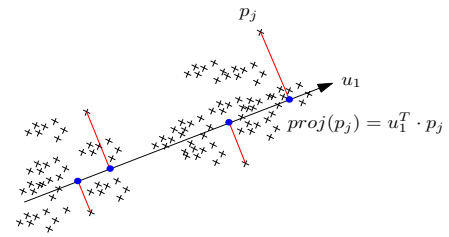
The PCA uses the basic *Singular Value Decomposition* (SVD) [GL96] to find the principal subspace. Let n, d denote the number of rows and columns of the document-term matrix respectively. The SVD in time $O(\min(nd^2, n^2d))$ and using $O(\log n)$ passes over the document-term matrix obtains an orthonormal basis of rank d for the document-term matrix in which the singular vectors of this basis are being sorted descending according to the singular values.

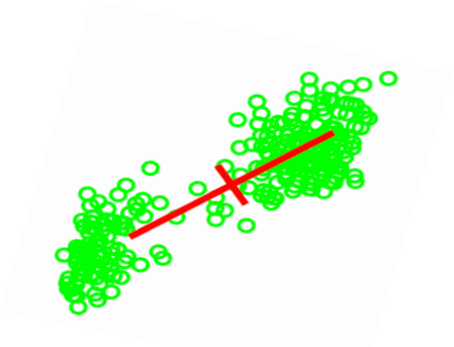
Recall that we assume that the vectors corresponding to the ideal documents $c_1, \cdots, c_k$ are orthonormal basis of a k-dimensional linear subspace. Using SVD (see for example, Page 561 of Chapter 12 of [Bis06]) we can show that the span of the first k singular vectors which correspond to the top k singular values contains the principal subspace and the best choice for $c_1, \cdots, c_k$ would be these first k singular vectors.

As an example we can consider the first, second and third pictures in Figure 1.2. The first picture shows projection onto a line that maximizes the variance of vectors (their end points are only shown). The second picture shows how PCA better clusters real data (Page 568

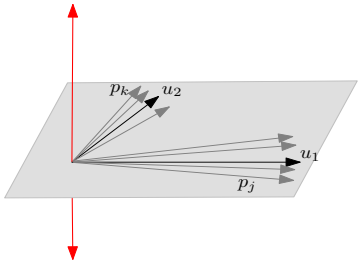
[Bis06]).

The third example shows projection onto a plane that maximizes the variance of vectors. The last picture of Figure 1.2 unravels topics and clusters found using PCA for the raw document-term matrix of Figure 1.1, see [CKVW06] for further discussion and examples.

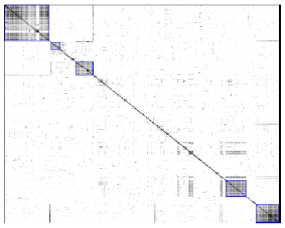




Maximizing the variance using a 1-dimensional subspace is the same as minimizing the sum-of-squares of the projection erros onto a 1-dimensional subspace



Maximizing the variance using a 2-dimensional subspace



Patterns found in document-term matrix

Fig. 1.2: Shaving the meaningless terms using the dimensionality reduction of principal component analysis. For further reading see Page 561 of Chapter 12 of [Bis06] and [CKVW06].

It is known in machine learning (see Chapter 12 of [Bis06]) that the problem of maximizing the variance of projected vectors is equivalent to the problem of minimizing the average projection cost defined as the sum of squared distances between the vectors and their projections onto the principal subspace. In fact, the latter problem can be seen as the problem of finding the $\mathrm{d} - \mathrm{k}$ singular vectors corresponding to the $\mathrm{d} - \mathrm{k}$ smallest singular values of the document-term matrix. The relation between these two problems is demonstrated via examples in the first three pictures of Figure 1.2.

Thus we end up with a new formulation of the first question which is defined as follows: Given a set of vectors in $\mathbb{R}^{\mathrm{d}}$ and k, find a k-dimensional linear subspace in $\mathbb{R}^{\mathrm{d}}$ such that the sum of squared distances of each vector to this subspace is minimum, where the distance of one vector to a subspace is the Euclidean distance between the end point of the vector and its projection on the subspace. We call this problem *subspace problem*.

Interestingly, as for the comparison between the k-means problem and the subspace problem, Drineas at al. [DFK$^+$04] show that the k-means problem is NP-hard even for $\mathrm{k} = 2$. On the other hand as we explained in the above the subspace problem can be solved exactly using SVD. It is simple to see that the cost of clustering according to k-means clustering is an upper bound to the cost of the subspace problem. On the other hand Drineas at al. in [DFK$^+$04] show

that the exact solution to subspace problem gives 2-approximation for the k-means problem.

So it seems SVD which solves the second formulation of the first question namely the subspace problem efficiently is a good option to use. Unfortunately this is not as easy as it seems. There are two issues with using SVD to solve the subspace problem:

- First, its running time which is $O(\min(nd^2, n^2d))$,
- Second, $O(\log n)$ passes that it needs to have over the whole document-term matrix.

We deal with the second issue in the next section.

For massive data sets we usually assume that n, d are very large numbers and k is constant; thus the quadratic dependency to n or d in the running time of SVD is one of the drawbacks of applying SVD to massive data set scenarios. In this thesis we mainly consider a variant of the subspace problem endowed with sum of distances not sum of squared distances as our measure. This is due to the fact that the sum of distances is more robust against outliers than sum of squared distances. For this problem in Chapter 6 we give an algorithm such that in time $O(nd \cdot \text{poly}(k/\epsilon) + (n + d) \cdot 2^{\text{poly}(k/\epsilon)})$ returns a k-dimensional linear subspace $C \subseteq \mathbb{R}^d$ which is $(1 + \epsilon)$-approximation to the optimal k-dimensional linear subspace. For constant k, this algorithm has linear dependency to n, d instead of quadratic dependency that we see in SVD.

Now we justify why we need non-uniform sampling instead of uniform sampling for this scenario. From now on we assume that each vector in $\mathbb{R}^d$ is represented with its end point. Let us assume we are given a point set $P = \{p_1, \cdots, p_n\} \subset \mathbb{R}^d$ and a center set $C \subseteq \mathbb{R}^d$ which can be k points in the k-means problem or a j-dimensional linear subspace represented as a span of k points in $\mathbb{R}^d$ in the subspace problem. We can see the distances of points to C as an n-dimensional vector a where the i-th coordinate a_i stores $\text{dist}^p(p_i, C)$ for $p \in [0, 2]$. Here $\text{dist}^p(p_i, C)$ is the p-th power of the distance of ith point $p_i \in P$ to the nearest center in C.

In order to approximate the p-th norm of a, i.e., $||a||_p^p = \sum_{i=1}^n |a|^p = \sum_{i=1}^n \text{dist}^p(p_i, C)$ within factor $(1 + \epsilon)$, we can use uniform sampling to sample s coordinates for some constant s and assign a weight of n/s to each sampled coordinate. However, this might be a problem when we have few coordinates of large absolute value, since uniform sampling does not hit them and therefore the estimator obtained using uniform random samples can be very far from the p-th norm of a. On the other hand if we do random sampling such that the probability of sampling the i-th coordinate depends on $|a_i|^p$, then we can prove that with enough probability we will have enough samples from coordinates of large absolute value that in turn helps us to get $(1 + \epsilon)$-estimator to $||a||_p^p$.

1.2 Formulation of the second question: Sampling in data streams

The second issue with SVD is exactly the second question that we posed in the beginning of this chapter. The SVD needs $O(\log n)$ passes over the document-term matrix to solve the subspace problem which may not always be possible. Also term-document matrices are in general huge and sparse and we cannot store them in the main memory.

Let us consider the following dynamic scenario. A search engine periodically runs its crawlers to download a subset of web pages. Building upon its previous crawls, the search engine has already constructed a document-term matrix, but since this matrix is sparse and huge, the search engine has stored for each row i of the matrix a compact representation containing some of the i's entries.

During the next run of crawlers, when a crawler is responsible to download a modified version of a document i, it reads the document from the beginning till the end and applies the updates (i.e., insertion or deletion of occurrences of terms) to the underlying row i of the matrix. However, since we do not store all entries of row i this will be a problem.

We model this scenario as a stream of updates of an d-dimensional vector a corresponding to the row i and we would like to devise a streaming algorithm which maintain entries conveying more information than the other entries in this vector. Here the claim is if we can maintain this type of entries we should be able to have a compact representation of a from which we can retrieve a approximately. However we do not know in advance which entries conveying more information. Therefore we consider this heuristic that those entries conveying more information are those that have greater absolute values than the others. We call them *heavy hitters* and our goal would be to maintain them as we see updates to the underlying vector a.

We can think of this problem as follows: Given an n-dimensional vector a as a stream of m updates to a for $m = n^{O(1)}$ we would like to approximate its p-th norm, i.e., $||a||_p^p = \sum_{i=1}^{n} |a_i|^p$ for $p \in [0, 2]$ within factor $(1 + \epsilon)$ using sample coordinates. Note that these sampled coordinates are mostly chosen from heavy hitters of a as otherwise we cannot approximate the p-th norm of a within factor $(1+\epsilon)$. So we came up with a formulation for the second problem which once again needs non-uniform sampling.

1.3 Approximating a sum without computing all the summands

In general both formulations that we discussed in the above are special cases of the basic problem *approximating a sum without computing all the summands* which is formally defined as follows:

Definition 1 (Approximating a sum without computing all the summands) *Let* $0 < \epsilon < 1$. *Let* $\delta > 0$. *Assume* Z *is the sum of* n *numbers,* $Z_1, \cdots, Z_n$ *i.e.,* $Z = Z_1 + \cdots + Z_n$. *Find an estimator* $X = X_1 + \cdots + X_s$ *for* $s << n$ *such that*

$$\mathbf{Pr}[(1 - \epsilon)Z \leq X \leq (1 + \epsilon)Z] \geq 1 - \delta,$$

under the restriction that each X_j *for* $1 \leq j \leq s$ *depends only on one of the summands of* Z*.*

In Chapter 3 we see that for $s \geq O(\epsilon^{-2}\log(1/\delta))$ we can get such an estimator X.

For the formulation of the first question, given a point set $P = \{p_1, \cdots, p_n\} \subset \mathbb{R}^d$ and a center set C which can be k points or a j-dimensional linear subspace in $\mathbb{R}^d$, we set $Z_i = \text{dist}^p(p_i, C)$, where $\text{dist}^p(p_i, C)$ is the p-th power of the distance of ith point $p_i \in P$ to the nearest center in C.

For the formulation of the second question, given an n-dimensional vector a, we set $Z_i = |a_i|^p$, $Z = \|a\|_p^p = \sum_{i=1}^n |a_i|^p$ for $p \in [0, 2]$.

1.4 Thesis Roadmap

Besides the introduction this thesis includes 7 more chapters:

- Chapter 2: Preliminaries, in which we give some known concentration bounds, the definitions of independent random variables and the k-means and the subspace clustering problems. We also introduce the merge and reduce method and the turnstile streaming model.
- Chapter 3: Overview, where we solve the problem of approximating a sum without computing all the summands and show that the k-means and the subspace clustering problems can be cast to this problem. We then give an overview of the way that we implement L_p-sampler using $O(\log n)$ passes and 1 pass.
- Chapter 4: Related Works, in which we survey the known results about random linear combination, random linear combination in data stream so called sketching, and coresets. We then explain the way that we can solve the k-means and subspace clustering problems using random linear combination. For the sketching section we compare known results with our results; but for the rest we only mention the known results and one can compare them by reading the overview chapter 3.
- Chapter 5:The k-Means Clustering, in which we give a full proof of the coreset construction for the k-means problem. We also obtain linear time $(1+\epsilon)$-approximation and 1-pass streaming algorithms for this problem.
- Chapter 6: The Subspace Clustering, in this chapter we extend the idea of non-uniform sampling from the k-means problem to the subspace clustering. We also give linear time $(1+\epsilon)$-approximation and 1-pass streaming algorithms for this problem.
- Chapter 7: $O(\log n)$-Pass L_p-Sampler, in this chapter we show an $O(\log n)$-pass implementation of L_p-sampler.
- Chapter 8: 1-Pass L_p-Sampler, in which we show how to reduce the number of passes from $O(\log n)$ to just 1 pass.

- Chapter 9: Applications of the L_p-Sampler, in this final chapter we give some applications of L_p-sampler including moment estimation, cascaded norms, heavy hitter and block heavy hitters.

Chapter 2

Preliminaries

We frequently use the following concentration bounds in the following chapters of this thesis. The proofs of these concentration bounds can be found in any textbook on the probabilistic method for example in [MR95] besides the original papers.

2.1 Concentration Bounds

Lemma 2 (Chebysev's Inequality) *[AS00] Let X denote a random variable with the expectation $\mathbf{E}[X]$ and the finite variance $\mathrm{Var}[X]$. For the real number $K > 0$ we get*

$$\mathbf{Pr}[|X - \mathbf{E}[X]| \geq K] \leq \frac{\mathrm{Var}[X]}{K^2}.$$

The proof can be found in Page 41 (Theorem 4.1.1) of [AS00].

Lemma 3 (Chernoff Bound) *[Che52] Let $Y_1, \cdots, Y_m$ denote m identically distributed and independent random variables such that $\mathbf{E}[Y_i] = p$ for $1 \leq i \leq n$ for a fixed $0 \leq p \leq 1$. Let $0 < t < 1, t \geq p$. For $Y = \sum_{i=1}^{m} Y_i$ it holds that*

$$\mathbf{Pr}[Y \geq t \cdot m] \leq \left[\left(\frac{p}{t}\right)^t \cdot \left(\frac{1-p}{1-t}\right)^{(1-t)} \right]^m.$$

Lemma 4 (Additive Hoeffding Inequality) *[Hoe63] Let $Y_1, \cdots, Y_m$ denote m independent random variables such that $1 \leq Y_i \leq M$ for $1 \leq i \leq n$. Then for $Y = \sum_{i=1}^{m} Y_i$ and $t > 0$ we get*

$$\mathbf{Pr}[|Y - \mathbf{E}[Y]| \geq t] \leq 2\exp\left(-\frac{mt^2}{3M^2}\right).$$

Lemma 5 (Multiplicative Hoeffding Inequality) *[Hoe63] Let $Z_1, \cdots, Z_m$ denote m independent random variables such that $0 \leq Z_i \leq 1$ for $1 \leq i \leq m$. Then for $Z = \sum_{i=1}^{m} Z_i$ and $0 < \epsilon < 1$ we get*

$$\mathbf{Pr}[|Z - \mathbf{E}[Z]| \geq \epsilon \mathbf{E}[Z]] \leq 2 \exp\left(-\frac{\mathbf{E}[Z] \cdot \min(\epsilon, \epsilon^2)}{3}\right).$$

2.2 Independent Random Variables

Consider a set of n random variables indexed by a set U, i.e., $\{Y_x : x \in U\}$ with $|U| = n$, that take on values from a set T, (i.e., $Y_x \in T$). Let $D : T^U \to [0, 1]$ denote their joint distribution function, i.e., $\mathbf{Pr}[Y_1 = \alpha_1 \wedge \cdots \wedge Y_n = \alpha_n] = D(\alpha_1, \cdots, \alpha_n)$ for $\alpha_i \in T$.

For finite $t = |T|$, a uniform distribution (i.e., $D(\alpha_1, \cdots, \alpha_n) = 1/t^n$) assigns $\mathbf{Pr}[Y_x = \alpha_x] = 1/t$, for all $x \in U$, $\alpha_x \in T$. If this distribution satisfies, for all $x \neq y \in U$ and for $\alpha, \beta \in T$,

$$\mathbf{Pr}[Y_x = \alpha \wedge Y_y = \beta] = \mathbf{Pr}[Y_x = \alpha] \cdot \mathbf{Pr}[Y_y = \beta] = 1/t^2,$$

then we refer to this distribution as *pairwise independent*.

In general, we say the random variables Y_x for $x \in U$ are d-*wise independent* if, for all d indexes $x_1, \cdots, x_d \in U$ and for all $\alpha_1, \cdots, \alpha_d \in T$, we have

$$\mathbf{Pr}[Y_{x_1} = \alpha_1 \wedge \cdots \wedge Y_{x_d} = \alpha_d] = \mathbf{Pr}[Y_{x_1} = \alpha_1] \cdots \mathbf{Pr}[Y_{x_d} = \alpha_d] = 1/t^d.$$

The following lemma shows that small sample spaces with arbitrary n d-wise independent random variables in them can be constructed using $O(\log n)$ bit space. This helps us to develop streaming algorithms for those problems which need d-wise independent random variables. To implement this construction we only need an irreducible polynomial of degree k, where 2^k is the smallest power of 2 greater than n. The construction can be from any linear error correcting code with appropriate parameters.

Lemma 6 *(Theorem 15.2.1 of [AS00]) Let $n = 2^k - 1, d = 2t + 1$. Assume that the field $F = GF(2^k)$ is represented as a k-dimensional vector over $GF(2)$. Then there exists a symmetric probability space Ω of size $2(n+1)^t$ and d-wise independent random variables $Y_1, \cdots, Y_n$ over Ω each of which takes the values 0 and 1 with probability $1/2$.*

Moreover, the space and the variables can be explicitly constructed.

In Theorem 70 of Chapter 8 we use the following concentration bound for d-wise independent random variables.

Theorem 7 *(Lemma 2.3 of [BR94]) Let $X_i \in [0, 1]$, $1 \leq i \leq n$, be d-wise independent for $d \geq 4$ an even integer, $X = \sum_{i=1}^{n} X_i$, and $A > 0$. Then $\mathbf{Pr}[|X - \mathbf{E}[X]| \geq A] \leq 8\left(\frac{d\mathbf{E}[X]+d^2}{A^2}\right)^{d/2}$.*

Definition 8 *A family of functions $\mathcal{H} \subseteq [N] \to [N]$ is called (ϵ, s)-min-wise independent if for any $X \subseteq [N]$ with $|X| \leq s$, and any $x \in [N] \setminus X$, we have*

$$\Pr_{h \in \mathcal{H}}[h(x) < \min h(X)] = (1 \pm \epsilon) \cdot \frac{1}{|X| + 1}.$$

Indyk in [Ind99] shows that any $(c' \log 1/\epsilon)$-wise independent family of functions is (ϵ, s)-min-wise independent. This allows us to use the construction of Lemma 6 to implement (ϵ, s)-min-wise independent family of functions in data stream as we do it in Algorithm Sample-Extraction and Lemma 80 In Section 8.1.

Theorem 9 *([Ind99]) There exist constants $c, c' > 1$ such that for any $\epsilon > 0$ and $s \leq \epsilon N/c$, any $(c' \log 1/\epsilon)$-wise independent family of functions is (ϵ, s)-min-wise independent.*

2.3 The k-Means Problem

A set of points P in $\mathbb{R}^d$ is weighted, if each point $p \in P$ is associated with a weight $w_p > 0$. We define $w(P) = \sum_{p \in P} w_p$ to be the total weight of P. We consider an (unweighted) set of points $P \subset \mathbb{R}^d$ as a weighted set with $w_p = 1$, for each $p \in P$.

For two points $p, q \in \mathbb{R}^d$ we use $\text{dist}(p, q) = \|p - q\|_2$ to denote the Euclidean distance between p and q. The $\text{dist}^2(p, q) = (\text{dist}(p, q))^2$ will denote the square of the Euclidean distance. We generalize these definitions to sets: Given a point $p \in \mathbb{R}^d$ and a set of points $Q \subseteq \mathbb{R}^d$ we define $\text{dist}(v, Q) = \min_{q \in Q} \text{dist}(v, q)$ and $\text{dist}^2(v, Q) = \min_{q \in Q} \text{dist}^2(v, q)$. Further, we define the distance between two sets $Q, R \subseteq \mathbb{R}^d$ as $\text{dist}(Q, R) = \min_{q \in Q} \text{dist}(q, R)$.

Given a point set Q and a center set $C = \{c_1, \cdots, c_k\}$ both in $\mathbb{R}^d$, we define

$$\text{cost}(Q, C) = \sum_{q \in Q} w_q \cdot \text{dist}^2(q, C).$$

We define the contribution of a point p to a center set C as $w_p \cdot \text{dist}^2(p, C)$.

The k-means problem is defined as follows:

Definition 10 (The k-means clustering) *Given a set $P = \{p_1, \cdots, p_n\}$ of points in the $\mathbb{R}^d$ the k-means problem is to find a set of k centers $C^* \subseteq \mathbb{R}^d$ such that $\text{cost}(P, C^*)$ is minimized.*

The point $\mu_P(P) = \frac{\sum_{p \in P} p}{|P|}$ is the *centroid* of P. For the 1-means problem the centroid is known to be the optimal cluster center, i.e. $\mathcal{OPT}(P, 1) = \sum_{p \in P} \text{dist}^2(p, \mu_P(P))$ where $\mathcal{OPT}(P, 1)$ is the optimal 1-mean cost of P. Inaba and et. al [IKI94] showed that if we draw a random sample U of size $2/\epsilon$ with constant probability the centroid of U is with constant probability a $(1 + \epsilon)$-approximation for the centroid of point set P, that is,

$$\text{cost}(P, \mu_P(U)) \leq (1 + \epsilon)\text{cost}(P, \mu_P(P)).$$

This implies that (see also [KSS04, KSS05])

Corollary 11 *Let P be a set of points in $\mathbb{R}^d$. Then there exists a set $U \subseteq P$ of size $2/\epsilon$, such that*

$$cost(P, \mu_P(U)) \leq (1+\epsilon)cost(P, \mu_P(P)) \ .$$

Given an integer $k \geq 1$, we denote by

$$\mathcal{OPT}(P,k) = \min_{|K|=k, K\subseteq\mathbb{R}^d} cost(P,K),$$

the optimal k-means cost of P. A set $C \subset \mathbb{R}^d$, $|C| = k$, is a β-approximation for an optimal k-means solution of P, if $cost(P,C) \leq \beta \cdot \mathcal{OPT}(P,k)$ for some constant β.

A set $\mathcal{T} \subseteq \mathbb{R}^d$ is a (k,ϵ)-*approximate centroid set* for P, if there exists a subset $C_\epsilon \subseteq \mathcal{T}$ of size k such that $cost(P,C_\epsilon) \leq (1+\epsilon)\cdot\mathcal{OPT}(P,k)$. By Corollary 11 the set of subsets of P each one of size $2/\epsilon$ points (with repetitions) is a (k,ϵ)-approximate centroid set for P. The size of this (k,ϵ)-approximate centroid set of P is $O(n^{2/\epsilon})$.

Definition 12 (Weak (k,ϵ)-coreset) *Let P be a (possibly) weighted set in $\mathbb{R}^d$. A pair $(S,\mathcal{T})$, $S \subseteq P$, is called a weak (k,ϵ)-coreset, if $\mathcal{T}$ is a (k,ϵ)-approximate centroid set for P and*

$$|cost(S,K) - cost(P,K)| \leq \epsilon \cdot cost(P,K),$$

for any set $K \subseteq \mathcal{T}$ with $|K| = k$.

Definition 13 (Strong (k,ϵ)-coreset) *Let P be a (possibly) weighted set in $\mathbb{R}^d$. A set S, $S \subseteq P$, is called a strong (k,ϵ)-coreset, if*

$$|cost(S,K) - cost(P,K)| \leq \epsilon \cdot cost(P,K),$$

for any set $K \subseteq \mathbb{R}^d$ with $|K| = k$.

Definition 14 (Linear time $(1+\epsilon)$-approximation algorithm for the k-means) *Given a set P of n points in the $\mathbb{R}^d$ and a parameter $\epsilon > 0$, an algorithm is called a linear time $(1+\epsilon)$-approximation algorithm for the k-means clustering problem if whenever we invoke it on P it returns a set $C_\epsilon \subset \mathbb{R}^d$ of k centers such that*

$$cost(P,C_\epsilon) \leq (1+\epsilon)\cdot\mathcal{OPT}(P,k).$$

Moreover, the running time of a linear time $(1+\epsilon)$-approximation algorithm is required to be linear in $n \cdot d$ for every fixed ϵ, k.

As an example assume we can compute a weak (k,ϵ)-coreset $(S,\mathcal{T})$ for the k-means problem in time $O(nd\cdot poly(|S|))$. A linear time $(1+\epsilon)$-approximation algorithm for the k-means would be to compute the cost of S to each subset $C \subseteq \mathcal{T}$ of size k and report the one with minimum cost. The running time of this linear time $(1+\epsilon)$-approximation algorithm will be $O(nd\cdot poly(|S|) + d\cdot|S|\cdot\mathcal{T})$ where d is because we are in $\mathbb{R}^d$.

2.4 The Subspace Problem

The Euclidean distance of a point $r \in \mathbb{R}^d$ to a j-subspace $C \subseteq \mathbb{R}^d$ is $\text{dist}(r, C) := \inf_{c \in C} \|r - c\|_2$. The C is called a *center*. We let $\text{cost}(P, C) = \sum_{r \in P} w(r) \cdot \text{dist}(r, C)$ to be the weighted sum of distances from the points of P to C where $w(r)$ is the weight of r in P. We allow weights to be negative in the subspace problem. Note that points with negative weights are also assigned to their closest (and not farthest) point $r \in C$. For a j-dimensional linear subspace or j-*subspace* for short, C, we define $\text{proj}(r, C)$ to be the closest point to r in C. We further define the weight of $\text{proj}(r, C)$ as $w(\text{proj}(r, C)) = w(r)$. Similarly, $\text{proj}(P, C) = \{\text{proj}(r, C) \mid r \in P\}$.

We should mention that in comparison to the k-means problem where we consider the sum of squared distances as our measure, for the j-subspace problem we consider the sum of distances.

Definition 15 (The j-subspace problem) *Given a set $P = \{p_1, \cdots, p_n\} \subset \mathbb{R}^d$ of n points, find a j-subspace $C^* \subseteq \mathbb{R}^d$ such that $\text{cost}(P, C^*)$ is minimized.*

We define approximate centroid set and linear time $(1+\epsilon)$-approximation algorithm for the j-subspace problem in the same line of the k-means problem.

Definition 16 ((j, ϵ)-*approximate centroid set*) *A set $\mathcal{T}$ of j-subspaces in $\mathbb{R}^d$ is called a* (j, ϵ)-approximate centroid set *for P, if there exists a j-subspace $C_\epsilon \in \mathcal{T}$ such that*

$$\text{cost}(P, C_\epsilon) \leq (1+\epsilon) \cdot \min_{j\text{-subspace } C \subset \mathbb{R}^d} \text{cost}(P, C).$$

Definition 17 (Linear time $(1+\epsilon)$-approximation algorithm for the j-subspace problem) *Given a set P of n weighted points in $\mathbb{R}^d$, and $j \geq 0, 1 > \epsilon > 0$, an algorithm is called a linear time $(1+\epsilon)$-approximation algorithm for the j-subspace problem if whenever we invoke it on P it returns a set $C_\epsilon \in \mathbb{R}^d$ of k centers such that*

$$\text{cost}(P, C_\epsilon) \leq (1+\epsilon) \cdot \min_{j\text{-subspace } C \subset \mathbb{R}^d} \text{cost}(P, C).$$

Moreover, the running time of a linear time $(1+\epsilon)$-approximation algorithm is required to be linear in $n \cdot d$ for every fixed ϵ, j.

A strong coreset for the j-subspace problem is defined as follows:

Definition 18 (Strong coreset [AHPV04, HPM04]) *Let P be a weighted point set in $\mathbb{R}^d$, and $j \geq 0, 1 > \epsilon > 0$. A weighted set of points Q is called a* strong ϵ-coreset *for the j-subspace problem, if for every j-subspace C of $\mathbb{R}^d$, we have*

$$(1-\epsilon) \cdot \text{cost}(P, C) \leq \text{cost}(Q, C) \leq (1+\epsilon) \cdot \text{cost}(P, C) \ .$$

2.5 Streaming Models

In this thesis we mostly consider the *insertion-only* and the *turnstile* streaming models. We define both models for an underlying n-dimensional vector a. The insertion-only model is defined as follows:

2.5.1 Insertion-only Model

In the insertion-only streaming model we are given a stream $\mathcal{S} = (b_1, \cdots, b_k, \cdots, b_m)$ of indexes in $[n]$ where $b_k \in [n]$, indicating that the b_k-th coordinate a_{b_k} of a n-dimensional vector a should be incremented by 1.

Merge and Reduce Method

Merge and reduce method inspired by a complete binary tree is a generic method in computational geometry to implement non-streaming algorithms in the insertion-only streaming model. Note that for geometric problems we usually consider the stream as a permutation of a point set $P \subset \mathbb{R}^d$ of n points. We can think of a point $p_i \in P$ as the i-th coordinate of an n-dimensional vector a.

For the k-means and the subspace problems, we use the merge and reduce method to maintain a coreset of the point set P in the insertion-only streaming model. Here we assume that we have a coreset procedure that given a subset $A \subseteq P$ of points generates a coreset for A. We call this coreset procedure a *merge operator*. Let b, $m_0, m_1, \cdots, m_{\log_b n}$ denote parameters which will be determined later for each problem.

First we explain the offline variant of the merge and reduce method. Assume we are given a point set $P \subset \mathbb{R}^d$ of size n, where n is known in advance. At the end of this section we deal with the case when n is not known in advance.We construct a b-ary tree T of depth $dep = \log_b n$. Let us assume level zero is the leaf level and level dep is the root of T. Starting from level zero we have leaves $c_1^0, \cdots, c_{n/m_0}^0$ each one has a bucket that can store points. We put the first m_0 points in c_1^0, the second m_0 points in c_2^0 and so on up to the n/m_0-th m_0 points that we store in c_{n/m_0}^0. We then apply the merge operator on buckets (or leaves) $c_{(r-1)b+1}^0$ up to c_{rb}^0 for $1 \leq r \leq n/(m_0 b)$ and put new coreset points in c_r^1 of size m_1 at level 1. Recursively and at level i, for $1 \leq i \leq \log_b n$ we merge buckets (or children) $c_{(r-1)b+1}^{i-1}$ up to c_{rb}^{i-1} for $1 \leq r \leq n/(m_0 b^{i-1})$ (of node c_r^i) and put new coreset points in c_r^i of size m_i at level i.

Now the streaming variant of this algorithm is as follows. Recall that since we are in the insertion-only model, the insertion of points is allowed but the deletion is not allowed. Assume the points are coming one by one and the result of the streaming algorithm should be correct with probability $\geq 1 - \delta$ for $0 < \delta < 1$.

We put the first m_0 points in c_1^0, the second m_0 points in c_2^0 and so on up to the b-th chunk of m_0 points which we store in c_b^0. At this time and in level zero we have $m_0 \times b$ points in b full leaves

and so we merge c_1^0 up to c_b^0 and put new points in c_1^1 of size m_1 at level 1 and deallocate the space assigned to $c_1^0, \cdots, c_b^0$ for new points. If we repeat this process for b times then we will have b full buckets at level 1 and so we must merge c_1^1 up to c_b^1 and put new points in c_1^2 of size m_2 at level 2 and deallocate the space assigned to $c_1^1, \cdots, c_b^1$ for new points.

As long as we see new points we do the same iteration, store them into b buckets at level 0 and then merge them into one of the b buckets at level 1, and deallocate the space assigned to the b buckets and so on. Therefore in each level i for $1 \leq i \leq \log_b n$ we have at most b buckets each one of size m_i and at each instance of time a union of all these buckets in all levels will be a compact representation of the stream up to that time. The merge step in level i w.l.o.g. at c_1^i is dependent to the problem that we consider and will be explained when we give the streaming algorithm for each problem.

If n is not known in advance. We start with some constant guess for n. If there are more points in the stream, compute a new value for n' such that the coreset size doubles. Use the streaming algorithm for the next n' points and keep the coreset for the first n points. Continue until no more points are coming. Since the coreset size doubles in each step, the space for the smaller coresets is at most the space for the largest one (by geometric progression). In order to make sure that everything works with good probability, one chooses confidence probability of δ/i^2 for the i-th step. Since sum $1/i^2$ is constant, we get an overall error of $O(\delta)$.

2.5.2 Turnstile Model

In the turnstile model a stream $\mathcal{S}$ is given as $m = \text{poly}(n, M)$ updates of the form (i, x), where $i \in [n]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$, indicating that the i-th coordinate a_i of a should be incremented by x.

2.5.3 Basic Problems in the Turnstile Model

Definition 19 (Approximating p-th Frequency Moment of a Stream) *Let $0 < \epsilon, \delta \leq 1$. Given a stream $\mathcal{S}$ as $m = \text{poly}(n, M)$ updates of the form (i, x), where $i \in [n]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$, find an $(1 \pm \epsilon)$-estimator $F'_p(a)$ for $F_p(a) = F_p(S) = \sum_{i=1}^n |a_i|^p$ such that*

$$\mathbf{Pr}[|F_p(a) - F'_p(a)| \leq \epsilon \cdot F_p(a)] \geq 1 - \delta.$$

The space of every streaming algorithm which returns a $(1 \pm \epsilon)$-estimator $F'_p(a)$ should be sublinear in n.

Definition 20 (Heavy Hitter) *Let $p \in [0, 2]$. The classical heavy hitters problem is to report all coordinates i for which $|a_i| \geq \phi \|a\|_p$, where ϕ is an input parameter.*

Definition 21 (Block Heavy Hitters) *The block heavy hitters problem is to report all rows a^i of an $n \times d$ matrix A for which $\|a^i\|_1$ is at least a ϕ-fraction of the L_1-norm $\|A\|_1$ of A (i.e., $\|A\|_1 = \sum_{j=1}^n \|a^j\|_1$). These rows are called the block heavy hitters.*

The block heavy hitters are a crucial building block in the streaming algorithm of Andoni, Indyk, and Kraughtgamer [AIK09] that constructs a small-size sketch for the Ulam metric under the edit distance.

Definition 22 (Estimating Cascaded Norms) *Let* $0 < \epsilon, \delta \leq 1$. *Let* $\mathcal{S}$ *be a stream of* $m = \text{poly}(n, M)$ *updates of the form* (i, j, x) *to items in a* $[n] \times [d]$ *matrix* a, *where* $i \in [n], j \in [d]$ *and* $x \in \{-M, -M+1, \ldots, M-1, M\}$. *The problem of estimating cascaded norms is to find a* $(1 \pm \epsilon)$*-approximation to* $F_k(F_p)(a)$, *where* $F_k(F_p)(a)$ *means we first applying* F_p *to each row of* a, *and then apply* F_k *to the resulting vector of values, i.e,*

$$F_k(F_p)(a) = \sum_{i \in [n]} (\sum_{j \in [d]} |a[i, j]|^p)^k.$$

Finally we give the definition of L_p-sampler problem that we introduces in this thesis as a basic primitive that gives a generic framework to solve all the aforementioned problems in the the turnstile model.

Definition 23 (L_p-sampler) *Given a stream* S *of* $m = \text{poly}(n, M)$ *updates of the form* (i, x), *where* $i \in [n]$ *and* $x \in \{-M, -M+1, \ldots, M-1, M\}$, *indicating that the* i*-th coordinate* a_i *of* a *should be incremented by* x, *an* L_p*-sampler for* $p \in [0, 2]$ *outputs the* i*-th coordinate with probability in the interval*

$$\left[(1-\epsilon)\frac{|a_i|^p}{F_p(a)} - n^{-C}, (1+\epsilon)\frac{|a_i|^p}{F_p(a)} + n^{-C}\right],$$

where $F_p(a) = \sum_{i=1}^{n} |a_i|^p$ *and for an arbitrarily large constant* C.

Chapter 3

Overview

$\mathcal{A}$t the end of Chapter 1 we motivated that both problems of clustering and sampling in data streams can be seen as an n-dimensional vector a, where we need to find an $(1 \pm \epsilon)$-estimator for its p-th norm $||a||_p^p = \sum_{i=1}^{n} |a_i|^p$. We saw when we let $Z = ||a||_p^p$ and $Z_i = |a_i|^p$, this problem will be the same as the problem of approximating a sum without computing all the summands which we rephrase it here once again.

Definition 1: (Approximating a sum without computing all the summands) Let $0 < \epsilon < 1$. Let $\delta > 0$. Assume Z is the sum of n numbers, $Z_1, \cdots, Z_n$ i.e., $Z = Z_1 + \cdots + Z_n$. Find an estimator $X = X_1 + \cdots + X_s$ for $s << n$ such that

$$\mathbf{Pr}[(1-\epsilon)Z \leq X \leq (1+\epsilon)Z] \geq 1 - \delta,$$

under the restriction that each X_j for $1 \leq j \leq s$ depends only on one of the summands of Z.

In this chapter we see that for $s \geq O(\epsilon^{-2}\log(1/\delta))$ we can get such an estimator X.

First we justify mathematically why uniform sampling is not a good candidate to solve this problem. According to the uniform sampling we choose a number Z_i with probability $\frac{1}{n}$ and assign the weight n to Z_i. The number nZ_i will be our estimation for Z. The expectation of this random variable is

$$\mathbf{E}[nZ_i] = \sum_{i=1}^{n} n \cdot Z_i \times \frac{1}{n} = Z.$$

However, the variance of nZ_i can be large.

$$Var(nZ_i) = \sum_{i=1}^{n} (n \cdot Z_i)^2 \times \frac{1}{n} = n \sum_{i=1}^{n} Z_i^2 \leq 2n^2 \cdot Z^2.$$

Therefore we are not able to apply the concentration bounds of Section 2.1 to sample few of the summands and approximate Z. Intuitively, the reason is if the number of large numbers is few, then the probability that the random sample does not take one of them will be high and if this happens the variance of nZ_i will be large.

Using non-uniform sampling we can bound the variance in terms of the expectation and therefore estimate Z within factor $(1 \pm \epsilon)$ as follows: We assign n probabilities $r_1, \cdots, r_n$ to the numbers $Z_1, \cdots, Z_n$ respectively such that $r_i \geq \frac{q_i}{\gamma}$ for $q_i = \frac{Z_i}{Z}$, $1 \leq i \leq n$, and $\sum_{i=1}^{n} r_i = 1$. We take a sample set $A = \{a_1, \cdots, a_j, \cdots, a_s\} \subseteq [n]$ of indexes according to the probabilities r_i and assign a weight of $w(Z_{a_j}) = \frac{1}{s \cdot r_{a_j}}$ to a sampled number Z_{a_j} for $1 \leq j \leq s$. Corresponding to a sampled number Z_{a_j} we define a random variable $X_j = w(Z_{a_j}) \cdot Z_{a_j}$. Let $X = \sum_{j=1}^{s} X_j$.

The following lemma shows that for $s \geq O(\frac{\gamma^2}{\epsilon^2} \log(2/\delta))$ the estimator X approximates Z within $(1+\epsilon)$-factor with probability $\geq 1 - \delta$.

Lemma 24 *Let $0 < \epsilon < 1$. Let $\delta > 0$. Assume Z is the sum of n numbers, $Z_1, \cdots, Z_n$ i.e., $Z = Z_1 + \cdots + Z_n$. Let $X = X_1 + \cdots + X_s$ denote the estimator in which each random variable X_j drawn according to the distribution given in the above. For $s \geq O(\frac{\gamma^2}{\epsilon^2} \log(2/\delta))$ we have*

$$\mathbf{Pr}[(1-\epsilon)Z \leq X \leq (1+\epsilon)Z] \geq 1 - \delta.$$

We give two proofs for this lemma. The first proof uses additive Hoeffding inequality (Lemma 4) while the second proof uses Chebysev's inequality (Lemma 2). We conduct a similar proof to the first one to get estimators for the k-means problem in Chapter 5 and for the subspace problem in Chapter 6. Section 9.1 shows an application of the second proof to estimate the p-th norm $\|a\|_p^p$ of an n-dimensional vector a for $p > 2$.

Proof. [First proof using additive Hoeffding inequality (Lemma 4).] The expectation of the random variable X_j is

$$\mathbf{E}[X_j] = \sum_{i=1}^{n} \frac{Z_i}{s r_i} r_i = \frac{Z}{s}.$$

So the expectation of the random variable X will be $\mathbf{E}[X] = s\mathbf{E}[X_j] = Z$. Now we compute an upper bound for the random variable X_j as follows:

$$X_j \leq \max_j \frac{Z_{a_j}}{s r_{a_j}} \leq \frac{q_{a_j} Z}{s r_{a_j}} \leq \frac{\gamma Z}{s}.$$

We use the Hoeffding inequality to prove that for the claimed sample size, the random variable X concentrates around its expectation which is Z. In order to apply additive Hoeffding inequality, we set $M = \gamma Z$, $t = \epsilon Z$, and $m = s$ to get for $s \geq \frac{3\gamma^2}{\epsilon^2} \log(2/\delta)$

$$\begin{aligned}
\mathbf{Pr}[|X - Z| \geq \epsilon \cdot Z] = \mathbf{Pr}[|X - \mathbf{E}[X]| \geq \epsilon \cdot \mathbf{E}[X]] &\leq 2\exp\left(-\frac{s\epsilon^2 Z^2}{3\gamma^2 Z^2}\right) \\
&\leq 2\exp\left(-\frac{s\epsilon^2}{3\gamma^2}\right) \\
&\leq \delta.
\end{aligned}$$

□

Proof. **[Second proof using Chebysev's inequality (Lemma 2).]** In order to use the concentration bound of the Chebysev's inequality we need to find the variance of the random variable X_j

$$\begin{aligned}
Var[X_j] = \mathbf{E}[X_j^2] - (\mathbf{E}[X_j])^2 \leq \mathbf{E}[X_j^2] &= \sum_{i=1}^{n} (\frac{Z_i}{sr_i})^2 r_i = \sum_{i=1}^{n} \frac{Z_i^2}{s^2 r_i} \\
&= \sum_{i=1}^{n} \frac{q_i^2 Z^2}{s^2 r_i} \leq \frac{\gamma Z^2}{s^2} \sum_{i=1}^{n} q_i \\
&= \frac{\gamma Z^2}{s^2}.
\end{aligned}$$

Since for independent random variables, the variance of their sum is the sum of their variances we get $Var[X] = \sum_{j=1}^{s} Var[X_j] \leq \frac{\gamma Z^2}{s} = \frac{\gamma}{s}(\mathbf{E}[X])^2$. Now we can apply the Chebysev's inequality to get that for $s \geq \frac{3\gamma}{\epsilon^2}$

$$\begin{aligned}
\mathbf{Pr}[|X - Z| \geq \epsilon \cdot Z] = \mathbf{Pr}[|X - \mathbf{E}[X]| \geq \epsilon \cdot \mathbf{E}[X]] &\leq \frac{Var[X]}{(\epsilon \cdot \mathbf{E}[X])^2} \\
&= \frac{\frac{\gamma}{s}(\mathbf{E}[X])^2}{\epsilon^2 \cdot (\mathbf{E}[X])^2} \\
&= \frac{\gamma}{s\epsilon^2} \\
&= 1/3.
\end{aligned}$$

Thus with a constant probability X is a $(1+\epsilon)$-approximation of Z. In order to increase the probability of correctness to $1-\delta$, we repeat the whole random experiment of X for $m \geq 20\log(1/\delta)$ times and take the median of them as an estimation of Z. The Chernoff bound (Lemma 3) shows that the median of these experiments is a $(1+\epsilon)$-approximation of Z with probability at least $1-\delta$.

Let Y_i corresponds to the i-th experiment of X such that $Y_i = 1$ if for the i-th experiment $|X - Z| > \epsilon \cdot Z$ which happens with probability at most $1/3$; otherwise $Y_i = 0$. Note that $\mathbf{Pr}[Y_i = 1] = p \leq 1/3$ and $\mathbf{Pr}[Y_i = 0] = 1 - p \geq 2/3$. Therefore $\mathbf{E}[Y_i] = p \leq 1/3$. Let $t = 1/2$ and $Y = \sum_{i=1}^{m} Y_i$.

$$\begin{aligned}
\mathbf{Pr}[Y \geq m/2] &\leq \left[\left(\frac{p}{t}\right)^t \cdot \left(\frac{1-p}{1-t}\right)^{(1-t)} \right]^m \\
&\leq \left[\sqrt{2/3} \cdot \sqrt{\frac{2/3}{1/2}} \right]^m \\
&\leq \left[\sqrt{8/9} \right]^m \\
&\leq \delta,
\end{aligned}$$

for $m \geq \log_{\sqrt{9/8}}(1/\delta)$. By the relation between the logarithms we get $m \geq 20\log(1/\delta)$. □

In this thesis, we study the applications of approximating a sum without computing all the summands problem, i.e., Definition 1, in high dimensional geometric clustering and streaming algorithms.

In the following sections, we give an overall view of the problems that we study in this thesis and the way that we apply non-uniform sampling to solve these problems.

3.1 The k-Means Clustering

In Chapter 5 we study the k-means clustering problem in high dimensional Euclidean space $\mathbb{R}^d$. In this setting the time and the space complexity of algorithms are not allowed to be exponentially dependent on the dimension of the space. We show that using non-uniform sampling we can find a weak coreset for this problem. We then use the weak coreset to find linear time $(1+\epsilon)$-approximation algorithm and insertion-only streaming algorithms for the k-means problem.

Let C^* denote the optimal cluster centers for the k-means problem. As a first step to get a weak coreset, we would like to approximate $\text{cost}(P, C^*)$ within factor $(1+\epsilon)$ using a random sample set $S \subseteq P$ of small size. The first try would be to take S u.a.r and assign a weight of $|P|/n$ to each sample point. Then use the estimator $\text{cost}(S, C^*) = \sum_{s\in S} w_s \cdot \text{dist}^2(s, C^*)$ as an approximation to $\text{cost}(P, C^*)$.

Unfortunately, the variance of the random variable $\text{cost}(S, C^*)$ can be arbitrarily large. As an example, imagine that a point set P lies on a line and $k = 1$. Assume $n-2$ points are very near to each other and two points say p_1, p_2 are located on the left and the right side of this cloud at distance x. According to the paragraph above Corollary 11, the 1-mean center c of P must be somewhere near to the 1-mean center of the cloud, therefore the sum of squared distances of the cloud points to c is a small number say y. However we can choose $x >> \sqrt{y/2}$ to be an arbitrarily large number.

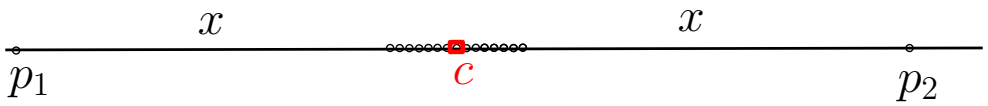


Fig. 3.1: Uniform sampling cannot hit a small set of points which are far from the center.

Taking a sample set u.a.r. does not hit p_1, p_2 with high probability so the contribution of the sample points to c will be very small in comparison to the contribution of p_1, p_2 to c which is arbitrarily large. We use the non-uniform sampling framework that we developed in the beginning of this chapter to bound the variance and therefore to find an $(1\pm\epsilon)$-estimator.

Put $Z = \text{cost}(P, C^*)$, $Z_i = \text{dist}^2(p_i, C^*)$, $q_i = \frac{Z_i}{Z}$, and $r_i = \frac{\text{dist}^2(p_i, C^*)}{\text{cost}(P, C^*)}$ for $1 \le i \le n$ so $\gamma = 1$. We take a sample set $A = \{a_1, \cdots, a_j, \cdots, a_s\} \subseteq [n]$ of indexes according to the probabilities r_i and assign a weight of $w(p_{a_j}) = \frac{1}{s\cdot r_{a_j}}$ to a sampled point p_{a_j} for $1 \le j \le s$. Corresponding to a sampled point p_{a_j} we define a random variable $X_j = w(p_{a_j}) Z_{a_j}$. Let $X = \sum_{j=1}^{s} X_j$.

Lemma 24 shows that for the sample set $S = \{p_{a_j} | a_j \in A\}$ of size $O(\frac{1}{\epsilon^2}\log(2/\delta))$ we get that

$$\mathbf{Pr}[|\text{cost}(P, C^*) - \text{cost}(S, C^*)| \leq \epsilon \cdot \text{cost}(P, C^*)] \geq 1 - \delta.$$

Now we want to approximate an arbitrary set $K \subseteq \mathcal{T}$ of k centers. Let C_β denote a β-approximation cluster centers for the k-means problem. Fix a subset $K \subseteq \mathcal{T}$ of size k. Once again we use the non-uniform sampling framework that we developed in the beginning of this chapter.

Put $Z = \text{cost}(P, K)$, $Z_i = \text{dist}^2(p_i, K)$, $q_i = \frac{Z_i}{Z}$, and $r_i = \frac{\text{dist}^2(p_i, C_\beta)}{\text{cost}(P, C_\beta)}$ for $1 \leq i \leq n$. We take a sample set $A = \{a_1, \cdots, a_j, \cdots, a_s\} \subseteq [n]$ of indexes according to the probabilities r_i and assign a weight of $w(p_{a_j}) = \frac{1}{s \cdot r_{a_j}}$ to a sampled point p_{a_j} for $1 \leq j \leq s$. Corresponding to a sampled point p_{a_j} we define a random variable $X_j = w(p_{a_j}) Z_{a_j}$. Let $X = \sum_{j=1}^{s} X_j$. Note that here we do the random sampling according to the contributions of points to C_β rather than C^*.

The explanation of the the above random process is as follows: we choose the points which are further from the centers C_β with higher probabilities than the points which are near to the centers C_β but they get lower weight than the near points. However there might be a problem with this random process. Although the near points are chosen with lower probabilities, they get high weights therefore if the contribution of very near points to C_β is high (i.e., we have many of them), one of them is chosen with constant probability and in this way this sample point gets a very high weight. In order to fix this problem we partition the point set P to points which are very near to the centers C_β say P_{near} and the rest $P \backslash P_{near}$. We then do the non-uniform sampling according to the probabilities r_i for $P \backslash P_{near}$ and uniform sampling for P_{near}.

Let S denote the union of samples from P_{near} and $P \backslash P_{near}$. In Chapter 5 we show that for an arbitrary set $K \subseteq \mathcal{T}$ of k centers we get

$$\mathbf{Pr}[|\text{cost}(P, K) - \text{cost}(S, K)| \leq \epsilon \cdot \text{cost}(P, K)] \geq 1 - \delta.$$

This means that random sampling according to C_β gives an *unbiased estimator* for a fixed $K \subseteq \mathcal{T}$ of size k. Using a union bound we can get a weak (k, ϵ)-coreset.

Having a weak (k, ϵ)-coreset $(\mathcal{S}, \mathcal{T})$, we can find a linear time $(1 + \epsilon)$-approximation algorithm and an insertion-only streaming algorithm for the k-means problem. In fact getting a linear time $(1 + \epsilon)$-approximation algorithm is fairly simple. We compute $\text{cost}(\mathcal{S}, K)$ for any set $K \subseteq \mathcal{T}$ with $|K| = k$ and report the center set with the minimum cost. Since every center set in $\mathcal{T}$ is $(1 + \epsilon)$-approximated we can get an approximation to the best center set in $\mathcal{T}$ which in turn gives an approximation to C^*.

For the streaming algorithm, we maintain a weak (k, ϵ)-coreset $(S, \mathcal{T})$ for the point set P seen as a stream using the merge and reduce method and at the end of the stream we extract the center set in $\mathcal{T}$ with minimum cost as we explained for the linear time $(1 + \epsilon)$-approximation algorithm in the previous paragraph.

3.2 The Subspace Clustering

In Chapter 6 we study the j-*subspace* problem which is the generalization of the k-means clustering problem. Ideally we would like to adapt the framework that we developed in the previous section using non-uniform sampling for the j-subspace problem as well. However the fact that a center is now a j-subspace not a point makes the problem harder. In order to explain the hardness of applying non-uniform sampling for the j-subspace problem we focus on 1-mean and 1-subspace problems in the 2-dimensional Euclidean plane. Let c denote the 1-mean center of a point set P. Let L denote the optimal 1-subspace of P. What simplifies applying non-uniform sampling for the 1-mean problem is that all the points that are at distance x of c have the same effect on it. On the other hand, points which have the same distance from the center line L may have different influences on L.

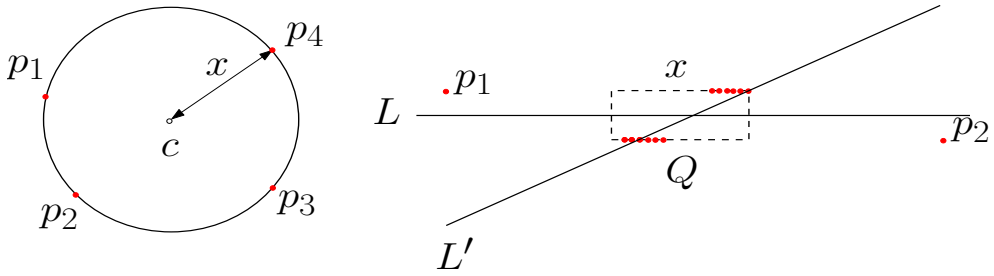


Fig. 3.2: Non-uniform sampling cannot preserve the direction of Line L.

Let us consider the example on the right side of Figure 3.2. In this example, all points in the set Q locating on the top and bottom sides of the dashed rectangle and the points p_1, p_2 have the same distance to L. Let $P = Q \cup p_1 \cup p_2$. Assume the maximum distance between two points on the top side and the maximum distance between two points on the bottom side of the dashed rectangle is small and p_1, p_2 are very far from Q. In this case, non-uniform sampling according to the distance of points to L choose all its samples points from Q and the points p_1, p_2 are not chosen. Now if we run an 1-subspace algorithm on the sample set S, it gives a solution like L' which is good for the point set Q but far from p_1, p_2. It is easy to adjust the distances so that $\text{cost}(P, L') >> \text{cost}(P, L)$.

Intuitively we need to preserve the direction of Line L and in parallel take sampling according to the distance of points to L. In order to implement this intuition, for every point $p_i \in P$ we project p_i on L and on the position of the projection we put two points, one $\text{proj}(p_i, L)$ having weight $+1$ and one $\text{proj}(p_i^-, L)$ having weight -1, where p_i^- is the point p_i with negative weight -1. See Figure 3.3

Let us assume that Line L is a constant factor approximation line i.e., $\text{cost}(P, L) \leq \beta \cdot \mathcal{OPT}$ for some constant β and L'' is an arbitrary line in $\mathbb{R}^d$ where $\mathcal{OPT}$ is the optimal cost of P with respect to the 1-subspace problem. For L'' the above geometric intuition corresponds to the following elementary formula

$$\begin{aligned} \text{dist}(p_i, L'') &= \text{dist}(p_i, L'') + \text{dist}(\text{proj}(p_i, L), L'') - \text{dist}(\text{proj}(p_i^-, L), L'') \\ &= \text{dist}(p_i, L'') - \text{dist}(\text{proj}(p_i^-, L), L'') + \text{dist}(\text{proj}(p_i, L), L''). \end{aligned}$$

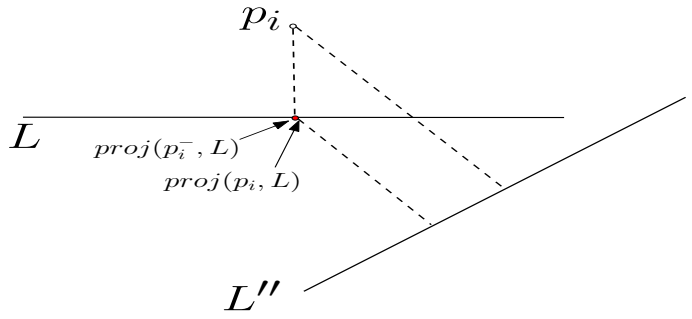


Fig. 3.3: Preserving the direction of L and in parallel taking sampling according to the distance of points to L.

We consider two points p_i and $\mathrm{proj}(p_i^-, L)$ as a pair and apply once again the non-uniform sampling framework that we developed in the beginning of this chapter on the pairs $(p_i, \mathrm{proj}(p_i^-, L))$ for $p_i \in P$ as follows.

Put $Z = \sum_{p_i \in P} \big(\mathrm{dist}(p_i, L'') - \mathrm{dist}(\mathrm{proj}(p_i^-, L), L'')\big)$, $Z_i = \mathrm{dist}(p_i, L'') - \mathrm{dist}(\mathrm{proj}(p_i^-, L), L'')$, $q_i = \frac{Z_i}{Z}$, and $r_i = \frac{\mathrm{dist}(p_i, L)}{s \cdot \mathrm{cost}(P,L)}$ for $1 \le i \le n$. We take a sample set $A = \{a_1, \cdots, a_\ell, \cdots, a_s\} \subseteq [n]$ of indexes according to the probabilities r_is and assign a weight of $w(a_\ell) = \frac{1}{s \cdot r_{a_\ell}}$ to a sampled pair $(p_{a_\ell}, \mathrm{proj}(p_{a_\ell}^-, L))$ for $1 \le \ell \le s$. Note that $\mathrm{proj}(p_{a_\ell}^-, L)$ gets the same weight as p_{a_ℓ} but with the negative sign. Corresponding to a sampled pair $(p_{a_\ell}, \mathrm{proj}(p_{a_\ell}^-, L))$ we define a random variable

$$X_\ell = w(a_\ell) Z_{a_\ell} = w(a_\ell) \left(\mathrm{dist}(p_{a_\ell}, L'') - \mathrm{dist}(\mathrm{proj}(p_{a_\ell}^-, L), L'')\right).$$

Let $X = \sum_{\ell=1}^{s} X_\ell$. Note that $\mathbf{E}[X_\ell] = \frac{Z}{s}$ and $\mathbf{E}[X] = Z$. By the triangle inequality we get

$$|\mathrm{dist}(p_{a_\ell}, L'') - \mathrm{dist}(\mathrm{proj}(p_{a_\ell}^-, L), L'')| \le \mathrm{dist}(p_{a_\ell}, \mathrm{proj}(p_{a_\ell}^-, L)) = \mathrm{dist}(p_{a_\ell}, L).$$

So we can get an upper bound of $|X_\ell| \le \frac{\mathrm{cost}(P,L)}{s} \le \frac{\beta \cdot \mathcal{OPT}}{s}$ for the random variable $|X_\ell|$. Note that we upper bound $|X_\ell|$ by $\frac{\beta \cdot \mathcal{OPT}}{s}$ not $\frac{\beta \cdot Z}{s}$. Now similar to Example 1 we use the additive Hoeffding with $M = \beta \cdot \mathcal{OPT}$, $t = \epsilon \cdot \mathcal{OPT}$, and $m = s$ to show that for the sample set $S_1 = \{(p_{a_\ell}, \mathrm{proj}(p_{a_\ell}^-, L)) | a_\ell \in A\}$ of size $O(\frac{\beta^2}{\epsilon^2} \log(2/\delta))$ we get

$$\mathbf{Pr}\big[|Z - \mathrm{cost}(S_1, L'')| \le \epsilon \cdot \beta \cdot \mathcal{OPT} \le \epsilon \cdot \beta \cdot \mathrm{cost}(P, L'')\big] \ge 1 - \delta,$$

where $\mathrm{cost}(S_1, L'') = \sum_{a_\ell \in A} w(a_\ell) \left(\mathrm{dist}(p_{a_\ell}, L'') - \mathrm{dist}(\mathrm{proj}(p_{a_\ell}^-, L), L'')\right)$.

In the general case of the j-subspace problem, the L, L'' are j-subspaces but the whole idea of the projection is the same. In fact, the above idea develops a dimensionality reduction technique for the j-subspace problem where we project the points onto a low-dimensional subspace and approximate the *difference* between the projected points and the original point set.

In order to obtain an unbiased estimator to approximate $\mathrm{cost}(P, L'')$ within $(1+\epsilon)$-factor, we apply our dimensionality reduction on $\mathrm{proj}(P, L)$ recursively using the fact that for points on an $(i+1)$-dimensional space it suffices to estimate the weighted sum of distances to an i-dimensional subspace even if one is interested in the sum of distance to a j-dimensional subspace.

This recursion is applied until $i = 0$ and the points project to the origin. This way, we obtain a small weighted sample set $S = \cup_{i=0}^{j} S_i$ such that for an arbitrary query j-space L'', we have

$$(1 - \epsilon) \cdot \text{cost}(P, L'') \leq \text{cost}(S, L'') \leq (1 + \epsilon) \cdot \text{cost}(P, L'').$$

This result is then used to construct a strong coreset by showing that the approximation guarantee holds simultaneously for all solutions from a certain grid near the input points.

Using the unbiased estimator and a (j, ϵ)-approximate centroid set for the j-subspace problem we get a weak coreset that helps us to get a linear time $(1 + \epsilon)$-approximation algorithm for j-subspace problem.

Using the merge and reduce method we maintain our coresets to obtain a streaming algorithm. However, the presence of negative points makes the process of maintaining a coreset harder. The problem is that the sum of the absolute weights of the coreset is about three times the size of the input point set. If we now apply our coreset construction several times (as is required during merge and reduce), we blow up the sum of absolute weights with each application by a constant factor. This blow-up, together with the fact that we have to estimate the difference between positively and negatively weighted points, cannot be controlled as well as in the case of the merge and reduce approach, and requires taking larger sample sizes with every merge step.

At the end we mention that although we apply the dimensionality reduction here in the context of subspaces, we remark that this technique can be easily generalized. In fact,we can replace the subspace by any closed set on which we can efficiently project points.

For example, we can easily extend the dimensionality reduction method to geometric clustering problems where the centers are low dimensional objects. We can also use the technique for any problem where we are trying to minimize the sum of distances to manifolds [BN03, BN04] as it for example might occur in the context of kernel methods [BN03, BN04].

3.3 $O(\log n)$-Pass L_p-Sampler

Think about Definition 1 once again. We would like to estimate a number $Z = Z_1 + \cdots + Z_n$ without computing all the summands.

Having a probability distribution $r_1, \cdots, r_n$ where $r_i \geq \frac{1}{\gamma}\frac{Z_i}{Z}$ for some constant $\gamma > 0$ the idea was to sample the summands according to r_i. However, in some settings finding a reasonable distribution r_i can be very hard. As an example consider the setting of insertion-only streaming model, see Section 2.5.1.

Recall that we have a vector a of length n where the i-th coordinate is represented by $a_i \geq 0$. We set $Z_i = (a_i)^p$ and $Z = F_p(a) = \sum_{i=1}^{n}(a_i)^p$ for some $p > 0$. The $F_p(a)$ is called the p-th frequency moment of a. In the insertion-only streaming model as we defined in Section 2.5.1 we are given a stream $\mathcal{S} = (b_1, \cdots, b_k, \cdots, b_m)$ of indexes in $[n]$ where $b_k \in [n]$, indicating that the b_k-th coordinate a_{b_k} of a should be incremented by 1. Our goal is again to estimate Z.

In this setting sampling according to probabilities $r_i = (1 \pm \epsilon)\frac{Z_i}{Z}$ is a non-trivial task. However we can still sample Z_i's according to $\frac{a_i}{\sum_{i=1}^{n} a_i}$ and estimate Z within factor $(1 \pm \epsilon)$ using the following known result due to Alon, Matias, and Szegedy [AMS96] which is weaker than the distribution $r_i \geq \frac{Z_i}{\gamma Z} = \frac{1}{\gamma}\frac{a_i^p}{\sum_{i=1}^{n}(a_i)^p}$ that we are interested in.

AMSSAMPLING

(1) For $i = 1, \cdots, s_2 = 2\log(1/\delta)$

(a) For $j = 1, \cdots, s_1 = 8pn^{1-1/p}/\epsilon^2$

i. Choose a random number b_k of the stream $\mathcal{S}$ uniformly at random.

ii. Let $b_k = \ell \in [n]$.

iii. Let $r = |\{q : q \geq k, b_q = \ell\}|$ be the number of occurrences of ℓ among the members of the stream $\mathcal{S}$ following b_k (inclusive).

iv. Let $X_{i,j} = m(r^p - (r-1)^p)$.

(b) Let $Y_i = \frac{\sum_{i=1}^{s_1} X_{i,j}}{s_1}$.

(2) Output $Y = \text{median}(Y_1, \cdots, Y_{s_2})$ as an estimator to $Z = F_p(a)$.

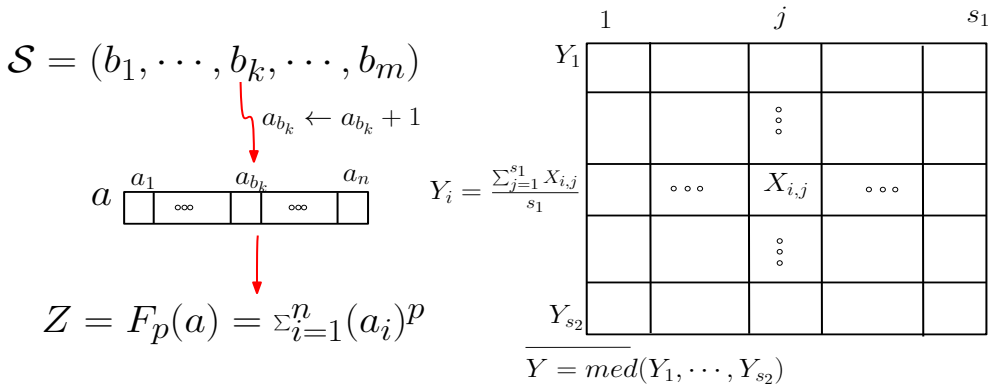


Fig. 3.4: The streaming model on the left and AMS Sampling on the right.

Lemma 25 *[AMS96] Let $p > 0, 0 < \epsilon < 1, \delta > 0$. Given a stream $\mathcal{S} = (b_1, \cdots, b_k, \cdots, b_m)$ of indexes in $[n]$ where $b_k \in [n]$, indicating that the b_k-th coordinate a_{b_k} of a should be incremented by 1, Algorithm* AMSSampling *using $O(\frac{p\log(1/\delta)}{\epsilon^2} n^{1-1/p}(\log n + \log m))$ memory bits returns a number Y for which we have*

$$\mathbf{Pr}[|Z - Y| \geq \epsilon \cdot Z] = \mathbf{Pr}[|F_p(a) - Y| \geq \epsilon \cdot F_p(a)] \leq 1 - \delta.$$

Proof. [From [AMS96]] We prove the lemma in the same line of the second proof that we had for Lemma 24 using Chebyshev's inequality. Fix $X_{i,j}$. Let $X = X_{i,j}$. In order to apply Chebyshev's inequality we need to obtain the expectation and the variance of the random

variable X. The expectation is

$$\begin{aligned}\mathbf{E}[X] &= \frac{m}{m}[(1^p + (2^p - 1^p) + \cdots + (m_1^p - (m_1 - 1)^p)) + \cdots + \\ &\qquad (1^p + (2^p - 1^p) + \cdots + (m_n^p - (m_n - 1)^p))] \\ &= \sum_{i=1}^{n} (a_i)^p = F_p(a).\end{aligned}$$

To estimate the variance $Var[X] = \mathbf{E}[X^2] - (\mathbf{E}[X])^2$ of X we bound $\mathbf{E}[X^2]$

$$\begin{aligned}\mathbf{E}[X^2] &= \frac{m^2}{m}[(1^{2p} + (2^p - 1^p)^2 + \cdots + (m_1^p - (m_1 - 1)^p)^2) + \cdots + \\ &\qquad (1^{2p} + (2^p - 1^p)^2 + \cdots + (m_n^p - (m_n - 1)^p)^2)] \\ &\le \frac{m^2}{m}[(k1^{2p-1} + k2^{p-1}(2^p - 1^p) + \cdots + km_1^{p-1}(m_1^p - (m_1 - 1)^p)) + \cdots + \\ &\qquad (k1^{2p-1} + k2^{p-1}(2^p - 1^p) + \cdots + km_1^{p-1}(m_n^p - (m_n - 1)^p))] \\ &\le mk[m_1^{2p-1} + m_2^{2p-1} + \cdots + m_n^{2p-1}] \\ &= kmF_{2p-1}(a) = kF_1(a)F_{2p-1}(a) \\ &\le kn^{1-1/p} \cdot (F_p(a))^2,\end{aligned}$$

where in the first inequality we use the fact that for any numbers $a > b > 0$ we have

$$a^p - b^p = (a - b)(a^{k-1} + a^{k-2}b + \cdots + ab^{k-2} + b^{k-1}) \le (a - b) \cdots ka^{k-1}$$

and for the last inequality we use the following claim.

Claim 26 *[AMS96] For every n positive reals $m_1, \cdots, m_n$ we have*

$$(\sum_{i=1}^{n} m_i)(\sum_{i=1}^{n} m_i^{2p-1}) \le n^{1-1/p} \cdot (\sum_{i=1}^{n} m_i^p)^2.$$

We also have that $\mathbf{E}[Y_i] = \mathbf{E}[X] = F_p(a)$ and

$$Var[Y_i] \le Var[X]/s_1 \le \mathbf{E}[X^2]/s_1 \le kF_1(a)F_{2p-1}(a)/s_1 \le kn^{1-1/p} \cdot (F_p(a))^2/s_1.$$

Now we can apply Chebyshev's inequality to get

$$\mathbf{Pr}[|Y_i - F_p(a)| \ge \epsilon \cdot F_p(a)] \le \frac{Var[Y_i]}{\epsilon^2 \cdot (F_p(a))^2} \le \frac{kn^{1-1/p} \cdot (F_p(a))^2}{s_1\epsilon^2 \cdot (F_p(a))^2} \le 1/8.$$

It follows that the probability that a single Y_i deviates from $F_p(a)$ by more than $\epsilon \cdot F_p(a)$ is at most $1/8$ and hence by standard Chernoff bound as we saw in the second proof of Lemma 24, the probability that more than $s_2/2$ of the variables Y_i deviates by more than $\epsilon \cdot F_p(a)$ from $F_p(a)$ is at most δ. In case that this does not happen, the median of Y_i supplies a good estimator to the required quantity $F_p(a)$, as needed. □

There are two issues with the above sampling process. The first issue is Algorithm AMSampling cannot be extended to the turnstile model, see Section 2.5.2. In this model, known $(1 \pm \epsilon)$-estimators for $Z = F_p(a)$ are due to Alon, Matias, and Szegedy [AMS96] for integers $p \leq 2$ and Indyk and Woodruf [IW05] for $p > 2$. However, neither these two estimators nor the estimator of AMSampling give a distribution r_i that we are interested in.

The second issue with Algorithm AMSampling is it does not use the optimal space upper bound. It has been shown in [BYJKS02, CKS03] that any algorithm that can estimate Z within factor $(1 \pm \epsilon)$ for $p > 2$ in the read only or turnstile streaming model needs a space lower bound of $\Omega(n^{1-2/p})$ bits. The sampling of AMSSampling offers a space of $O(\frac{p \log(1/\delta)}{\epsilon^2} n^{1-1/p}(\log n + \log m))$ memory bits which is far from the lower bound.

Interestingly enough, in Section 9.1 we show that if we can sample according to a distribution $\frac{1}{\gamma} \cdot \frac{|a_i|^q}{\sum_{i=1}^n |a_i|^q}$ for $q \in [0, 2], \gamma = (1 \pm \epsilon)$ we can $(1 \pm \epsilon)$-estimate $Z = \sum_{i=1}^n Z_i = \sum_{i=1}^n |a_i|^p$, for $p > 2$ in the turnstile streaming model using a space of $n^{1-2/p} \cdot \mathrm{poly}(\epsilon^{-1} \log n)$ memory bits which is optimal with respect to the term $n^{1-2/p}$.

Thus our goal is to sample the coordinate i with probability $r_i = (1 \pm \epsilon)\frac{|a_i|^p}{\sum_{i=1}^n |a_i|^p}$ for $p \in [0, 2]$. In order to simplify the problem we relax a little bit the problem and allow that the r_i have a very small additive error n^{-C} for an arbitrarily large constant C. We call this new problem L_p*-sampler* which is defined formally as follows:

Definition 27 (L_p-sampler) *Given a stream S of $m = \mathrm{poly}(n, M)$ updates of the form (i, x), where $i \in [n]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$, indicating that the i-th coordinate a_i of a should be incremented by x, an L_p-sampler for $p \in [0, 2]$ outputs the i-th coordinate with probability in the interval*

$$\left[(1-\epsilon)\frac{|a_i|^p}{F_p(a)} - n^{-C}, (1+\epsilon)\frac{|a_i|^p}{F_p(a)} + n^{-C}\right],$$

where $F_p(a) = \sum_{i=1}^n |a_i|^p$ and for an arbitrarily large constant C.

In Chapter 7 we use the $O(\log n)$-pass variant of the merge and reduce method to obtain an $O(\log n)$-pass L_p-sampler. The idea is as follows: Let F_p-Estimation be the optimal-space algorithm for estimating the F_p-value of a vector due to Kane *et al* [KNW10]. Imagine we build a binary tree on the vector a. Given a stream $\mathcal{S}$ corresponding to the vector a, we think of it as two interleaved streams $\mathcal{S}_L$ and $\mathcal{S}_U$, where $\mathcal{S}_L$ consists of the subsequence of updates to the first $n/2$ coordinates of the vector a, and $\mathcal{S}_U$ the subsequences consisting of the updates to the remaining coordinates. Denote the vector consisting of the lower $n/2$ coordinates of a by a^L, and the vector consisting of the upper $n/2$ coordinates of a by a^U.

We run F_p-Estimation($\mathcal{S}_L, a^L, \eta, \delta$) and F_p-Estimation($\mathcal{S}_U, a^U, \eta, \delta$) independently and in parallel with error parameter $\eta = \Theta(\epsilon / \log n)$ and failure probability $\delta = n^{-C}$. Assuming both algorithms succeed, we obtain numbers L, U with

$$L \in [(1-\eta)\|a^L\|_p, (1+\eta)\|a^L\|_p],$$

$$U \in [(1-\eta)\|a^U\|_p, (1+\eta)\|a^U\|_p].$$

We then recurse on the lower $n/2$ coordinates with probability $L/(L+U)$, and recurse on the upper $n/2$ coordinates with probability $U/(L+U)$. After $\log n$ recursive steps, an individual coordinate $i \in [n]$ will be sampled. Assuming F_p-Estimation never fails, fixing any $i \in [n]$, the probability that it is sampled is a telescoping product, putting it in the interval

$$\left[\frac{(1-\eta)^{\log n}|a_i|^p}{F_p(a)}, \frac{(1+\eta)^{\log n}|a_i|^p}{F_p(a)}\right],$$

which is contained in the interval

$$\left[\frac{(1-\epsilon)|a_i|^p}{F_p(a)}, \frac{(1+\epsilon)|a_i|^p}{F_p(a)}\right]$$

for sufficiently small η.

As non-uniform sampling had many applications in the k-means and the subspace problems, we are expecting that the L_p-sampler also has a lot of applications which is in fact true. In Chapter 9 we see that L_p-sampler leads to many improvements and a unification of well-studied streaming problems.

3.4 1-Pass L_p-Sampler

In the previous section we gave an overview of how to implement the L_p-sampler using $O(\log n)$ passes. However in most applications of the turnstile model the number of passes over the data is restricted to one and having $O(\log n)$ passes would be an unrealistic assumption. In order to implement the L_p-sampler in one pass we depart from the merge and reduce method and do it using the idea of finding heavy hitters in substreams that was first proposed by Indyk and Woodruff [IW05].

We conceptually divide the coordinates into classes

$$S_t = \{i \mid |a_i| \in [\eta^{t-1}, \eta^t)\},$$

where $\eta = 1 + \Theta(\epsilon)$. We say that a class S_t *contributes* if

$$|S_t|\eta^{pt} \geq \gamma F_p(a),$$

where $\gamma = \mathrm{poly}(\epsilon \log^{-1} n)$ is a sufficiently small parameter.

Let $h : [n] \to [n]$ be a hash function with some amount of limited independence. We form $r = O(\log n)$ substreams $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_r$, where $\mathcal{S}_0$ denotes the input stream and $\mathcal{S}_j$ denotes the stream of updates which pertain only to coordinates i for which $h(i) \leq n2^{-j}$. We say that such an i is a *survivor* (with respect to a given $\mathcal{S}_j$).

If S_t contributes, then there will be a substream $\mathcal{S}_j$ for which all survivors i in S_t are heavy hitters with respect to the p-norm, that is, $|a_i|^p \geq \gamma' \|a^j\|_p^p$, where a^j denotes the vector a restricted to the survivors in $\mathcal{S}_j$, and $\gamma' = \mathrm{poly}(\epsilon \log^{-1} n)$. The heavy hitters in each substream can be found with $\mathrm{poly}(\epsilon^{-1} \log n)$ space using known heavy hitter algorithms [CCFC02, GSS08], which work for any $p \in (0, 2]$, and even have a fast $\mathrm{poly}(\epsilon^{-1} \log n)$ update and reporting time [GSS08] in the turnstile model. We call the algorithm of [GSS08] HeavyHitters.

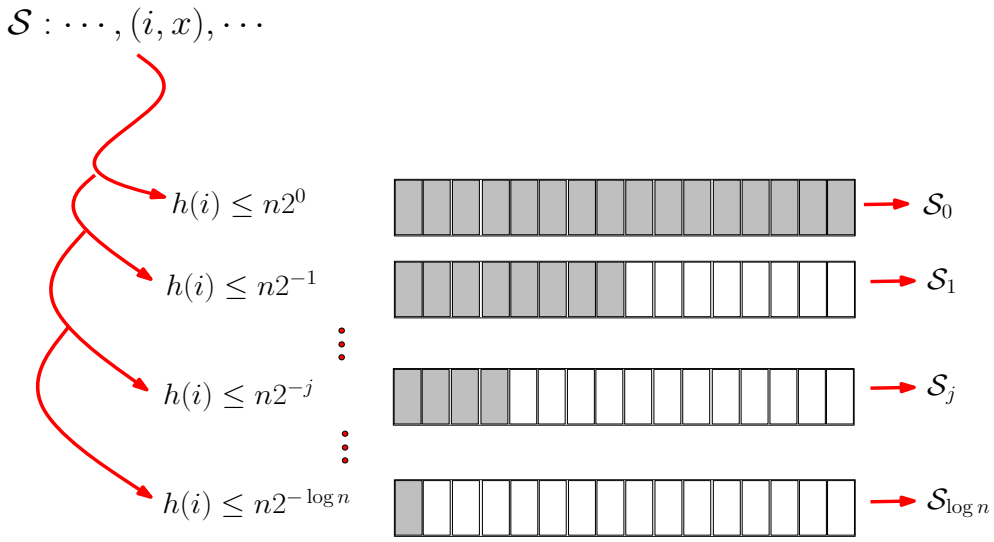


Fig. 3.5: Substreams shown as the percentage of the vector a which is preserved.

By zooming in on the appropriate substream and counting the number of survivors that are heavy hitters, we can estimate $|S_t|$ to within $(1 \pm \Theta(\epsilon))$ for all S_t that contribute. Notice that here we need the HeavyHitters algorithm to also provide $(1 \pm \Theta(\epsilon))$-approximations to the values $|a_i|$ for each heavy hitter i, which can be achieved by increasing the space by a $\mathrm{poly}(\epsilon^{-1} \log n)$ factor. We also need the values $|a_i|$ to not be close to the value $(1 + \Theta(\epsilon))^{t-1}$, as otherwise we might mis-classify i as belonging to S_t when in fact it belongs to S_{t-1}.

For S_t that do not contribute, we only obtain an estimate E for which

$$0 \leq E \leq (1 + \Theta(\epsilon))|S_t|.$$

This sub-sampling approach is the basis of the algorithm of Indyk and David Woodruff [IW05] for estimating frequency moments.

As observed by Jayram and Woodruff [JW09] this approach yields a 1-pass $\Theta(\epsilon)$-additive-error L_p-sampler in the following way. We simply ignore the S_t that do not contribute, and let the $\tilde{s}_t$ be our approximations to the set sizes $|S_t|$ for contributing S_t. We sample a contributing S_t with probability

$$\frac{\tilde{s}_t \eta^{pt}}{\sum_{\text{contributing } S_t} \tilde{s}_t \eta^{pt}}.$$

Condition on the event that S_t is chosen, we output a heavy hitter found in the substream $\mathcal{S}_j$ which is in S_t and minimizes h_j. Note that this heavy hitter is one of the survivors of S_t and is used to estimate $|S_t|$ within factor $(1 \pm \epsilon)$. Now we use Theorem 9 which says that if h is sufficiently independent then it is also ϵ-min-wise independent, see Definition 8, to show that the heavy hitter which minimizes h is a random element of S_t, up to a relative $(1 \pm \epsilon)$-error. It is now straightforward to show that we output i with probability

$$(1 \pm \Theta(\epsilon)) \cdot \frac{|a_i|^p}{F_p(a)}.$$

The problem with the above approach is that it leads to an additive error rather than a relative error.

Indeed, items in non-contributing classes will *never* be sampled. This is problematic if the sampling algorithm is to be used in a subroutine that performs more than $\mathrm{poly}(\epsilon^{-1}\log n)$ samples, as some items that should be reported by an exact sampler will not be detected.

Our main novelty is the following idea. Suppose we *force* every class to contribute. If we could do this, then we could try to apply the above sampling procedure to obtain a L_p-sampler. To force each class to contribute, the idea is to inject new coordinates into the stream. That is, let $s = O(\epsilon^{-1}\log n)$ be the number of classes. For class S_t, we inject $\Theta(\epsilon)F_p(a)/(s(1+\epsilon)^{pt}))$ coordinates i for which $|a_i| \in [\eta^{t-1}, \eta^t)$. It follows that F_p changes by at most a $(1+\Theta(\epsilon))$ factor. Moreover, now $|S_t|\eta^{pt} = \Omega(\epsilon F_p(a)/s)$ for all t, and so provided $\gamma = O(\epsilon/s) = O(\epsilon^2/\log n)$, every class now contributes.

Notice that we do not know $F_p(a)$ in advance, but it suffices to guess a $(1+\Theta(\epsilon))$-approximation to it, and then verify our guess at the end of the stream by running an efficient F_p-approximation algorithm in parallel, taking only $\mathrm{poly}(\epsilon^{-1}\log n)$ bits of space, e.g., the space optimal algorithm of Kane *et al* [KNW10], or the earlier algorithms of Indyk [Ind06] and Li [Li08]. The number of guesses we need is only $O(\epsilon^{-1}\log n)$.

We now run the sampling algorithm above. If we obtain an injected coordinate, then we output FAIL, otherwise we output the coordinate and its approximate value returned by HeavyHitters. Notice that the injected items contribute at most an $O(\epsilon)$ mass to the F_p-value, so we output FAIL with probability at most $O(\epsilon)$. By repeating this procedure in parallel a logarithmic number of times, at least one of our samples will not be an injected coordinate with probability at least $1 - n^{-C}$ for an arbitrarily large constant $C > 0$.

Chapter 4

Related Work

This chapter surveys the known results about *random linear combination* 4.1, random linear combination in data stream so called *sketching* 4.2, and *coresets* 4.3. We also explain the way that we can solve the k-means and the subspace problems using random linear combination. For the sketching section we compare known results with our results; but for the rest we only mention the known results and one can compare them by reading the overview Chapter 3.

4.1 Johnson-Lindensrauss vs. non-uniform sampling

Let $Z_i = |a_i|^p$ for $p \in [0,2], i \in [n]$. In the previous chapter we explained how to solve the problem of approximating a sum without computing the summands (see Definition 1) using non-uniform sampling. A relaxed version of this problem can be solved using *random linear combination*. In this relaxed version that we call *estimating a sum* we drop the restriction that each estimator X_j for $1 \leq j \leq s$ depends only on one of the summands of Z and we are also able to compute all the summands. However, the number of estimators s should still be $O(\epsilon^{-2}\log(1/\delta))$. This general problem is formally defined as follows:

Definition 28 (Estimating a Sum) *Let $0 < \epsilon < 1, \delta > 0$. Let $p \in [0,2]$. Let a denote a n-dimensional real vector. Assume Z is the sum of n numbers, $Z_1 = |a_1|^p, \cdots, Z_n = |a_n|^p$ i.e., $Z = Z_1 + \cdots + Z_n$. Find an estimator $X = X_1 + \cdots + X_s$ for $s \geq O(\epsilon^{-2}\log(1/\delta))$ such that*

$$\mathbf{Pr}[(1-\epsilon)Z \leq X \leq (1+\epsilon)Z] \geq 1-\delta.$$

Here we focus on $p = 2$; however the analysis for general $p \in [0,2]$ is the same. In the previous chapter we solved this problem using non-uniform sampling, where we sampled the number Z_i using a probability which depends on the contribution of Z_i to the sum Z. In this chapter we want to solve this problem using the random linear combination offered by *Johnson-Lindenstrauss Lemma* (see Theorem 32). The idea is as follows. We define a suitable probability distribution $\mathcal{F}$ on he set of all linear maps from $\mathbb{R}^n \to \mathbb{R}^s$. Then we prove the following statement which in fact states a *dimensionality reduction* for $p = 2$, i.e., Euclidean distance.

Statement 29 (Dimensionality Reduction in ℓ_2) *Let a denote an arbitrary vector in $\mathbb{R}^n$. Let $T : \mathbb{R}^n \to \mathbb{R}^s$ denote a random linear mapping drawn from a suitable probability distribution $\mathcal{F}$ to be determined later. Let*

$$X_1 = |T(a)_1|^2, \cdots, X_s = |T(a)_s|^2.$$

Then we have

$$\mathbf{Pr}[(1-\epsilon)Z \leq X_1 + \cdots + X_s \leq (1+\epsilon)Z] \geq 1 - \delta.$$

The random linear map T is usually chosen as the orthogonal projection on a random s-dimensional subspace of $\mathbb{R}^n$. In particular, we define a $s \times n$ matrix A and put the s orthonormal vectors of this subspace as the s rows of S. In this way, T would be $T(a) = Sa$ multiplied by $\sqrt{n/s}$ for proper scaling. What is interesting is that Indyk and Motwani [IM98] noted that the orthogonality condition can be dropped and in their proof of Statement 29 they choose the entries of S as *independent* random variables with the standard normal distribution $N(0,1)$ which is much easier to generate.

Later Achlioptas [Ach03] showed that the entries of S can be chosen from a much more efficient distribution. Namely he proved that the entries of S attain value 0 with probability $2/3$ and values $+1$ and -1 with probability $1/6$ each. This setting is in fact 3 times faster the setting proposed by Indyk and Motwani, since S is sparse, i.e., only about one third of the entries are nonzero.

An ingenious idea was further developed by Ailon and Chazelle [AC09] where they showed that the entries of S attain value 0 with probability $1 - q$ and a value drawn from the normal distribution with zero mean and variance $1/q$ with probability q where $q = \frac{(\log n)^{O(1)}}{n}$. An obstacle they had to overcome was since now the matrix S is very sparse, with the fraction of nonzero entries tending to 0, the estimator $X_1 + \cdots + X_s$ is not concentrated around its mean which is Z for some sparse vectors a like $a = (1, 0, \cdots, 0)$. Therefore, they premultiplied the vector a by a random Fourier transform to distribute the mass of a over many coordinates of a. In this way they were able to find a very sparse matrix S for the linear mapping T which can be implemented efficiently since now in expectation only $\epsilon^{-2} \cdot (\log n)^{O(1)}$ entries of S are non-zero. See also [DKS10] and [KN10] for new results in this line of research.

Matousek [Mat08] shows that all these aforementioned distributions can be unified if we define the entries of S as independent random variables with zero mean and unit variance, and a subgaussian tail. See Figure 4.1 for a summary of all known distributions for the dimensionality reduction in ℓ_2 and Johnson-Lindenstrauss Lemma.

Definition 30 (Subgaussian upper tail) *Let Y be a real random variable with $\mathbf{E}[Y] = 0$. We say Y has a subgaussian upper tail if there exists a constant $b > 0$ such that for all $\lambda > 0$,*

$$\mathbf{Pr}[Y > \lambda] \leq e^{-b\lambda^2}.$$

We say that Y has a subgaussian upper tail up to λ_0 if the previous bound holds for all $\lambda \leq \lambda_0$. We say that Y has a subgaussian tail if both Y and $-Y$ have subgaussian upper tails. Let $Y_1, \cdots, Y_n$ be a sequence of random variables. We say that they have a *uniform subgaussian tail* if all of them have subgaussian upper tails with the same constant b.

Now we are ready to prove Statement 29. The proof is due to Matousek [Mat08]. First we rephrase the statement using matrix notations:

Ref.	Distribution	Sparsity	Independency	Time to Compute
[IM98]	$N(0,1)$	———	Fully	$O(\epsilon^{-2}n)$
[Ach03]	Pr(s[i,j]=1)=Pr(s[i,j]=-1)=1/6 Pr(s[i,j]=0)=2/3	2/3	Fully	$O(\epsilon^{-2}n)$
[AMS96]	Pr(s[i,j]=1)=Pr(s[i,j]=-1)=1/2	———	4-wise	$O(\epsilon^{-2}n)$
[Mat08]	E[s[i,j]]=0, Var[s[i,j]]=1 Subgaussian Tail	———	Fully	$O(\epsilon^{-2}n)$
[AC09]	Pr(s[i,j]=N(0,q))=q Pr(s[i,j]=0)=1-q $q=(\log(n))^{O(1)}/n$	$O(\epsilon^{-2}(\log n)^{O(1)})$	Fully	$O(\epsilon^{-2}(\log n)^{O(1)})$
[Mat08]	$Pr(s[i,j]=1/\sqrt{q})=q/2$ $Pr(s[i,j]=-1/\sqrt{q})=q/2$ Pr(s[i,j]=0)=1-q , $q=(\log(n/\epsilon))/n$	$O(\epsilon^{-2}(\log n))$	Fully	$O(\epsilon^{-2}(\log n))$

Fig. 4.1: A summary of known distributions for the dimensionality reduction in ℓ_2 and Johnson-Lindenstrauss Lemma. *Time to compute* is the time to multiply the random matrix and one vector a, where the vector is in $\mathbb{R}^n$. The $\tilde{O}(\cdot)$ in the *Time to Compute* hides a factor of $\log(2/\delta)$. The $s[i,j]$ means for every entry i,j of random matrix S. The *Sparsity* measure means how many of entries of S are non-zero in expectation.

Theorem 31 (Dimensionality Reduction in ℓ_2: [Mat08]) *Let a denote an arbitrary vector in $\mathbb{R}^n$. Let $Z_i = |a[i]|^2$ for $i \in [n]$ and $Z = Z_1 + \cdots + Z_n$. Let $s = C\epsilon^{-2}\log(2/\delta)$ where C is a suitable constant. Let $T : \mathbb{R}^n \to \mathbb{R}^s$ denote a random linear mapping defined by*

$$T(a)_i = \frac{1}{\sqrt{s}} \sum_{j=1}^{n} s[i,j] \cdot a[j], i = 1, \cdots, s,$$

where the $s[i,j]$ are independent random variables with $\mathbf{E}[s[i,j]] = 0$, $\mathrm{Var}[s[i,j]] = 1$ and a uniform subgaussian tail. Let $X_1 = |T(a)_1|^2, \cdots, X_s = |T(a)_s|^2$ and $X = X_1 + \cdots + X_s$. Then we have

$$\mathbf{Pr}[(1-\epsilon)Z \leq X \leq (1+\epsilon)Z] \geq 1 - \delta.$$

The beauty of the estimator given by Theorem 31 in comparison to the estimator given by non-uniform sampling is, the random matrix S does not use any assumption or information of the underlying vector a which means we can apply it to any arbitrary vector $a \in \mathbb{R}^n$. For example, we can prove the Johnson-Lindenstrauss Lemma which informally states that if we project a point set P of size m onto a s-subspace spanned by the rows of S, where $s = O(\epsilon^{-2}\log m)$ then all distances between pairs of points will be approximated within factor $(1+\epsilon)$.

Theorem 32 (Johnson-Lindenstrauss Lemma: [JL84]) *Let $0 < \epsilon \leq 1/2$. Let $P = \{p_1, \cdots, p_m\}$ be a set of m points in $\mathbb{R}^n$. Let $s = C\epsilon^{-2}\log m$, where C is sufficiently large constant. Then linear mapping $T : \mathbb{R}^n \to \mathbb{R}^s$ of Theorem 31 with probability at least $1/2$ fulfills*

$$(1-\epsilon)\|p_i - p_j\| \leq \|T(p_i) - T(p_j)\| \leq (1+\epsilon)\|p_i - p_j\|$$

for all $i, j \in [m]$, where $\|\cdot\|$ denotes the Euclidean distance.

Proof. For every $i < j$ using linearity of T and Theorem 31 with $a = p_i - p_j$ we get that T fails to satisfy

$$(1-\epsilon)\|p_i - p_j\| \leq \|T(p_i) - T(p_j)\| \leq (1+\epsilon)\|p_i - p_j\|$$

with probability at most $1/m^2$. Thus the probability that any of the pairwise distances is distorted by T by more than $1 \pm \epsilon$ is at most $1/2$. Therefore, the random linear mapping T which is instantiated by random matrix S works with probability at least $1/2$. □

In order to prove Theorem 31 we first prove the following lemmas.

Lemma 33 *[Mat08] Let $Y_1, \cdots, Y_n$ be independent random variables, satisfying $\mathbf{E}[Y_i] = 0, \mathrm{Var}[Y_i] = 1$, and having a uniform subgaussian tail. Let $\alpha_1, \cdots, \alpha_n$ be real coefficients satisfying $\alpha_1^2 + \cdots + \alpha_n^2 = 1$. Then the sum*

$$Y = \alpha_1 Y_1 + \cdots + \alpha_n Y_n$$

has $\mathbf{E}[Y] = 0, \mathrm{Var}[Y] = 1$, and having a subgaussian tail.

Proof. We need the following Lemmas.

Lemma 34 *[Mat08] Let* X *be random variable with* $\mathbf{E}[X] = 0$*. If* $\mathbf{E}[e^{uX}] \leq e^{Cu^2}$ *for some constant* C *and for all* $u > 0$*, then* X *has a subgaussian upper tail. If* $\mathbf{E}[e^{uX}] \leq e^{Cu^2}$ *holds for all* $u \in (0, u_0)$*, then* X *has a subgaussian upper tail up to* $2Cu_0$*.*

Proof. For all $u \in (0, u_0)$ and all $t \geq 0$ and using the Markov inequality we have

$$\begin{aligned}\mathbf{Pr}[X \geq t] &= \mathbf{Pr}[e^{uX} \geq e^{ut}] \\ &\leq e^{-ut}\mathbf{E}[e^{uX}] \\ &\leq e^{-ut+Cu^2}\end{aligned}$$

For $t \leq 2Cu_0$ we can set $u = t/(2C)$, use the above estimate and obtain $\mathbf{Pr}[X \geq t] \leq e^{-t^2/(4C)}$. □

Lemma 35 *[Mat08] If* $\mathbf{E}[X] = 0$*,* $\mathrm{Var}[X] = \mathbf{E}[X^2] = 1$ *and* X *has a subgaussian upper tail, then* $\mathbf{E}[e^{uX}] \leq e^{Cu^2}$ *for all* $u > 0$ *and some constant* C*.*

Proof. Let F be the distribution function of X, that is, $F(t) = \mathbf{Pr}[X < t]$. We have $\mathbf{E}[e^{uX}] = \int_{-\infty}^{\infty} e^{ut} dF(t)$. We split the integration into two subintegrals, corresponding to $ut \leq 1$ and $u \geq 1$. For the first case we use this fact that for all $x \leq 1$, $e^x \leq 1 + x + x^2$ to get

$$\begin{aligned}\int_{-\infty}^{1/u} e^{ut} dF(t) &\leq \int_{-\infty}^{1/u} (1 + ut + u^2t^2) dF(t) \\ &\leq \int_{-\infty}^{\infty} (1 + ut + u^2t^2) dF(t) \\ &= 1 + ut\mathbf{E}[X] + u^2\mathbf{E}[X^2] = 1 + u^2.\end{aligned}$$

For the second case we estimate the integral by sum as follows:

$$\begin{aligned}\int_{1/u}^{\infty} e^{ut} dF(t) &\leq \sum_{k=1}^{\infty} e^{k+1}\mathbf{Pr}[X \geq k/u] \\ &\leq \sum_{k=1}^{\infty} e^{2k} e^{-bk^2/u^2} \\ &\leq \sum_{k=1}^{\infty} e^{k(2-bk/u^2)}.\end{aligned}$$

which is upper bounded by $e^{O(u^2)}$ by elementary calculus. Thus $\mathbf{E}[e^{uX}] \leq e^{O(u^2)}$ as desired. □

Using the above Lemmas we prove Lemma 33. First observe that by linearity of expectation $\mathbf{E}[Y] = 0$. Next, since the variance is additive for independent random variables we get

$$\mathrm{Var}[Y] = \alpha_1^2 \mathrm{Var}[Y_1] + \cdots + \alpha_n^2 \mathrm{Var}[Y_n] = \alpha_1^2 + \cdots + \alpha_n^2 = 1.$$

By Lemma 35, $\mathbf{E}[e^{uY_i}] \leq e^{Cu^2}$ so we get

$$\mathbf{E}[e^{uY}] \leq \prod_{i=1}^{n} \mathbf{E}[e^{u\alpha_i X_i}] \leq e^{Cu^2(\alpha_1^2+\cdots+\alpha_n^2)} = e^{Cu^2},$$

Which proves that Y has a subgaussian upper tail using Lemma 34. □

A pretty interesting distribution that satisfies the criteria of Lemma 33 is when all Y_i have the standard normal distribution. Then by the 2-stability of the normal distribution which is defined below we get that Y has the standard normal distribution.

Definition 36 (Stable distributions) *A distribution D over $\mathbb{R}$ is called p-stable, if there exists $p \geq 0$ such that for any n real numbers $b_1, \cdots, b_n$ and i.i.d. variables $Y_1, \cdots, Y_n$ with distribution D, the random variable $\sum_i b_i \cdot Y_i$ has the same distribution as the variable $(\sum_i |b_i|^p)^{1/p} Y$, where Y is a random variable with distribution D.*

It is known that stable distributions exist for any $p \in (0, 2]$. In particular, a normal distribution D_G defined by the density function $g(x) = \frac{1}{\sqrt{2\pi}} \cdot e^{-x^2/2}$ is 2-stable.

We also need the following lemma to prove Theorem 31.

Lemma 37 *[Mat08] Let $s \geq 1$. Let $Y_1, \cdots, Y_s$ be independent random variables, satisfying $\mathbf{E}[Y_i] = 0, \mathrm{Var}[Y_i] = 1$, and having a uniform subgaussian tail. Then*

$$U = \frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2 - s)$$

has a subgaussian tail up to $\sqrt{k}$.

Proof. The proof of the following lemma is similar to the proof of Lemma 35 and we do not give it here.

Lemma 38 *[Mat08] Let Y denote a random variable for which we have $\mathbf{E}[Y] = 0$, $\mathrm{Var}[Y] = 1$ and Y has a subgassian tail. Then there are constants C, u_0 such that for all $u \in [0, u_0]$ we have $\mathbf{E}[e^{u(Y^2-1)}] \leq e^{Cu^2}$ and $\mathbf{E}[e^{u(1-Y^2)}] \leq e^{Cu^2}$.*

Having this lemma in our hands we prove Lemma 37.

For $U = \frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2 - s)$ and $u \in [0, u_0]$ and with the help of Lemma 38 we calculate

$$\begin{aligned}
\mathbf{E}[e^{uU}] &= \mathbf{E}[e^{\frac{u}{\sqrt{s}}(Y_1^2+\cdots+Y_s^2-s)}] \\
&= (\mathbf{E}[e^{\frac{u}{\sqrt{s}}(Y^2-1)}])^s \\
&= (e^{Cu^2/s})^s \\
&= e^{Cu^2}.
\end{aligned}$$

Lemma 34 then implies that U has a subgaussian upper tail up to $C\sqrt{s}$. The calculation for the lower tail is identical. □

Finishing the proof of Theorem 31

Assume we normalize a so that a is now a unit vector, i.e., $a^T \cdot a = Z = 1$.

Recall that $X_1 = |T(a)_1|^2, \cdots, X_s = |T(a)_s|^2$ and $X = X_1 + \cdots + X_s$. Let

$$Y_1 = \sum_{j=1}^{n} s[1,j] \cdot a[j], \cdots, Y_s = \sum_{j=1}^{n} s[s,j] \cdot a[j].$$

Then

$$X = \frac{1}{s}(Y_1^2 + \cdots + Y_s^2) = \frac{1}{\sqrt{s}} \cdot \frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2).$$

The fact that we need to prove is that with probability $\geq 1 - \delta$ we have $(1-\epsilon)Z \leq X \leq (1+\epsilon)Z$ which is equivalent to $|X - 1| \leq \epsilon$. This is equal to

$$|\frac{1}{\sqrt{s}} \cdot \frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2) - 1| \leq \epsilon$$

or

$$|\frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2 - s)| \leq \epsilon \cdot \sqrt{s}.$$

Next for $U = \frac{1}{\sqrt{s}}(Y_1^2 + \cdots + Y_s^2 - s)$ and $\epsilon \leq 1/2$ since $\epsilon \cdot \sqrt{s}$ is in the allowed range of Lemma 37 we get that

$$\mathbf{Pr}[U \geq \epsilon \cdot \sqrt{s}] \leq e^{-b(\epsilon \cdot \sqrt{s})^2} = e^{-b\epsilon^2 \cdot C\epsilon^{-2} \log(2/\delta)} \leq \delta/2.$$

The calculation showing that

$$\mathbf{Pr}[U \leq -\epsilon \cdot \sqrt{s}] \leq \delta/2$$

is almost the same.

Remark 39 *It is interesting to see where exactly we need the condition of being independent. We observe that in the prove of Lemmas 33 and 37 we needed the variables to be fully independent. Being fully independent in data streams is difficult to provide. This is due to the fact that in order to have all random variables to be fully independent we need to store a matrix of size* $O(sn)$ *which is larger than the length of the vector* a*. However, in Section 4.2 we see that if we can replace this being fully independent condition for entries of* S *with some limited independency then we are able to store and maintain* S *implicitly in a space of* $O(\log n)$ *bits.*

In Sections 3.1 and 3.2 we studied the k-means and the subspace clusterings as applications of non-uniform sampling. Next we explain that these two problems can also be solved using random linear combination or better to say dimensionality reduction offered by Theorem 31. The k-means result is due to Ostrovsky and Rabani [OR02], and the subspace clustering result is due to Sarlos [Sar06]. See Figure 4.2 for a comparison between random linear combination and non-uniform sampling results in clustering.

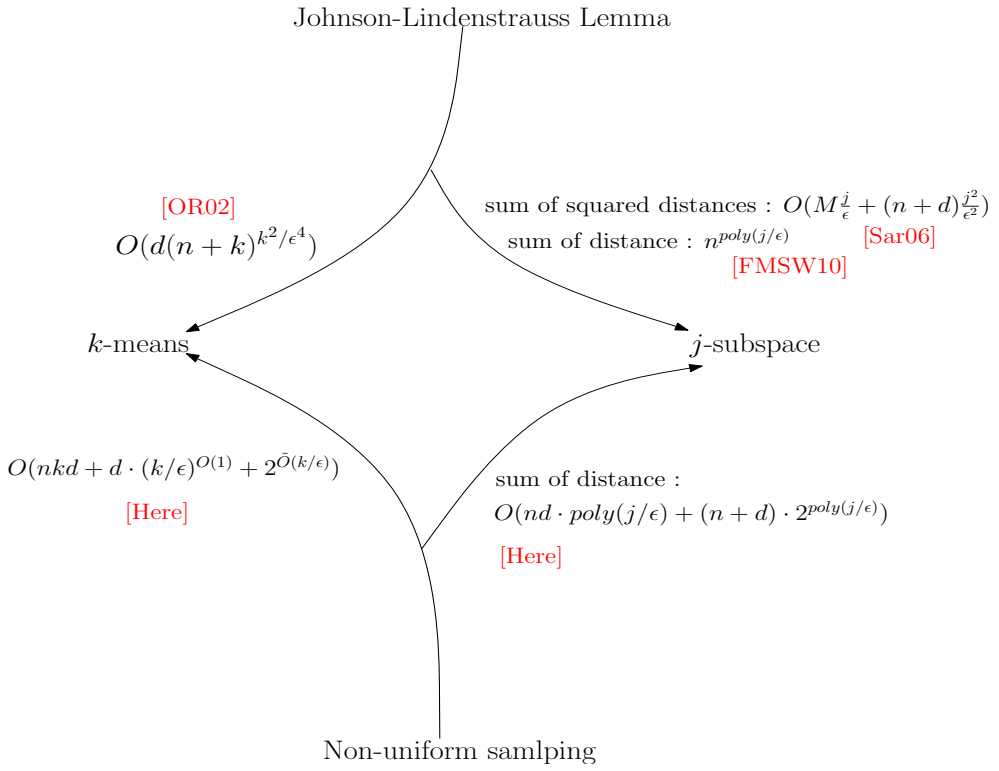


Fig. 4.2: A comparison between random linear combination and non-uniform sampling results in clustering. Here M denote the number of non-zero entries of a matrix whose rows are points in P.

4.1.1 The k-Means Clustering [OR02]

We first consider the problem in the Hamming cube (Z_2^d, H_d), where Z_2^d is a d-dimensional vector over field Z_2 and H_d is the d-dimensional Hamming distance. Note that once again the dimension d is not constant. The k-means algorithm (depicted in Figure 4.3) for $k = 2$ proceeds as follows (see [OR02] for general k). Let P denote a point set of size n in the Hamming cube (Z_2^d, H_d). We first guess the distance ℓ between the two optimal centers a_1, a_2 (by enumerating over all d possible values). We then project the data points into a space $Z_2^{d'}$ for $d' = O(\log n)$. In this low-dimensional space, we enumerate over all $2^{2d'}$ possible locations for the projections of the optimal projected cluster centers b_1, b_2.

Each choice induces a partition of the data set into two subsets. Each subset is associated with a cluster center projection and contains all the points whose projections are closer to this center's projection than to the other center's projection, ties broken arbitrarily. We check each possible partition in the original space by computing the best cluster center for each subsets and summing up the distances from the points to their assigned centers. Finally we output the best partition over all guesses of ℓ and over all guesses of the cluster centers projections.

The projection in the above algorithm is done using a variation of Theorem 31. In particular, for the current guess ℓ of distance between the two optimal centers a_1, a_2, we choose a random matrix S of size $d' \times d$ where the entries of S are independent, identically distributed (i.i.d.) random $0/1$ variables with $\mathbf{Pr}[s[i,j] = 1] = \epsilon^2/\ell$ and $\mathbf{Pr}[s[i,j] = 0] = 1 - \epsilon^2/\ell$.

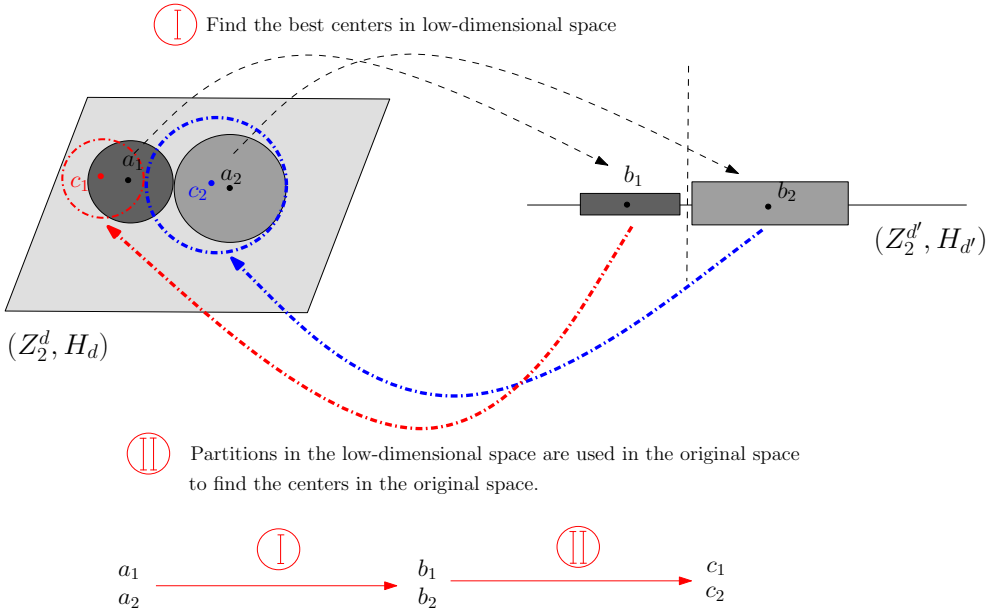


Fig. 4.3: k-means using JL-Lemma.

We then prove that for $d' = O(\epsilon^{-4} \cdot \log n)$ the linear mapping furnished by S is $(\epsilon, \ell/2, \ell/(2\epsilon))$-distorted on $P \cup \{a_1, a_2\}$ which means that for every $x, y \in P \cup \{a_1, a_2\}$

(1) if $\text{dist}(x, y) < \ell/4$, then $\text{dist}'(Sx, Sy) < (1+\epsilon)\alpha \cdot \ell/4$;

(2) if $\text{dist}(x, y) > \ell/(2\epsilon)$, then $\text{dist}'(Sx, Sy) > (1-\epsilon)\alpha \cdot \ell/(2\epsilon)$;

(3) if $\ell/4 \leq \text{dist}(x, y) \leq \ell/(2\epsilon)$, then $(1-\epsilon)\alpha \cdot \text{dist}(x, y) \leq \text{dist}'(Sx, Sy) \leq (1+\epsilon)\alpha \cdot \text{dist}(x, y)$,

where dist is the d-dimensional Hamming distance, dist' is the d'-dimensional Hamming distance, and α is a scaling factor.

Intuitively, a $(\epsilon, \ell/2, \ell/(2\epsilon))$-distorted mapping approximately preserves distances between $\ell/4$ and $\ell/(2\epsilon)$, and further it doesn't shrink large distances too much and does not expand small distances too much.

Let ℓ denote the actual distance between the optimal centers a_1, a_2. For the projection step where the probability of being one for the entries of S is ϵ^2/ℓ, the S gives a linear $(\epsilon, \ell/2, \ell/(2\epsilon))$-distorted mapping on $P \cup \{a_1, a_2\}$. Using this mapping we then prove that if for a point $p \in P$ we have that $\text{dist}'(Sp, \min\{b_1, b_2\}) \leq \text{dist}'(Sp, \min\{Sa_1, Sa_2\})$, then $\text{dist}(p, \min\{S^- b_1, S^- b_2\}) \leq (1+\epsilon) \cdot \text{dist}(p, \min\{a_1, a_2\})$, where S^- is the inverse of the mapping S. Since b_1, b_2 are the optimal centers in $Z_2^{d'}$, i.e.,

$$\sum_{p \in P} \text{dist}'(Sp, \min\{b_1, b_2\}) \leq \sum_{p \in P} \text{dist}'(Sp, \min\{Sa_1, Sa_2\})$$

we get that

$$\sum_{p \in P} \text{dist}(p, \min\{S^- b_1, S^- b_2\}) \leq (1+\epsilon) \sum_{p \in P} \text{dist}(p, \min\{a_1, a_2\}).$$

On the other hand, since c_1, c_2 are the optimal centers in Z_2^d for the partitions corresponding to b_1, b_2 in $Z_2^{d'}$

$$\sum_{p \in P} \text{dist}(p, \min\{c_1, c_2\}) \leq \sum_{p \in P} \text{dist}(p, \min\{S^- b_1, S^- b_2\}).$$

Thus we prove that

$$\begin{aligned}\sum_{p \in P} \text{dist}(p, \min\{c_1, c_2\}) &\leq \sum_{p \in P} \text{dist}(p, \min\{S^- b_1, S^- b_2\}) \\ &\leq (1+\epsilon) \sum_{p \in P} \text{dist}(p, \min\{a_1, a_2\}).\end{aligned}$$

For the general Euclidean space $\mathbb{R}^d$ we first project the data set into a Hamming cube $(Z_2^{d'}, H_{d'})$ for $d' = \text{poly}(n, d, \epsilon^{-1})$ and then run the above algorithm on the projected data set.

Besides this result Ostrovsky and Rabani [OR02] found a generic framework using dimensionality reduction of Johnson-Lindenstrauss Lemma for the following problems:

- **k-median in the Hamming cube:**
  Given a point set P in a Hamming cube Z_2^d, find k centers in Z_2^d such that the sum of Hamming distances of points in P to these centers is minimum.

- **k-median in $\mathbb{R}^d$ endowed with Manhattan distance:**
  Given a point set P in a $\mathbb{R}^d$, find k centers in $\mathbb{R}^d$ such that the sum of $\|\cdot\|_1$ distances of points in P to these centers is minimum.

- **k-median in $\mathbb{R}^d$ endowed with Euclidean distance:**
  Given a point set P in a $\mathbb{R}^d$, find k centers in $\mathbb{R}^d$ such that the sum of $\|\cdot\|_2$ distances of points in P to these centers is minimum.

- **k-means in $\mathbb{R}^d$ endowed with Euclidean distance:**
  Given a point set P in a $\mathbb{R}^d$, find k centers in $\mathbb{R}^d$ such that the sum of squared $\|\cdot\|_2$ distances of points in P to these centers is minimum.

Theorem 40 *[OR02] For every $\delta > 0$ and for every $0 < \epsilon \leq 1/8$ the general scheme that we sketched above in time $O(d(n+k)^{k^2/\epsilon^4})$ finds a solution for the problems of k-median in Hamming cube, k-median in $\mathbb{R}^d$ endowed with Manhattan distance, k-median in $\mathbb{R}^d$ endowed with Euclidean distance, and k-means in $\mathbb{R}^d$ endowed with Euclidean distance such that the value of this solution is within a factor of $(1+8\epsilon)^2$ of the optimum, with probability at least $1 - n^\delta$.*

4.1.2 Subspace Clustering [Sar06]

It is known that a j-subspace that minimizes the sum of squared distances to a point set $P = \{p_1, \cdots, p_n\}$ is spanned by the top j right singular vectors of its matrix $A \in \mathbb{R}^{n \times d}$, where the rows $p_1, \cdots, p_n$ of this matrix correspond to the points in the point set P. The singular vectors can be computed using singular value decomposition (SVD) in time $O(min\left\{nd^2, n^2d\right\})$. The matrix obtained by projecting each point on this subspace then provides a matrix of low rank that well-approximates the original matrix.

A sequence of papers on the one j-subspace that minimizes the sum of squared distances has been initiated by the seminal work of Frieze, Kannan, and Vempala [FKV04, DRVW06, HP06, DV06, Sar06, DV07]. This work has focused on obtaining approximation algorithms in time $O(nd \cdot (j/\epsilon)^{O(1)})$, because the running time of SVD is too large for massive data sets.

The strongest results obtained so far has been recently proved by Deshpande and Vempala in [DV06, SV07] where they show that if we sample $(4j/\epsilon + 2j\log(j+1))$ points from the point set P or rows of A, interchangeably using $2(j+1)(\log(j+1)+1)$ passes in an adaptive manner [DRVW06], then their span contains a $(1+\epsilon)$-approximation to the optimum j-subspace. The running time and the space of this algorithm are

$$O\left(M(\frac{j}{\epsilon} + j^2\log j) + (n+d)\left(\frac{j^2}{\epsilon^2} + \frac{j^3\log j}{\epsilon} + j^4\log j\right)\right)$$

and $(n+d)\cdot(j/\epsilon + j^2\log j)$ respectively where M is the number of nonzero entries of A.

Later Deshpande and Varadarajan [DV07] extend some techniques that have been previously proposed for the low rank matrix approximation, i.e. volume sampling and dimension reduction [DRVW06, DV06], to the j-subspace problem endowed with the sum of distances measure (i.e., the problem that we study in this thesis). In particular, they give a linear time $(1+\epsilon)$-approximation algorithm for this problem in time

$$O(nd(j/\epsilon)^{O(1)} + nd(j)^{O(j^2)} + (\frac{\log n}{\epsilon})^{\tilde{O}(j^{20}/\epsilon^{10})}).$$

Very recently Sarlos [**?**] shows that random linear combination or better to say dimensionality reduction offered by Theorem 31 can speed up the time to extract the j-subspace that minimizes the sum of squared distances. In particular, he gives a $(1+\epsilon)$-approximation to this j-subspace using merely 2 passes, in time $O\left(M\frac{j}{\epsilon} + (n+d)\cdot\frac{j^2}{\epsilon^2}\right)$ and using $O((n+d)\cdot\frac{j^2}{\epsilon^2})$ memory bits. We sketch the proof of his algorithm below:

Let U_k denote the optimal j-subspace for the sum of squared distance measure, i.e,

$$U_k = arg\min_{F:j\text{-subspace}\subset\mathbb{R}^d}\sum_{p\in P} dist^2(p,F).$$

Let A denote a $n\times d$ matrix for which the rows of A correspond to the points of P. The sketching algorithm that Sarlos proposed works as follows. We multiply A by a $s\times d$ random matrix S with i.i.d. zero mean ± 1 entries where $s = \Theta(j/\epsilon)$. This gives the subspace $Span\left\{AS^T\right\}$ of dimension at most s in which there exists a j-subspace that $(1+\epsilon)$-approximates U_k. Recall

that S^T is the transpose matrix of S. Then, we project the rows of matrix A onto the subspace spanned by the rows of AS^T. The multiplication and projection times are together $O(Mj/\epsilon)$. Finally, we compute the best j-subspace inside $\text{Span}\{AS^T\}$ using SVD in time $O((n+d)(j/\epsilon)^2)$.

Overall the time to extract a j-subspace is $O(M(j/\epsilon) + (n+d)(j/\epsilon)^2)$. Note that we need two passes over the matrix A, where in the first pass we multiply A by S and at the end of first pass we apply the gram-schmidt orthogonalization to find an orthonormal basis V of the span of AS^T. During the second pass we project the rows of matrix A onto V and at the end of the second pass we run one of the SVD algorithms (see [GL96]) to compute the best j-subspace inside $\text{Span}\{AS^T\} = \text{Span}\{V\}$.

The space complexity of this algorithm is $O((n+d)\frac{j^2}{\epsilon^2})$ since we need to keep an orthonormal basis of AS^T. Note that since the whole matrix S is of size $O(sd)$ we can store S as well; although S is fully independent here, but the space to do the whole computation is dominating so we can store S. This model is in fact called *semi-streaming model*.

The proof boils down to showing that if we multiply each row p of the matrix A by S, the length of the segment which connects p to the orthogonal projection of Sp on the U_k is $(1+\epsilon)$-approximation of the length of p's orthogonal projection on the U_k. See Figure 4.4.

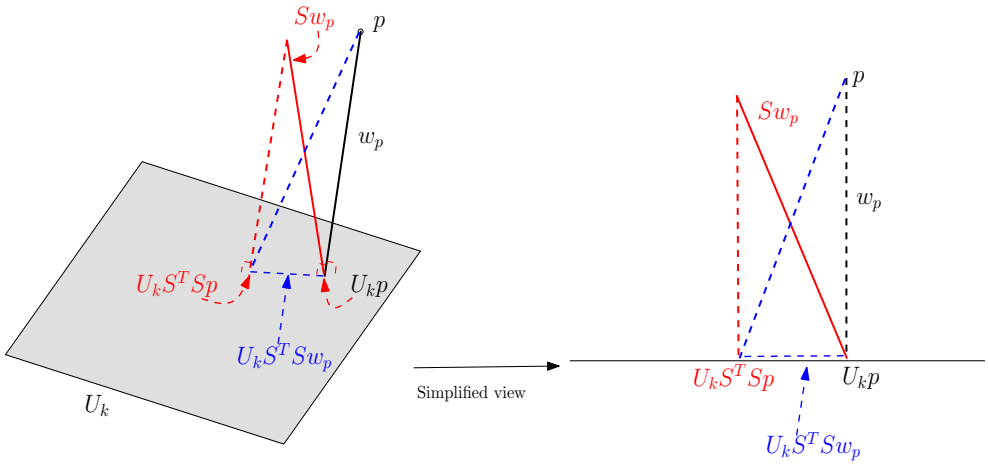


Fig. 4.4: $(1+\epsilon)$-approximation to the best j-subspace using Johnson-Lindenstrauss Lemma.

Let the orthogonal projection of p on the U_k be U_kp. Set $w_p \in \mathbb{R}^{d-j}$ such that $w_p = p - U_kp$. Thus w_p is orthogonal to U_k. Note that $\|w_p\|^2 = \|p - U_kp\|^2$.

The multiplication of S by p changes w_p into Sw_p. The orthogonal projection of Sw_p onto U_kS^T is $U_kS^TSw_p$. Note that the span of U_kS^T is the same as U_k since S^T linearly combines the row of U_k. The interesting part of the proof is now here where we prove that for the $s \times d$ random matrix S if $s = \Theta(j/\epsilon)$, then the Euclidean length of $U_kS^TSw_p$ is ϵ-fraction of the Euclidean length of w_p, i.e, $\|U_kS^TSw_p\| \leq \epsilon \cdot \|w_p\|$. Thus using Pythagorean Theorem we get

$$\|U_kS^TSp\|^2 = \|U_kS^TSw_p\|^2 + \|w_p\|^2 \leq (1+\epsilon)\|w_p\|^2.$$

4.2 Sketching vs. L_p-Sampling

Similar to Sections 3.3 and 3.4 where we discussed how to implement non-uniform samples in the turnstile streaming model, we are also interested to implement random linear combination of Theorem 31 in this model. Note that if we want to use Theorem 31 we need to store the whole random matrix S which needs a space of $O(sn)$ which is larger than the length of the vector a. Recall that if we use S to approximately solve the j-subspace problem as we saw in the previous section we need a space of $O(ns^2)$; so we are happy to keep S as well. However, in general we would like to solve the problem of estimating a sum that we introduced in the beginning of this chapter using a space smaller than n, thus keeping S is not what we are looking for. The obstacle that we have with the implementation of this theorem in data stream is that the entries of S are fully independent and thus we can not keep few of the entries of S and retrieve the rest from those that we have, whenever that we need.

Interesting enough Alon, Matias, and Szegedy [AMS96] show that in order to approximate Z in Theorem 31 within factor $(1+\epsilon)$ using random linear combination we merely need that the entries of S to be 4-wise independent and as we saw in Lemma 6 to implement n random variables which are 4-wise independent we only need an irreducible polynomial of degree $O(\log n)$ which can be stored in a space of $O(\log n)$. Here we give this prove which is known as *sketching* or *AMS-sketch*. Figure 4.6 depicts the hierarchy of problems following up the AMS-sketch. We explain the problems in this hierarchy later.

Theorem 41 *Let a denote an arbitrary vector in $\mathbb{R}^n$. Let $Z_i = |a[i]|^2$ for $i \in [n]$ and $Z = Z_1 + \cdots + Z_n$. Let $s_1 = 16\epsilon^{-2}$ and $s_2 = 2\log(1/\delta)$. Let $T : \mathbb{R}^n \to \mathbb{R}^{s_1}$ denote a random linear mapping defined by*

$$T(a)_i = \sum_{j=1}^{n} s[i,j] \cdot a[j], i = 1, \cdots, s_1,$$

where the $s[i,j]$ are 4-wise independent random variables with $\mathbf{E}[s[i,j]] = 0$, $\mathrm{Var}[s[i,j]] = \mathbf{E}[(s[i,j])^2] = 1$. Let $X_1 = |T(a)_1|^2, \cdots, X_{s_1} = |T(a)_{s_1}|^2$ and $X = \frac{X_1+\cdots+X_{s_1}}{s_1}$. Then we have

$$\mathbf{Pr}[(1-\epsilon)Z \leq X \leq (1+\epsilon)Z] \geq 7/8.$$

Further, if we repeat this random process for s_2 times in parallel and take the median of these trials, this median deviates from Z by more than ϵZ with probability at most δ.

Note that here we drop the condition of being a uniform subgaussian tail for the entries of S.

Proof. Similar to the second proof of Lemma 24 we compute the expectation and variance of X_i and then use Chebyshev's Inequality (see also Lemma 2) to bound the deviation of the random variable X.

Since the random variables $s[i,j]$ for $j \in [n]$ are pairwise independent we get

$$\begin{aligned}\mathbf{E}[X_i] &= \mathbf{E}[|T(a)_i|^2] \\ &= \mathbf{E}[(\sum_{j=1}^{n} s[i,j] \cdot a[j])^2] \\ &= \sum_{j=1}^{n} (a[j])^2 \mathbf{E}[(s[i,j])^2] + 2 \sum_{1 \le j < k \le n} a[j]a[k]\mathbf{E}[s[i,j]]\mathbf{E}[s[i,k]] \\ &= \sum_{j=1}^{n} (a[j])^2 = Z.\end{aligned}$$

Similarly since the random variables $s[i,j]$ for $j \in [n]$ are 4-independent we get

$$\begin{aligned}\mathbf{E}[(X_i)^2] &= \mathbf{E}[|T(a)_i|^4] \\ &= \sum_{j=1}^{n} (a[j])^4 + 6 \sum_{1 \le j < k \le n} (a[j])^2 (a[k])^2.\end{aligned}$$

It follows that

$$Var[X_i] = \mathbf{E}[(X_i)^2] - (\mathbf{E}[X_i])^2 = 4 \sum_{1 \le j < k \le n} (a[j])^2 (a[k])^2 \le 2Z^2.$$

Therefore by Chebyshev's Inequality we get that

$$\mathbf{Pr}[|X - Z| \ge \epsilon \cdot Z] \le \frac{Var[X]}{(\epsilon \cdot Z)^2} \le \frac{2Z^2}{s_1 \epsilon^2 \cdot Z^2} = \frac{1}{8}.$$

□

4.2.1 Following up results by AMS-Sketch, Figure 4.6

Approximating the p-th Frequency Moment of a Stream

Alon, Matias, and Szegedy [AMS96] by Theorem 41 give an $(1 \pm \epsilon)$-estimator for $F_p(a)$ for $p = 2$ using a optimal space of $O(\epsilon^{-2} \log n \log(1/\delta))$ memory bits. Later, Indyk [Ind06] shows that using pseudorandom generators (PRGs) and stable distributions we can find an AMS-sketch for $p \in [0,2]$. Very recently, Kane, Nelson and Woodruff [KNW10] prove that in order to find an AMS-sketch for $p \in [0,2]$ we do not need to use PRGs and using $O(\log n)$-wise hash functions would be enough to produce the random matrix S.

For the case $p > 2$, it has been shown in [BYJKS02, CKS03] that any algorithm that can estimate $F_p(a)$ within factor $(1 \pm \epsilon)$ in the read only or turnstile streaming model needs a space lower bound of $\Omega(n^{1-2/p})$ bits. For $p > 2$, Indyk and Woodruf [IW05] give an $(1 \pm \epsilon)$-estimator $F'_p(a)$ using an optimal space of $O(\frac{p \log(1/\delta)}{\epsilon^2} n^{1-1/p}(\log n + \log m))$ memory bits. However, they need pseudorandom generators (PRGs) to reduce the number of random bits of their estimators. In Section 9.1 we give an optimal space estimator which does not need PRGs, potentially making it more practical.

Heavy Hitter and Block Heavy Hitter

Indyk and Woodruf algorithm [IW05] and our algorithm use known heavy hitter algorithms. All known heavy hitters algorithms are in fact variants of AMS-sketch. For $p = 1$ heavy hitter problem is solved by the CountMin data structure of Cormode and Muthukrishnan [CM05b], and for $p = 2$ by the CountSketch data structure by Charikar, Chen, and Farach-Colton [CCFC02]. In Section 9.4 we give a heavy hitter algorithm for every $p \in [0, 2]$.

A generalization of heavy hitter problem is called block heavy hitter problem which is introduced in Section 21. In [ABI08], Andoni, DoBa, and Indyk devise a 1-pass algorithm for this problem using $\mathrm{poly}(\phi^{-1} \log n)$ bits of space. In Section 9.4 we give a block heavy hitter algorithm for every $p \in [0, 2]$.

Estimating Cascaded Norms

In [JW09], Jayram and Woodruff give 1-pass algorithms for estimating cascaded moments of an $n \times d$ matrix A (see Definition 22). Namely, they show that for $k \geq 1$ and $p \geq 2$, the 1-pass space complexity of outputting a $(1 \pm \epsilon)$-approximation to $F_k(F_p)(A)$ is $n^{1-2/(kp)} d^{1-2/p} \mathrm{poly}(\epsilon^{-1} \log(nd))$.

They leave open the question of estimating $F_k(F_p)(A)$ for $k \geq 1$ and $p < 2$, though they prove an $\Omega(n^{1-1/k})$ space lower bound for this problem. The only other work in this regime is due to Cormode and Muthukrishnan [CM05b] who prove an $n^{1/2} \mathrm{poly}(\epsilon^{-1} \log(nd))$ upper bound for estimating $F_2(F_0)(A)$ assuming that all stream updates are positive.

In Section 9.3 we give a near-optimal $n^{1-1/k} \mathrm{poly}(\epsilon^{-1} \log(nd))$-space 1-pass upper bound for any $k \geq 1$ and $p \in [0, 2]$, which, together with the results of [JW09], closes the problem for $k \geq 1$ and any $p \geq 0$, up to small factors. As our algorithm works in the general turnstile model, this also improves the algorithm of [CM05b] for estimating $F_2(F_0)$. The space complexity for all regimes of k, p is depicted in Figure 4.5.

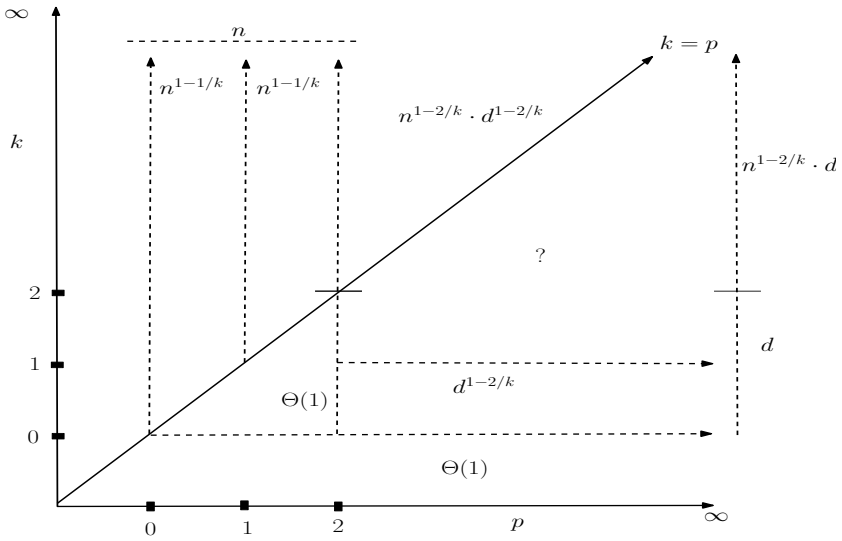


Fig. 4.5: Cascaded norms.

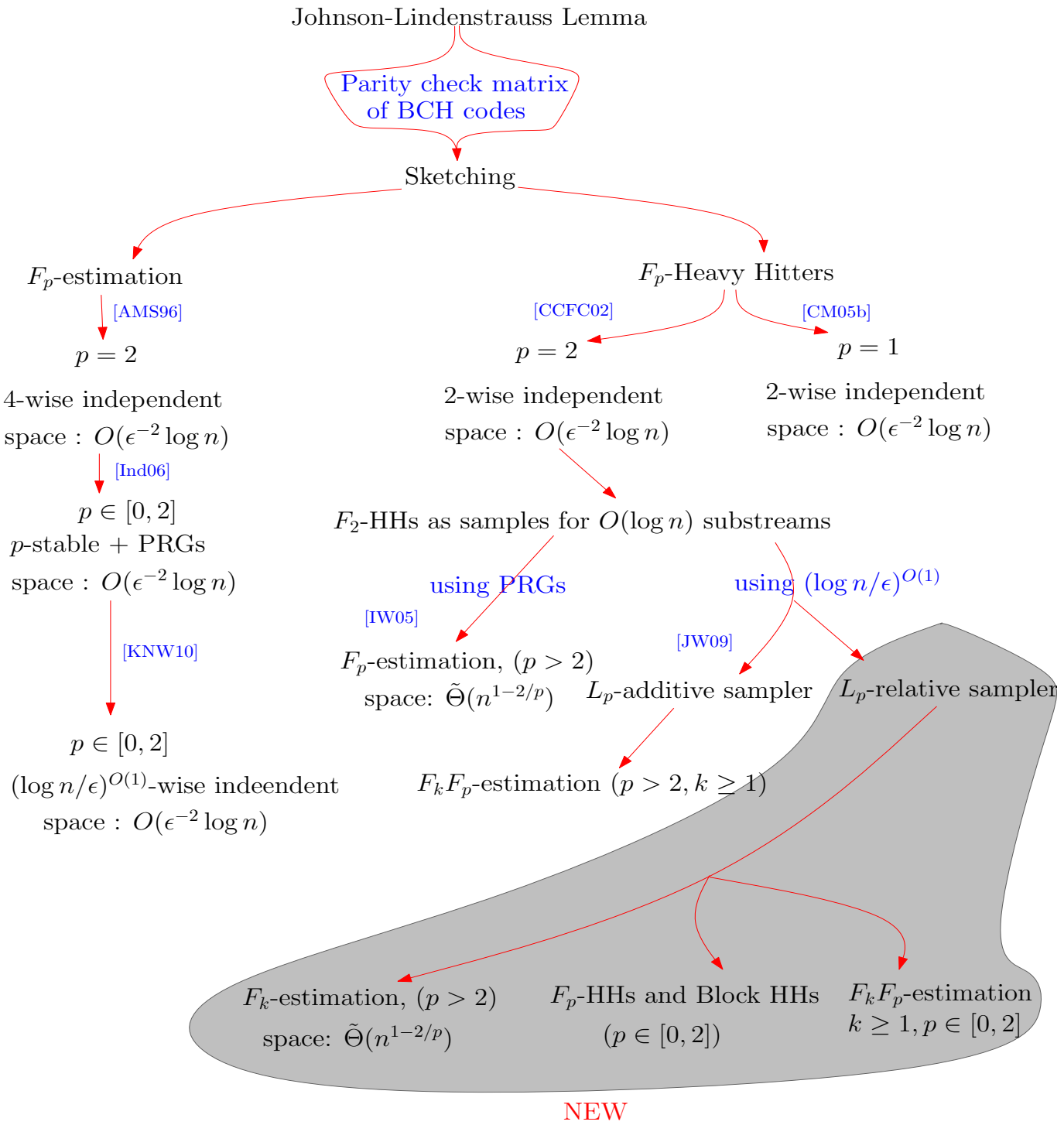


Fig. 4.6: Sketching Hierarchy due to [JW09].

4.3 Coresets

The concept of coreset for the first time was formalized for the L_∞-shape fitting problems by Agarwal, Har-peled, and Varadarajan [AHPV04]. Assume in order to encode the fitting information for a given shape γ (for example, a j-subspace) for the points of $P \subset \mathbb{R}^d$, we create a point $\text{dist}^p(P,\gamma) \subset \mathbb{R}^n$, where the ith coordinate is the p-th power of the distance of the i-th point of P from γ.

A L_∞-shape fitting problem is to find the shape γ that realizes $\min_{\gamma\in\mathcal{F}} \|d^p(P,\gamma)\|_\infty$, where $\mathcal{F}$ is the set of all shapes similar to γ in $\mathbb{R}^d$. Agarwal, Har-peled, and Varadarajan formalize the coreset for this problem as follows:

Definition 42 (Strong L_∞-shape fitting-coreset) *Let P be a (possibly) weighted set in $\mathbb{R}^d$. A set S, $S \subseteq P$, is called a strong L_∞-shape fitting-coreset, if*

$$|\|\text{dist}^p(P,\gamma)\|_\infty - \|\text{dist}^p(S,\gamma)\|_\infty| \leq \epsilon \cdot \|\text{dist}^p(P,\gamma)\|_\infty,$$

for any shape $\gamma \in \mathcal{F}$.

Later Har-Peled and Mazumdar [HPM04] extend the coreset definition to the k-median and k-means clustering problems and they use coresets to obtain fast algorithms for these problems. Here we give a comprehensive survey of following up coresets for k-means and j-subspace clustering problems. See also Figures 4.8 and 4.9.

4.3.1 Har-Peled and Mazumdar [HPM04]

The first step of this construction is to invoke the algorithm [MP04] which returns a set C of k centers such that $\text{cost}(P,C) \leq \beta \cdot \mathcal{OPT}(P,k)$ for some constant β. Recall that $\mathcal{OPT}(P,k)$ is the optimal k-means cost of P. Let C_i be the points of P having c_i as their nearest center in C. Let $R = \sqrt{\text{cost}(P,C)/(\beta\cdot n)}$ be a lower bound estimate of the average mean radius $\sqrt{\mathcal{OPT}(P,k)/n}$.

Next, we construct an appropriate exponential grid around each c_i and snap the points of P to those grids. Let $Q_{i,j}$ be an axis-parallel square with side length $R2^j$ centered at c_i, for $i \in [k], j \in [2\log(\beta n)]$. Let $V_{i,0} = Q_{i,0}$ and $V_{i,j} = Q_{i,j} - Q_{i,j-1}$. We then partition $V_{i,j}$ into a grid with side length $r_j = \epsilon R 2^j/(\beta d)$ and let G_i denote the resulting exponential grid for $V_{i,0} \cdots, V_{i,[2\log(\beta n)]}$.

Next, compute for every point of C_i, the grid cell that contains it. For every non-empty grid cell, pick an arbitrary point of C_i inside it as a representative point for the coreset and assign it a weight equal to the number of points of C_i in this grid cell. Let $S_{i,j}$ denote the resulting weighted set for $V_{i,j}$. Let $S = \cup_{i=1}^{k} \cup_{j=0}^{2\log(\beta n)} S_{i,j}$ be our coreset.

4.3.2 Har-Peled and Kushal [HPK05]

The first step of this construction is again to invoke the algorithm [MP04] which returns a set C of k centers such that $\mathrm{cost}(P, C) \leq \beta \cdot \mathcal{OPT}(P, k)$ for some constant β. Let C_i be the points of P having c_i as their nearest center in C. Around each of the points c_i we place a fan L_i of lines passing through it, see Figure 4.7. This is done by taking a unit sphere centered at c_i and placing an $\epsilon/(3\beta)$-net (see [AS00]) N_i on this sphere. For every $p \in N_i$ we generate the line spanning the segment $c_i p$. Note that $|N_i| = O(\epsilon^{-d})$. For each point $p \in C_i$ let $\ell_i(p)$ be its closest line in L_i and let p' be the projection of p into $\ell_i(p)$.

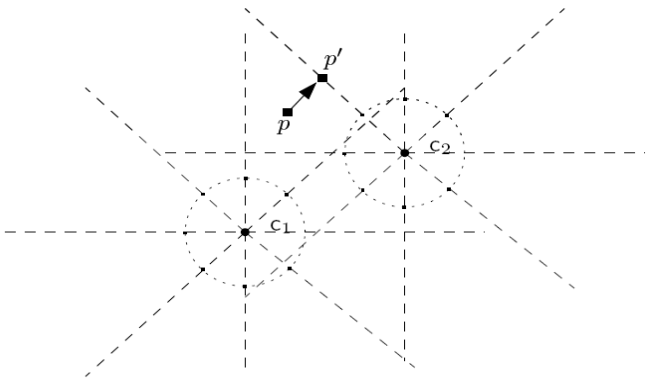


Fig. 4.7: Fans around the centers in C.

Fix one $\ell_{i,j} \in L_i$ for $i \in [k], j \in [O(\epsilon^{-d})]$. Let $P'_{i,j}$ be the projected points on $\ell_{i,j}$. We scan the points on the line $\ell_{i,j}$ from left to right and group them into batches with cumulative error equal to $\Phi_{i,j} = \frac{\epsilon^2 \cdot \mathrm{cost}(P,C)}{c\beta \cdot k^2}$ for some constant c, where the cumulative error is $\sum_{p \in P'_{i,j}} w_p \cdot \mathrm{dist}(p, \mu_P(P'_{i,j}))$. Let $\mathcal{B} = \{B_1, \cdots, B_u\}$ denote the resulting batches. It is straightforward to verify that $u = |\mathcal{B}| = O(k^2 \epsilon^{-2})$.

Fix a batch B_v for $v \in [u]$ and set $\bar{m} = B_v^r$. Let the leftmost point of B_v be p_ℓ and the rightmost point be p_r. Next we set two points B_v^ℓ and B_v^r. The point B_v^ℓ is the leftmost extreme of B_v and the point B_v^r is the rightmost extreme of B_v. For each point $p \in B_v$ which is to the right of $\mu_P(B_v)$ we assign a weight of $\frac{\mathrm{dist}(p,\bar{m})}{\mathrm{dist}(p_r,\bar{m})}$ to B_v^r. Similarly, for each point $p \in B_v$ which is to the left of $\mu_P(B_v)$ we assign a weight of $\frac{\mathrm{dist}(p,\bar{m})}{\mathrm{dist}(p_\ell,\bar{m})}$ to B_v^ℓ. At the end we scale up the weights so that $w_{B_v^\ell} + w_{B_v^r} = |\mathcal{B}|$.

4.3.3 Frahling and Sohler [FS05]

Frahling and Sohler consider construction of coresets in the discrete space $[\Delta]^d = \{1, \cdots, \Delta\}^d$ instead of $\mathbb{R}^d$. This allows them to maintain their coreset in the dynamic geometric streaming model which consists of a sequence of $m = \max(n, \Delta)^{O(1)}$ insertion and deletion operations of points from the space $[\Delta]^d$. Here we assume that n and Δ are polynomially dependent to each other.

The offline construction of their coreset for the k-means problem works as follows:

We impose $\log \Delta$ nested square grids $G_1, \cdots, G_{\log \Delta}$ over the point space. The side length of the grid cells in grid G_i is 2^i. Our goal will be to identify for each grid G_i its heavy cells. A cell in grid G_i is called heavy if it contains at least $\delta \frac{\mathcal{OPT}(P,k)}{2^i}$ points of P for $\delta < 1$ that turns out to be the reverse of the coreset size, see 4.8. As for $\mathcal{OPT}(P, k)$ we can replace it with a constant factor approximation cost or its guesses.

We start the construction with the coarsest grid $G_{\log \Delta}$. First, we identify every heavy cell in $G_{\log \Delta}$. Then we proceed similar to the process of building a quadtree. We subdivide every heavy cell c into 2^d equal size quadratic subcells. These subcells are in $G_{\log \Delta - 1}$. If none of these subcells is heavy we mark c. Otherwise we recurse this process with all heavy subcells. the recursion eventually stops because at some point a heavy cell is required to have more than n points inside.

At the end of this recursive process we choose the center p_c of every marked cell c to be its representative point and assign a weight to p_c which is equal to the number of points inside the cell c. The set of centers of all marked would be our coreset.

4.3.4 Chen [Che06]

The first step of this construction is again to invoke the algorithm [MP04] which returns a set C of k centers such that $\text{cost}(P, C) \leq \beta \cdot \mathcal{OPT}(P, k)$ for some constant β. Recall that $\mathcal{OPT}(P, k)$ is the optimal k-means cost of P. Let C_i be the points of P having c_i as their nearest center in C. Let $R = \sqrt{\text{cost}(P, C)/(\beta \cdot n)}$ be a lower bound estimate of the average mean radius $\sqrt{\mathcal{OPT}(P, k)/n}$.

Next, we construct an $O(\log n)$ balls of exponentially growing radii around each c_i and sample uniformly at random the points of P inside each ring. Let $\text{ball}(c_i, R2^j)$ be the ball centered at c_i of radius $R2^j$ for $i \in [k], j \in [2 \log(\beta n)]$. Let $V_{i,0} = \text{ball}(c_i, R)$ and $V_{i,j} = \text{ball}(c_i, R2^j) - \text{ball}(c_i, R2^{j-1})$ be the jth ring set for the ith center. Let $s = O(ck^2 d\beta^2 \epsilon^{-2} \log(nd/(\epsilon\delta)) \cdot \log \log n)$ for small enough number δ and large enough constant c. We then pick s samples from $V_{i,j}$ independently and uniformly (with replacement) and assign each point weight $|V_{i,j}|/s$. Let $S_{i,j}$ be the resulting weighted sample set. Let $S = \cup_{i=1}^{k} \cup_{j=0}^{2\log(\beta n)} S_{i,j}$ be our coreset.

4.3.5 Feldman, Fiat, and Sharir [FFS06]

In the continuation of the coresets development in low-dimensional spaces, Feldman, Fiat, and Sharir develop a coreset for the j-subspace problems using sum of distances and sum of squared distances as measures. They realized that given a point set P on a line F we can reduce the problem of finding a coreset for the j-subspace problem to the problem of finding a coreset for a center which is a point somewhere in $\mathbb{R}^d$, but now this center has a weight which is some positive real number.

As an example, see Figure 4.9 for the 1-subspace problem. Let F be a line where the points of P are lying on it and let L be an arbitrary line. Let us assume that L and F are in a plane of dimension 2 and they intersect at some point O. Our goal here is to find a coreset Q such that

$$(1-\epsilon)\cdot \text{cost}(P,L) \leq \text{cost}(Q,L) \leq (1+\epsilon)\cdot \text{cost}(P,L).$$

Let p be a point of P. In this case we can have this fact that $\text{dist}(p,L) = \text{dist}(p,O)\cdot \sin(\alpha_L)$, where α_L is the angle between L and F. This simple observation says for this special case we can replace line L by point O, find a coreset for this point by using known coresets for this problem and then multiply the cost of the coreset by $\sin(\alpha_L)$.

In general and when L and F are in $\mathbb{R}^d$ using Trigonometry, we can prove that there always exists a point $O \in \mathbb{R}^d$ such that $\text{dist}(p,L) = w_L \cdot \text{dist}(p,O)$, where w_L is a positive real number which depends on the line L and its position with respect to L.

See [FFS06] for the general setting when $j \geq 1$ and points of P are in $\mathbb{R}^d$.

Ref.	Problems	Size	Streaming Size	Streaming Model	Dimension
[HPM04]	k-means, k-median	$O(k\epsilon^{-d}\log n)$	$O(k\epsilon^{-d}\log^{2d+2} n)$	Read-only	Low
[HPK05]	k-means, k-median	$O(k^3\epsilon^{-d-1})$	$O(k\epsilon^{-d}\log^{2d+2} n)$	Read-only	Low
[FS05]	k-means, k-median MaxWM, MaxTSP MaxST, MaxCut	$O(k\epsilon^{-d}\log n)$	$O(k^2\epsilon^{-2d+6}\log^7 n)$	Insertion and Deletion model	Low
[Che06]	k-means, k-median	$\tilde{O}(dk^2\epsilon^{-2}\log n)$	$O(d^2k^2\epsilon^{-2}\log^8 n)$	Read-only	High
[FFS06]	j-subspace	$O((\frac{j\cdot\log n}{\epsilon^2})^{j^2})$	didn't compute	———	Low
Here	k-means (weak coreset)	$O(\frac{k}{\epsilon^5}))$	$O(k^2\frac{\log^{10} n}{\epsilon^5})$	Read-only	High
Here	k-means (strong coreset)	$O(\frac{kd\log\log n}{\epsilon^5}))$	$O(dk^2\frac{\log^{10} n}{\epsilon^5})$	Read-only	High

Fig. 4.8: The complexity measures of the known k-means and k-median coresets, where $\tilde{O}(\cdot)$ notation hides $\log\log n$ and $\log(1/(\epsilon\delta))$ terms. The coreset size is given only for the k-median problem.

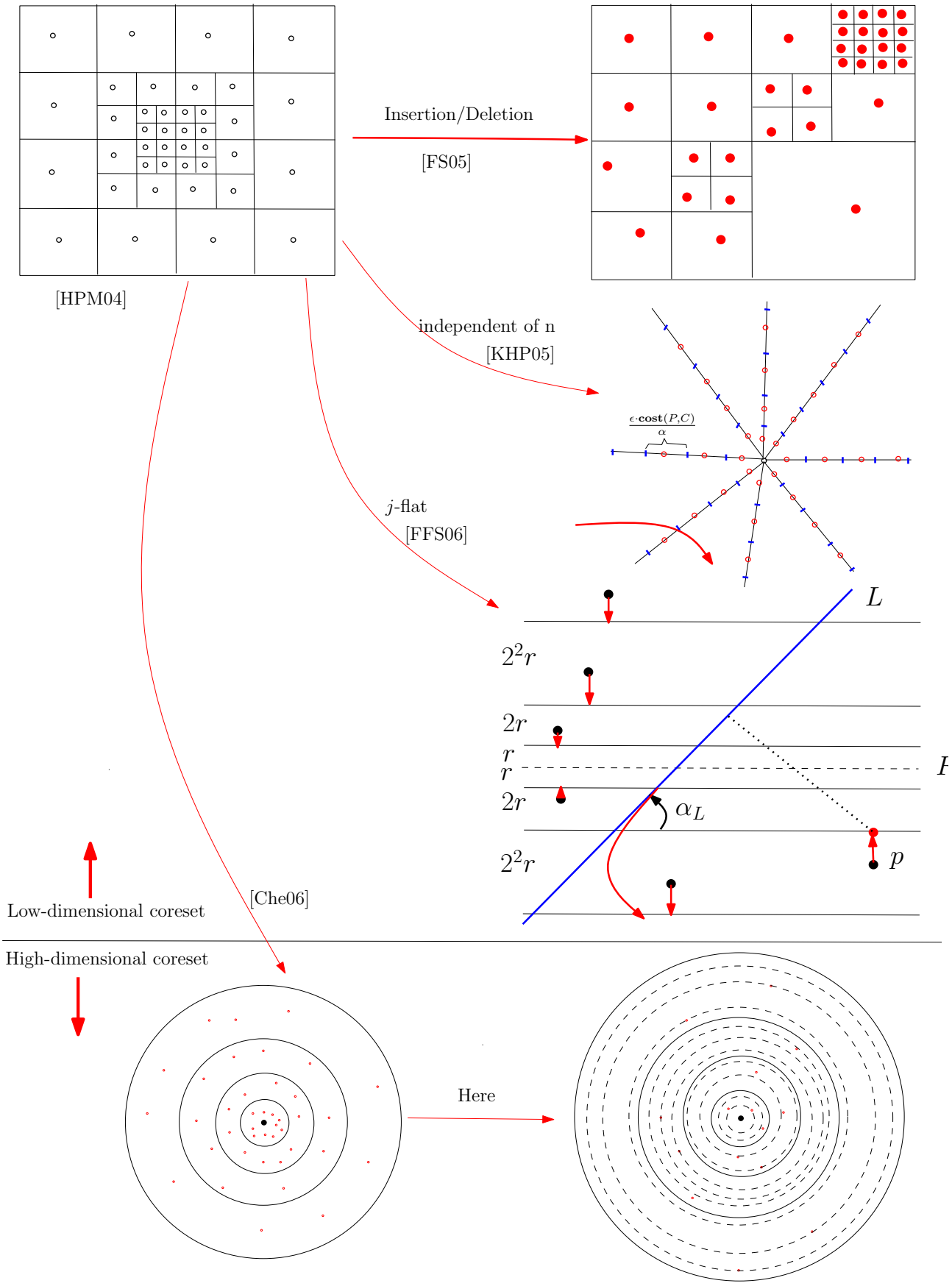


Fig. 4.9: An evolutionary process of known k-means and k-median coresets.

Chapter 5

k-Means Clustering

In Section 3.1 we gave an overview of the application of non-uniform sampling in the k-means problem. In this chapter we explain the idea in details. First in Section 5.1 we give a weak (k, ϵ)-coreset. We then use the weak coreset to find a linear time $(1+\epsilon)$-approximation algorithm in Section 5.2 and to give a streaming algorithm in Section 5.3.

5.1 Weak (k, ϵ)-coreset Construction

The first step of our algorithm is similar to the coreset constructions in [HPM04] and [Che06]. We run the constant factor approximation algorithm for k-means clustering due to [MP04] that in $O(nkd)$ time returns k centers $C = \{c_1, \cdots, c_k\}$. Let β denote the approximation factor of this algorithm and let C_i denote the set of points from P that are nearer to c_i than to any other center in C (ties can be broken arbitrarily).

We partition each C_i into two sets C_i^{in} and C_i^{out}. The set C_i^{in} contains all points that are close to c_i, i.e. all points contained in a closed ball $B(c_i, r_i) = \{p \in \mathbb{R}^d \mid \text{dist}(p, c_i) \leq r_i\}$ with center c_i and radius $r_i = \sqrt{\frac{\text{cost}(C_i, c_i)}{\epsilon \cdot |C_i|}}$). Thus we have $C_i^{in} = C_i \bigcap B(c_i, r_i)$. The set C_i^{out} contains the remaining points of C_i, i.e. $C_i^{out} = C_i \setminus B(c_i, r_i)$.

To construct the coreset we proceed differently for the points in C_i^{in} and C_i^{out}. From the sets C_i^{in} we draw a set of s_i^{in} points independently and uniformly at random. Then we assign to each point the weight $|C_i^{in}|/s_i^{in}$. Let S_i^{in} denote the resulting weighted sample. From each set C_i^{out} we draw a sample set $S_i^{out} = \{s_1, \cdots, s_{s_i^{out}}\}$ according to a non-uniform probability distribution. The weights of the points will also be distributed non-uniformly.

We proceed for each cluster separately. The probability of choosing point $q \in C_i^{out}$ is $p_q = \mathbf{Pr}[q \in S_i^{out}] = \frac{\text{dist}^2(q, c_i)}{\text{cost}(C_i^{out}, c_i)}$. Each sample point q is assigned a weight $w_q = \frac{\text{cost}(C_i^{out}, c_i)}{s_i^{out} \text{dist}^2(q, c_i)}$, i.e. the weight of a point depends on its distance to the center c_i. The further a point is away from the center, the smaller its weight.

Finally, we set $\mathcal{S} = \bigcup_{i=1}^{k} \left(S_i^{in} \cup S_i^{out}\right)$ and $\mathcal{T}$ will be the set of all centroids of combinations of $2/\epsilon$ points from $\mathcal{S}$ (we allow repetition of points). We will show that for large enough sample sizes, our construction indeed gives a weak coreset. We remark that the set $\mathcal{T}$ depends on the choice of the set $\mathcal{S}$.

Coreset(P)

(1) $C = \{c_1, \cdots, c_k\} =$ The k centers returned by the β-approximation algorithm [MP04].

(2) For $i = 1$ to k do

(a) Let $r_i = \sqrt{\frac{\text{cost}(C_i, c_i)}{\epsilon \cdot |C_i|}}$).

(b) $C_i^{in} = C_i \bigcap B(c_i, r_i)$ and $C_i^{out} = C_i \setminus B(c_i, r_i)$.

(c) Let S_i^{in} denote a sample set taken u.a.r. from C_i^{in}, where each sample has the weight of $|C_i^{in}|/s_i^{in}$.

(d) Pick s_i^{out} points $s_1, \cdots, s_{s_i^{out}}$ i.i.d. from C_i^{out} such that

$$p_q = \mathbf{Pr}[q \in S_i^{out}] = \frac{\text{dist}^2(q, c_i)}{\text{cost}(C_i^{out}, c_i)}.$$

(e) For $i = 1$ to s_i^{out} do: $\quad w(s_i) = \dfrac{1}{s_i^{out} \cdot p_{s_i}}$.

(3) $\mathcal{S} = \bigcup_{i=1}^{k} \left(S_i^{in} \cup S_i^{out}\right)$.

(4) $\mathcal{T} =$ The set of all centroids of combinations of $2/\epsilon$ points from $\mathcal{S}$ (we allow repetition of points).

(5) return $\mathcal{S}$ and $\mathcal{T}$.

Analysis We first show that $\mathcal{T}$ is a $(k, 6\epsilon)$-approximate centroid set, if the cost of any subset $K \subseteq \mathcal{T}$, $|K| = k$, and the cost of an optimal solution $\mathcal{OPT}$ are approximated within a factor of $(1 \pm \epsilon)$. Then we show that for an arbitrary set K of k centers

$$|\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| \leq \epsilon \cdot \text{cost}(P, K)$$

with probability $1 - \lambda$ for large enough s_i^{in} and s_i^{out}. This implies that with this probability $\mathcal{T}$ is a $(k, 6\epsilon)$-approximate centroid set. Finally, we show that for any subset of $\mathcal{T}$ of size k we get

$$|\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| \leq \epsilon \cdot \text{cost}(P, K)$$

Here, the difficulty is that the set $\mathcal{T}$ depends on the random process and hence there are dependencies (this is, why we cannot immediately apply the previous result).

$\mathcal{T}$ is a (k, ϵ)-approximate centroid set

Lemma 43 *Let $P \subseteq \mathbb{R}^d$ be a point set and let $0 < \epsilon < 1/2$ and $k \geq 1$. Let $\mathcal{S} \subseteq \mathbb{R}^d$ be a weighted point set, and let $\mathcal{T}$ be the set of all centroids of combinations (with repetition) of $2/\epsilon$ points from $\mathcal{S}$. If we have*

$$|\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| \leq \epsilon \cdot \text{cost}(P, K)$$

for every set $K \subseteq \mathcal{T}$ of k points and if

$$|\text{cost}(\mathcal{S}, \mathcal{OPT}) - \text{cost}(P, \mathcal{OPT})| \leq \epsilon \cdot \text{cost}(P, \mathcal{OPT})$$

for an optimal set $\mathcal{OPT} \subseteq \mathbb{R}^d$ of k centers, then $\mathcal{T}$ is a $(k, 6\epsilon)$-approximate centroid set.

Proof. Let $\mathcal{OPT}$ denote an optimal set of cluster centers and let $O_1, \ldots, O_k$ be the induced clustering. By Corollary 11 we know that for every cluster O_i the set $\mathcal{T}$ contains a $(1+\epsilon)$-approximation. Since the cost of $\mathcal{OPT}$ is approximated within a factor of $(1 \pm \epsilon)$ and we lose at most another factor of $(1+\epsilon)$ when we move each center to the nearest point from $\mathcal{T}$, we have a set K^* of k centers in $\mathcal{T}$ with

$$\text{cost}(\mathcal{S}, K^*) \leq (1+\epsilon)^2 \cdot \text{cost}(P, O).$$

Since we also know that

$$\text{cost}(\mathcal{S}, K^*) \geq (1-\epsilon)\text{cost}(P, K^*),$$

we know that

$$\text{cost}(P, K^*) \leq (1+\epsilon)^2/(1-\epsilon) \cdot \text{cost}(P, \mathcal{OPT}).$$

For $\epsilon \leq 1/2$ we can obtain that $\mathcal{T}$ is a $(k, 6\epsilon)$-approximate centroid set. □

Arbitrary centers are approximated within a factor $(1 \pm \epsilon)$ The first step of our analysis will be to show that for an arbitrary fixed set $K = \{k_1, \cdots, k_l, \cdots, k_k\}$ of k centers, the cost of $\mathcal{S}$ is a $(1 \pm \epsilon)$-approximation of the cost of P. We will prove the following lemma.

Lemma 44 *Given a point set P in $\mathbb{R}^d$ and a set $K \subseteq \mathbb{R}^d$ of k centers. Let $\epsilon, \lambda > 0$ be parameters. Then there is a constant c such that for $s_i^{in}, s_i^{out} \geq c \cdot \frac{\ln(k/\lambda)}{\epsilon^4}$, $1 \leq i \leq k$, the sample set $\mathcal{S} = \bigcup_{i=1}^{k} \left(S_i^{in} \cup S_i^{out}\right)$ computed by our algorithm satisfies $|\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| \leq \epsilon \cdot \text{cost}(P, K)$ with probability $\geq 1 - \lambda$.*

Proof. We will assume $\epsilon \leq 1/2$.. Let $S_i = S_i^{in} \cup S_i^{out}$and let k_l denote the nearest center from K to $B(c_i, r_i)$. The analysis will distinguish between the cases

(1) $\text{dist}(k_l, c_i) \geq r_i + \frac{r_i}{\epsilon} = \frac{r_i(1+\epsilon)}{\epsilon}$, and

(2) $\text{dist}(k_l, c_i) < r_i + \frac{r_i}{\epsilon} = \frac{r_i(1+\epsilon)}{\epsilon}$.

Case (1) Every point $p \in B(c_i, r_i)$ has distance at least $\mathrm{dist}(B(c_i, r_i), k_l)$ to the nearest center from K. Since we are in case (1), it has distance at most $\mathrm{dist}(B(c_i, r_i), k_l) + 2r_i \leq (1+2\epsilon) \cdot \mathrm{dist}(B(c_i, r_i), k_l)$ to the nearest center from K. By our construction we have that the sum of the weights of the points in S_i^{in} is exactly $|C_i^{in}|$. Hence, we get

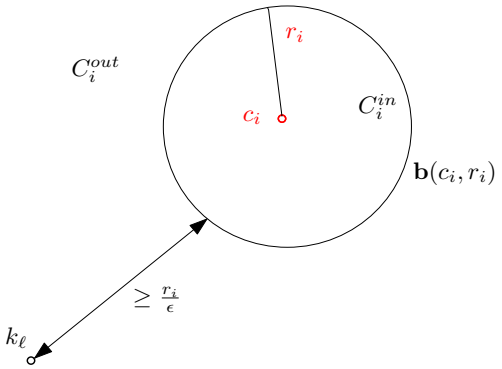


Fig. 5.1: Case (1)

$$\begin{aligned} \left|\mathrm{cost}(C_i^{in}, K) - \mathrm{cost}(S_i^{in}, K)\right| &\leq |C_i^{in}| \left(\left((1+2\epsilon) \cdot \mathrm{dist}(B(c_i, r_i), k_l)\right)^2 - (\mathrm{dist}((c_i, r_i), k_l))^2 \right) \\ &\leq 8 \cdot \epsilon \cdot |C_i^{in}| \cdot \mathrm{dist}^2((c_i, r_i), k_l) \\ &\leq 8 \cdot \epsilon \cdot \mathrm{cost}(C_i^{in}, K) \ . \end{aligned}$$

Next we consider the points from C_i^{out}. Let $W = \sum_{p \in S_i^{out}} w_p$ be the random variable for the sum of weights of points in S_i^{out}. In case (1) we will approximate the error for the outer points by the sum of their contributions. We have

$$\begin{aligned} |\mathrm{cost}(C_i^{out}, K) - \mathrm{cost}(S_i^{out}, K)| &\leq \mathrm{cost}(C_i^{out}, K) + \mathrm{cost}(S_i^{out}, K) \\ &\leq \sum_{q \in C_i^{out}} \mathrm{dist}^2(q, k_l) + \sum_{p \in S_i^{out}} w_p \cdot \mathrm{dist}^2(p, k_l) \ . \end{aligned}$$

Now we use the doubled triangle inequality, i.e. $\mathrm{dist}^2(p, r) \leq 2(\mathrm{dist}^2(p, q) + \mathrm{dist}^2(q, r))$ for all $p, q, r \in \mathbb{R}^d$, to obtain

$$\begin{aligned} & \sum_{q \in C_i^{out}} \mathrm{dist}^2(q, k_l) + \sum_{p \in S_i^{out}} w_p \cdot \mathrm{dist}^2(p, k_l) \\ \leq & \sum_{q \in C_i^{out}} 2 \left(\mathrm{dist}^2(q, c_i) + \mathrm{dist}^2(c_i, k_l) \right) \\ + & \sum_{p \in S_i^{out}} 2 w_p \left(\mathrm{dist}^2(p, c_i) + \mathrm{dist}^2(c_i, k_l) \right) \\ \leq & \sum_{q \in C_i^{out}} 2 \big(\mathrm{dist}^2(q, c_i) + 2(r_i^2 + \mathrm{dist}^2(B(c_i, r_i), k_l)) \big) \\ + & \sum_{p \in S_i^{out}} 2 w_p \big(\mathrm{dist}^2(p, c_i) + 2(r_i^2 + \mathrm{dist}^2(B(c_i, r_i), k_l)) \big) \end{aligned}$$

$$
\begin{aligned}
&\leq \Big(6\mathrm{cost}(C_i, c_i) + 4|C_i^{\mathrm{out}}| \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\Big) \\
&+ \Big(\sum_{p \in S_i^{\mathrm{out}}} 6w_p \mathrm{dist}^2(p, c_i) + \sum_{p \in S_i^{\mathrm{out}}} 4w_p \mathrm{dist}^2(B(c_i, r_i), k_l)\Big) \\
&\leq \big(6r_i^2 \epsilon \cdot |C_i| + 4\epsilon \cdot |C_i| \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\big) \\
&+ \big(6\mathrm{cost}(C_i^{\mathrm{out}}, c_i) + 4\mathrm{dist}^2(B(c_i, r_i), k_l) \sum_{p \in S_i^{\mathrm{out}}} w_p\big) \\
&\leq \big(6r_i^2 \epsilon \cdot |C_i| + 4\epsilon \cdot |C_i| \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\big) \\
&+ \big(6 \cdot \mathrm{cost}(C_i, c_i) + 4W \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\big) \\
&\leq \big(6r_i^2 \epsilon \cdot |C_i| + 4\epsilon \cdot |C_i| \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\big) \\
&+ \big(6r_i^2 \epsilon \cdot |C_i| + 4W \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)\big)
\end{aligned}
$$

Since $\mathrm{dist}((c_i, r_i), k_l) \geq \frac{r_i}{\epsilon}$ implies $r_i^2 \leq \epsilon^2 \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)$ we have for $\epsilon \leq 1/2$:

$$
\begin{aligned}
&\leq (2\epsilon \cdot |C_i| \cdot \mathrm{dist}^2(C_i^{\mathrm{in}}, k_l) \cdot (6\epsilon^2 + 2) + 4W \cdot \mathrm{dist}^2((c_i, r_i), k_l) \\
&\leq 10\epsilon \cdot |C_i^{\mathrm{in}}| \cdot \mathrm{dist}^2(B(c_i, r_i), k_l) + 4W \cdot \mathrm{dist}^2(B(c_i, r_i), k_l)
\end{aligned}
$$

Next we show that W is at most $\epsilon \cdot |C_i^{\mathrm{in}}|$ with high probability.

Claim 45 $\mathbf{Pr}[W = \sum_{p \in S_i^{\mathrm{out}}} w_p \geq 4\epsilon \cdot |C_i^{\mathrm{in}}|] \leq 1 - \lambda/k$

Proof. Define the random variable $Y_j = w_{s_j}$ to be the weight of the jth sample point in S_i^{out}. Hence $W = \sum_{j=1}^{s_i^{\mathrm{out}}} Y_j$. The expected value $\mathbf{E}[Y_j]$ of Y_j is, by definition, $\mathbf{E}[Y_j] = \sum_{q \in C_i^{\mathrm{out}}} p_q w_q = \frac{|C_i^{\mathrm{out}}|}{s_i^{\mathrm{out}}} \leq \frac{\epsilon |C_i|}{s_i^{\mathrm{out}}}$. We also have $|C_i^{\mathrm{in}}| \geq (1 - \epsilon)|C_i|$. Hence, $\mathbf{E}[Y_j] \leq \frac{2\epsilon \cdot |C_i^{\mathrm{in}}|}{s_i^{\mathrm{out}}}$ for $\epsilon \leq 1/2$. Thus, $\mathbf{E}[W] \leq 2\epsilon \cdot |C_i^{\mathrm{in}}|$. An upper bound for the weight of sample point $q \in C_i^{\mathrm{out}}$ is given by

$$
w_q = \frac{\mathrm{cost}(C_i^{\mathrm{out}}, c_i)}{s_i^{\mathrm{out}} \mathrm{dist}^2(q, c_i)} \leq \frac{\mathrm{cost}(C_i^{\mathrm{out}}, c_i)}{s_i^{\mathrm{out}} \frac{\mathrm{cost}(C_i, c_i)}{\epsilon |C_i|}} \leq \frac{\epsilon |C_i|}{s_i^{\mathrm{out}}}.
$$

Since we have an upper bound for the random variables we can once again apply the framework of Example 1 to find s_i^{out}. In Example 1 we used additive Hoeffding inequality to get the size of the sample set; however we can also use multiplicative Hoeffding inequality 5 to get a similar bound for s_i^{out}.

Define $Z_j = \frac{Y_j}{\frac{\epsilon |C_i|}{s_i^{\mathrm{out}}}} \leq 1$. Let $Z = \sum_{j=1}^{s_i^{\mathrm{out}}} Z_j$ then $\mathbf{E}[Z] \leq s_i^{\mathrm{out}}$.

By multiplicative Hoeffding we obtain:

$$\begin{aligned}
& \mathbf{Pr}\Big[|\sum_{j=1}^{s_i^{out}} Y_j - \mathbf{E}[Y]| > \epsilon |C_i|\Big] = \mathbf{Pr}\Big[|Z - \mathbf{E}[Z]| > s_i^{out}\Big] \\
\leq\ & \mathbf{Pr}\Big[|Z - \mathbf{E}[Z]| > \frac{s_i^{out}}{\mathbf{E}[Z]}\mathbf{E}[Z]\Big] \\
\leq\ & 2\exp\left(-\frac{\mathbf{E}[Z]\cdot\min\left\{\left(\frac{s_i^{out}}{\mathbf{E}[Z]}\right),\left(\frac{s_i^{out}}{\mathbf{E}[Z]}\right)^2\right\}}{3}\right) \\
=\ & 2\exp\left(-s_i^{out}/3\right)\ .
\end{aligned}$$

We choose $s_i^{out} \geq 3\ln(2k/\lambda)$. This implies $\mathbf{Pr}[|Y - \mathbf{E}[Y]| > \epsilon|C_i|] \leq \lambda/k$. Hence, we get that $\mathbf{Pr}[W > 4\epsilon \cdot |C_i^{in}|] \leq \lambda/k$. □

It follows that

$$\begin{aligned}
\sum_{q\in C_i^{out}} cost(q,k_l) + \sum_{p\in S_i^{out}} w_p \cdot cost(p,k_l) &\leq 26\epsilon\cdot|C_i^{in}|\cdot dist^2(B(c_i,r_i),k_l) && (5.1)\\
&\leq 26\epsilon cost(C_i^{in},K) && (5.2)
\end{aligned}$$

with probability at least $1-\lambda/k$.

Since

$$|cost(C_i^{out},K) - cost(S_i^{out},K)| \leq cost(C_i^{out},K) + cost(S_i^{out},K)$$

this is an upper bound for the error of the outer points. Overall error for the sample set $S_i^{in}\cup S_i^{out}$ in case (1) would be

$$\begin{aligned}
|cost(S_i^{in}\cup S_i^{out},K) - cost(C_i^{in}\cup C_i^{out},K)| &\leq 8\epsilon cost(C_i^{in},K) + 26\epsilon cost(C_i^{in},K)\\
&\leq 34\epsilon cost(C_i^{in},K).
\end{aligned}$$

Case (2)

Lemma 46 (Haussler [Hau92]) *Let $h(\cdot)$ be a function defined on a set P, such that for all $p\in P$, we have $0\leq h(p)\leq M$, where M is a fixed constant. Let $\mathcal{S}=\{p_1,\ldots,p_s\}$ be a multiset of s samples drawn independently and identically from P, and let $\delta>0$ be a parameter. If $s\geq (M^2/2\delta^2)\cdot\ln(2/\lambda)$, then $\mathbf{Pr}\Big[\left|\frac{h(P)}{|P|}-\frac{h(\mathcal{S})}{|\mathcal{S}|}\right|\geq\delta\Big]\leq\lambda$, where $h(\mathcal{S})=\sum_{s\in\mathcal{S}}h(s)$.*

We want to apply Lemma 46 to analyze the error of the uniform sampling from C_i^{in}. Therefore, let $h(p)=dist^2(p,K)$ for $p\in C_i^{in}$. Since C_i^{in} is contained in a ball of radius r_i we get that

$$\max_{p\in P} h(p) \leq (dist(C_i^{in},K)+2r_i)^2 \leq 2(dist^2(C_i^{in},K)+4r_i^2).$$

Hence, we can use $M = 2(\text{dist}^2(C_i^{\text{in}}, K) + 4r_i^2)$.

We define $\text{cost}_{avg}(C_i^{\text{in}}, K) = \frac{h(C_i^{\text{in}})}{|C_i^{\text{in}}|}$, and $\text{cost}_{avg}(S_i^{\text{in}}, K) = \frac{h(S_i^{\text{in}})}{|S_i^{\text{in}}|}$. Then we set $\delta = \xi M$. Thus, if $s_i^{\text{in}} \geq \frac{1}{2\xi^2} \cdot \ln(4k/\lambda)$, then

$$\mathbf{Pr}\left[|\text{cost}_{avg}(C_i^{\text{in}}, K) - \text{cost}_{avg}(S_i^{\text{in}}, K)| \geq \xi 2(\text{dist}^2(C_i^{\text{in}}, K) + 4r_i^2)\right] \leq \lambda/2k.$$

As $w(C_i^{\text{in}}) = w(S_i^{\text{in}})$ we have with probability at least $1 - \lambda/2k$

$$|\text{cost}(C_i^{\text{in}}, K) - \text{cost}(S_i^{\text{in}}, K)| \leq 2\xi |C_i^{\text{in}}|(\text{dist}^2(C_i^{\text{in}}, K) + 4r_i^2)$$

It is easy to see that $|C_i^{\text{in}}|\text{dist}^2(C_i^{\text{in}}, K) \leq \text{cost}(C_i^{\text{in}}, K)$ and summing this up for all sets C_i^{in} , for $i = 1, \cdots, k$, we have

$$|\text{cost}(\cup_i C_i^{\text{in}}, K) - \text{cost}(\cup_i S_i^{\text{in}}, K)| \leq 2\xi \left(\sum_i \text{cost}(C_i^{\text{in}}, K) + \sum_i |C_i^{\text{in}}| 4r_i^2 \right).$$

As $r_i = \sqrt{\frac{\text{cost}(C_i, c_i)}{\epsilon \cdot |C_i|}}$ and $C_i^{\text{in}} \leq C_i$, we have

$$\begin{aligned} |\text{cost}(\cup_i C_i^{\text{in}}, K) - \text{cost}(\cup_i S_i^{\text{in}}, K)| &\leq 2\xi \left(\sum_i \text{cost}(C_i^{\text{in}}, K) + \sum_i 4 \frac{\text{cost}(C_i, c_i)}{\epsilon} \right) \\ &\leq 2\xi \left(\sum_i \text{cost}(C_i^{\text{in}}, K) + 4 \frac{\mathcal{OPT}(P, k)}{\beta \epsilon} \right). \end{aligned}$$

Recall that β is the approximation factor of the solution $\{c_1, \ldots, c_k\}$. We set $\xi = \beta\epsilon^2/10$. Then we get

$$|\text{cost}(\cup_i C_i^{\text{in}}, K) - \text{cost}(\cup_i S_i^{\text{in}}, K)| \leq \epsilon \left(\sum_i \text{cost}(C_i^{\text{in}}, K) \right) \leq \epsilon \cdot \text{cost}(P, K)$$

which holds with probability at least $1 - \lambda/2$ and for $s_i^{\text{in}} \geq \frac{50}{\beta^2 \epsilon^4} \ln(4k/\lambda)$.

Let $\text{cost}_{avg}(C_i^{\text{out}}, K) = \frac{\text{cost}(C_i^{\text{out}}, K)}{|C_i^{\text{out}}|}$. Define the random variable $X_j = \frac{w_{s_j}}{|C_i^{\text{out}}|} \cdot \text{dist}^2(s_j, K)$ for the average contribution of the jth sample point in S_i^{out} to the nearest center of K. The expected value of $\mathbf{E}[X_j]$ is

$$\begin{aligned} \mathbf{E}[X_j] &= \sum_{q \in C_i^{\text{out}}} p_q \cdot \frac{w_q}{|C_i^{\text{out}}|} \cdot \text{dist}^2(q, K) \\ &= \frac{1}{s_i^{\text{out}} \cdot |C_i^{\text{out}}|} \sum_{q \in C_i^{\text{out}}} \text{dist}^2(q, K) \\ &= \frac{\text{cost}_{avg}(C_i^{\text{out}}, K)}{s_i^{\text{out}}}. \end{aligned}$$

We define $\text{cost}_{avg}(S_i^{\text{out}}, K) = \frac{1}{|C_i^{\text{out}}|} \cdot \sum_{p \in S_i^{\text{out}}} w_p \cdot \text{dist}^2(p, K) = \sum_{j=1}^{s_i^{\text{out}}} X_j$ (notice that the averaging is done by dividing by $|C_i^{\text{out}}|$ and not by the sum of the weights of the points in S_i^{out}).

Hence, $\mathbf{E}[\text{cost}_{avg}(S_i^{out}, K)] = \sum_{j=1}^{s_i^{out}} \mathbf{E}[X_j] = \text{cost}_{avg}(C_i^{out}, K)$. For each $q \in C_i^{out}$ we have

$$\begin{aligned} \text{dist}^2(q, k_l) &\leq 2\left(\text{dist}^2(q, c_i) + \text{dist}^2(c_i, k_l)\right) \\ &\leq 2\left(\text{dist}^2(q, c_i) + (\frac{r_i(1+\epsilon)}{\epsilon})^2\right) \\ &\leq 4\text{dist}^2(q, c_i)\left[\frac{(1+\epsilon)^2}{\epsilon^2}\right]. \end{aligned}$$

Observe that each X_j satisfies

$$\begin{aligned} X_j &= \frac{w_{s_j}}{|C_i^{out}|} \cdot \text{dist}^2(s_j, K) \\ &\leq \frac{w_{s_j}}{|C_i^{out}|} \cdot \text{dist}^2(s_j, k_l) \leq \frac{w_{s_j}}{|C_i^{out}|} \cdot 4\text{dist}^2(s_j, c_i)\left[\frac{(1+\epsilon)^2}{\epsilon^2}\right] \\ &\leq \frac{\text{cost}(C_i^{out}, c_i)}{\text{dist}^2(s_j, c_i) \cdot s_i^{out} \cdot |C_i^{out}|} \cdot 4\text{dist}^2(s_j, c_i)\left[\frac{(1+\epsilon)^2}{\epsilon^2}\right] \\ &\leq 4\left[\frac{(1+\epsilon)^2}{\epsilon^2}\right] \cdot \frac{\text{cost}_{avg}(C_i^{out}, c_i)}{s_i^{out}}. \end{aligned}$$

Define random variable

$$Z_j = \frac{X_j}{4\left[\frac{(1+\epsilon)^2}{\epsilon^2}\right] \cdot \frac{\text{cost}_{avg}(C_i^{out}, c_i)}{s_i^{out}}} \leq 1$$

and let $Z = \sum_{j=1}^{s_i^{out}} Z_j$. Applying Hoeffding bound we have

$$\begin{aligned} & \mathbf{Pr}[|\text{cost}_{avg}(S_i^{out}, K) - \text{cost}_{avg}(C_i^{out}, K)| \geq \epsilon \cdot \text{cost}_{avg}(C_i^{out}, c_i)] \\ = \; & \mathbf{Pr}[|\sum_{j=1}^{s_i^{out}} X_j - \sum_{j=1}^{s_i^{out}} \mathbf{E}[X_j]| \geq \epsilon \cdot \text{cost}_{avg}(C_i^{out}, c_i)] \\ = \; & \mathbf{Pr}[|Z - \mathbf{E}[Z]| \geq \frac{\epsilon^3 s_i^{out}}{4(1+\epsilon^2)\mathbf{E}[Z]}\mathbf{E}[Z]] \\ \leq \; & 2\exp\left(-\frac{\mathbf{E}[Z] \cdot \min(\frac{\epsilon^3 s_i^{out}}{4((1+\epsilon)^2)\mathbf{E}[Z]}, \left(\frac{\epsilon^3 s_i^{out}}{4((1+\epsilon)^2)\mathbf{E}[Z]}\right)^2)}{3}\right) \end{aligned}$$

Choosing $s_i^{out} \geq \frac{12((1+\epsilon)^2)}{\epsilon^3}\ln(4k/\lambda)$ gives

$$\mathbf{Pr}[|\text{cost}_{avg}(S_i^{out}, K) - \text{cost}_{avg}(C_i^{out}, K)| \geq \epsilon \cdot \text{cost}_{avg}(C_i^{out}, c_i)] \leq \lambda/(2k).$$

Multiplying by $|C_i^{out}|$ gives

$$|\text{cost}(S_i^{out}, K) - \text{cost}(C_i^{out}, K)| \leq \epsilon \cdot \text{cost}(C_i^{out}, c_i)$$

with probability at least $1 - \lambda/(2k)$.

Now we can combine cases (a) and (b). Summing up over all sets C_i^{in} and C_i^{out}, for $i = 1, \cdots, k$, there exists a constant c' such that for $s_i^{in}, s_i^{out} \geq c' \cdot \frac{\ln(k/\lambda)}{\epsilon^4}$:

$$\begin{aligned} |\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| &\leq 34\epsilon \cdot \text{cost}(P, C) && (5.3)\\ &\leq 34\epsilon\beta \cdot \text{cost}(P, \mathcal{OPT}) && (5.4)\\ &\leq 34\epsilon\beta \cdot \text{cost}(P, K) && (5.5) \end{aligned}$$

with probability at least $1 - \lambda$ and where $\mathcal{S} = \bigcup_{i=1}^{k} \left\{S_i^{in} \cup S_i^{out}\right\}$ and $\mathcal{OPT}$ is an optimal solution for P. Replacing ϵ by $\epsilon/(34\beta)$ gives Lemma 44.

□

Centers from $\mathcal{T}$ are approximated within a factor $(1 \pm \epsilon)$ Now we want to prove the following lemma.

Lemma 47 *Let P be a set of points in $\mathbb{R}^d$ and let $0 < \epsilon, \delta < 1/2$ and $k \geq 1$ be parameters. Let $\mathcal{S}$ be a weighted set of points sampled from P according to our coreset construction using*

$$s_i^{in}, s_i^{out} \geq c \cdot \frac{k \ln(k/\delta)}{\epsilon^5} \cdot \ln(k/\epsilon \cdot \ln(1/\delta))$$

for some large enough constant c. Let $\mathcal{T}$ be the set of centroids of subsets from $\mathcal{S}$ (with repetition) of size $2/\epsilon$. Then with probability $1 - \delta$ we get

$$\forall K \subseteq \mathcal{T}, |K| = k : \ |\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)| \leq \epsilon \cdot \text{cost}(P, K) \ .$$

Proof. Let $\mathcal{N}$ denote the set of centroids of all subsets from P of size $2/\epsilon$. We say that $K \subseteq \mathcal{N}$ *is well approximated*, if $\left|\text{cost}(\mathcal{S}, K) - \text{cost}(P, K)\right| \leq \epsilon \cdot \text{cost}(P, K)$. We want to show that every set $K \subseteq \mathcal{T}$, $|K| = k$, is also well approximated. Recall that $\mathcal{T} \subseteq \mathcal{N}$ consists of the centroids of all subsets (with repetition) of size $2/\epsilon$ of $\mathcal{S}$. Wlog. we will assume that for each point $p \in \mathcal{T}$ there is a unique multiset $\mu_P^{-1}(p)$ of $2/\epsilon$ points from $\mathcal{S}$ that generates p, i.e. $\mu_P(\mu_P^{-1}(p)) = p$.

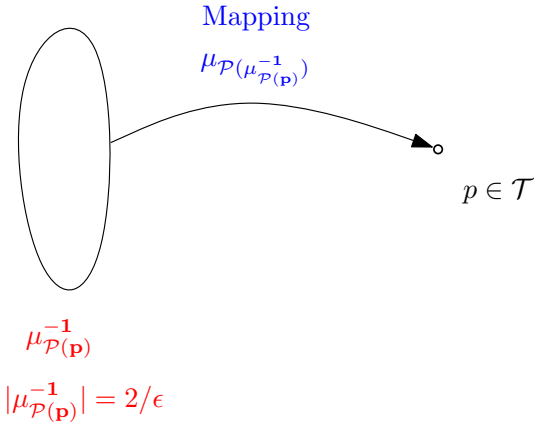


Fig. 5.2: $\mu_P(\mu_P^{-1}(p)) = p$

We cannot directly apply Lemma 44 to show that $K \subseteq \mathcal{T}$ is well approximated, because $K \subseteq \mathcal{T}$ imposes the condition that $\mu_P^{-1}(K) \subseteq \mathcal{S}$, where $\mu_P^{-1}(K) = \bigcup_{p \in K} \mu_P^{-1}(p)$.

Here and in the following we regard both $\mu_P^{-1}(K)$ and $\mathcal{S}$ as (unweighted) multisets, i.e. we replace each point p with weight w_p by w_p copies of p. We assume that all relations between multisets take the multiplicity of points into account. For example, the expression $\mu_P^{-1}(K) \subseteq \mathcal{S}$ implies that if $\mu_P^{-1}(K)$ contains a point multiple times, it appears at least the same number of times in $\mathcal{S}$. Given $\delta > 0$ we want to show that

$$\begin{aligned} & \mathbf{Pr}[\forall K \subseteq \mathcal{T}, |K| = k : K \text{ is well approximated }] \\ = \; & 1 - \mathbf{Pr}[\exists K \subseteq \mathcal{T}, |K| = k : K \text{ is not well approximated }] \\ \geq \; & 1 - \delta \ . \end{aligned}$$

We use the fact that

$$\mathbf{Pr}[\exists K \subseteq \mathcal{T}, |K| = k : K \text{ is not well approximated }] \tag{5.6}$$

$$\leq \sum_{K \subseteq \mathcal{N}, |K| = k} \mathbf{Pr}[K \text{ is not well approximated } \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \cdot \mathbf{Pr}[\mu_P^{-1}(K) \subseteq \mathcal{S}] \ . \tag{5.7}$$

We have

$$\begin{aligned} & \mathbf{Pr}[K \text{ is not } \textbf{well} \text{ approximated } \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \\ \leq \; & \mathbf{Pr}[|\mathrm{cost}(\cup_{i=1}^k S_i^{out}, K) - \mathrm{cost}(\cup_{i=1}^k C_i^{out}, K)| > \epsilon \cdot \mathrm{cost}(\cup_{i=1}^k C_i^{out}, K) \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \\ & + \mathbf{Pr}[|\mathrm{cost}(\cup_{i=1}^k S_i^{in}, K) - \mathrm{cost}(\cup_{i=1}^k C_i^{in}, K)| > \epsilon \cdot \mathrm{cost}(\cup_{i=1}^k C_i^{in}, K) \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \ . \end{aligned}$$

The condition fixes $2k/\epsilon$ points of the sample set. All remaining points are drawn at random according to the specified distribution. Let us denote by F_i^{in} and F_{out}^{in} these random points, i.e.

$$S_i^{in} = F_i^{in} \cup \left(C_i^{in} \cap \mu_P^{-1}(K)\right)$$

and

$$S_i^{out} = F_i^{out} \cup \left(C_i^{out} \cap \mu_P^{-1}(K)\right).$$

We get

$$\mathbf{Pr}[|\mathrm{cost}(S_i^{in}, K) - \mathrm{cost}(C_i^{in}, K)| > \epsilon \cdot \mathrm{cost}(C_i^{in}, K) \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \tag{5.8}$$

$$= \mathbf{Pr}[|\mathrm{cost}(F_i^{in}, K) + \mathrm{cost}(C_i^{in} \cap \mu_P^{-1}(K), K) - \mathrm{cost}(C_i^{in}, K)| > \epsilon \cdot \mathrm{cost}(C_i^{in}, K)] \tag{5.9}$$

$$\leq \mathbf{Pr}[|\mathrm{cost}(F_i^{in}, K) - \mathrm{cost}(C_i^{in}, K)| > \epsilon \cdot \mathrm{cost}(C_i^{in}, K) - \mathrm{cost}(C_i^{in} \cap \mu_P^{-1}(K), K)] \tag{5.10}$$

In a similar way we obtain

$$\mathbf{Pr}[|\mathrm{cost}(S_i^{out}, K) - \mathrm{cost}(C_i^{out}, K)| > \epsilon \cdot \mathrm{cost}(C_i^{out}, K) \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \tag{5.11}$$

$$\leq \mathbf{Pr}[|\mathrm{cost}(F_i^{out}, K) - \mathrm{cost}(C_i^{out}, K)| > \epsilon \cdot \mathrm{cost}(C_i^{out}, K) - \mathrm{cost}(C_i^{out} \cap \mu_P^{-1}(K), K)] \tag{5.12}$$

After rescaling the weights of points in F_i^{in} by $|S_i^{in}|/|F_i^{in}|$ we can apply the proof of Lemma 44. Let Err_i^{in}, Err_i^{out} denote the error bounds derived in the proof of Lemma 44. We distinguish between cases (1) and (2). In case (1) we obtain by Lemma 44 for $Err_i^{in} = 8 \cdot \epsilon \cdot cost(C_i^{in}, K,)$ and $|F_i^{in}| \geq c \cdot \frac{\ln(k/\lambda)}{\epsilon^4}$

$$\begin{aligned}
& \lambda/(2k) \\
\geq\ & \mathbf{Pr}\big[\big|\frac{|S_i^{in}|}{|F_i^{in}|} \cdot cost(F_i^{in}, K) - cost(C_i^{in}, K)\big| > Err_i^{in}\big] \\
=\ & \mathbf{Pr}\big[\big|cost(F_i^{in}, K) - \frac{|F_i^{in}|}{|S_i^{in}|} cost(C_i^{in}, K)\big| > \frac{|F_i^{in}|}{|S_i^{in}|} \cdot Err_i^{in}\big] \\
\geq\ & \mathbf{Pr}\big[\big|cost(F_i^{in}, K) - cost(C_i^{in}, K)\big| > \frac{|F_i^{in}|}{|S_i^{in}|} \cdot Err_i^{in} + (1 - \frac{|F_i^{in}|}{|S_i^{in}|}) \cdot cost(C_i^{in}, K)\big] \\
\geq\ & \mathbf{Pr}\big[\big|cost(F_i^{in}, K) - cost(C_i^{in}, K)\big| > Err_i^{in} + \epsilon \cdot cost(C_i^{in}, K)\big]
\end{aligned}$$

Similarly, we obtain for $Err_i^{out} = 26\epsilon \cdot |C_i^{in}| \cdot dist^2(B(c_i, r_i), k_l)$

$$\begin{aligned}
& \lambda/(2k) \\
\geq\ & \mathbf{Pr}\big[\big|cost(F_i^{out}, K) - cost(C_i^{out}, K)\big| > Err_i^{out} + \epsilon \cdot cost(C_i^{out}, K)\big]
\end{aligned}$$

In a similar way we can obtain bounds for case (2). Summing up over all clusters gives for $\mathcal{F} = \bigcup_{i=1}^{k}(F_i^{in} \cup F_i^{out})$:

$$\mathbf{Pr}\big[\big|cost(\mathcal{F}, K) - cost(P, K)\big| \leq 2\epsilon \cdot cost(P, K)\big] \leq \lambda \ . \tag{5.13}$$

In Lemma (48) we prove $cost(\mu_P^{-1}(K), K)K \leq \epsilon/2 \cdot cost(P, K)$. Then replacing ϵ by $\epsilon/4$ in equation (5.13) and combining it with equations (5.8) and (5.12) gives

$$\mathbf{Pr}[K \text{ is not well approximated} \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \leq \lambda \ .$$

Lemma 48 *For* $s_i^{in}, s_i^{out} \geq \frac{ck}{\epsilon^5}$, *where* $c \geq 8$, *we have*

$$cost(\mu_P^{-1}(K), K) \leq \epsilon/2 \cdot cost(P, K) \ .$$

Proof. The analysis will again distinguish between the cases (1) $dist(k_l, c_i) \geq r_i + \frac{r_i}{\epsilon} = \frac{r_i(1+\epsilon)}{\epsilon}$ and (2) $dist(k_l, c_i) < r_i + \frac{r_i}{\epsilon} = \frac{r_i(1+\epsilon)}{\epsilon}$. We will assume $\epsilon \leq 1/2$.

Case (1) First for C_i^{in}

$$\begin{aligned}
& cost(C_i^{in} \cap \mu_P^{-1}(K), K) \\
\leq\ & \frac{2k}{\epsilon} \frac{|C_i^{in}|}{s_i^{in}} \Big[2\Big((2r_i)^2 + dist^2(B(c_i, r_i), k_l)\Big)\Big] \\
\leq\ & \frac{2k}{\epsilon} \frac{|C_i^{in}|}{\frac{ck}{\epsilon^5}} \Big[2\Big(4\epsilon^2 dist^2(B(c_i, r_i), k_l) + dist^2(B(c_i, r_i), k_l)\Big)\Big] \\
\leq\ & \epsilon^4 |C_i^{in}| \cdot dist^2(B(c_i, r_i), k_l) \leq \epsilon^4 cost(C_i^{in}, K).
\end{aligned}$$

Then for C_i^{out}

$$\begin{aligned}
\text{cost}(C_i^{out} \cap \mu_P^{-1}(K), K) &\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} w_p \cdot \text{cost}(p, k_l) \\
&\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 2w_p \cdot (\text{dist}^2(p, c_i) + \text{dist}^2(c_i, k_l)) \\
&\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 2w_p \cdot (\text{dist}^2(p, c_i) + 2(r_i^2 + \text{dist}^2(B(c_i, r_i), k_l))) \\
&\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 2w_p \cdot (\text{dist}^2(p, c_i) + 2(\text{dist}^2(p, c_i) + \text{dist}^2(B(c_i, r_i), k_l))) \\
&\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 2w_p \cdot (3\text{dist}^2(p, c_i) + 2\text{dist}^2(B(c_i, r_i), k_l)) \\
&\leq 6r_i^2 \epsilon |C_i^{in}| + \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 4w_p \cdot \text{dist}^2(B(c_i, r_i), k_l) \\
&\leq 6\epsilon^3 |C_i^{in}| \text{dist}^2(B(c_i, r_i), k_l) + \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} 4w_p \cdot \text{dist}^2(B(c_i, r_i), k_l) \\
&\leq 2\epsilon |C_i^{in}| \text{dist}^2(B(c_i, r_i), k_l)(3\epsilon^2 + 2) \\
&\leq 6\epsilon |C_i^{in}| \text{dist}^2(B(c_i, r_i), k_l) \leq 6\epsilon \text{cost}(C_i^{in}, k_l).
\end{aligned}$$

Case (2) Again first for C_i^{in} and having $r_i = \sqrt{\frac{\text{cost}(C_i, c_i)}{\epsilon . |C_i|}}$

$$\begin{aligned}
\text{cost}(C_i^{in} \cap \mu_P^{-1}(K), K) &\leq \frac{2k}{\epsilon} \frac{|C_i^{in}|}{s_i^{in}} \left(\frac{r_i(1+\epsilon)}{\epsilon} \right)^2 \\
&\leq \frac{2k}{\epsilon} \frac{|C_i^{in}|}{\frac{ck}{\epsilon^5}} \left(\frac{r_i(1+\epsilon)}{\epsilon} \right)^2 \leq \epsilon^2 |C_i^{in}| r_i^2 \leq \epsilon/6 \cdot \text{cost}(C_i^{in}, c_i).
\end{aligned}$$

Then for C_i^{out}

$$\begin{aligned}
\text{cost}(C_i^{out} \cap \mu_P^{-1}(K), K) &\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} w_p \cdot \text{cost}(p, k_l) \\
&\leq \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} \frac{\text{cost}(C_i^{out}, c_i)}{\frac{ck}{\epsilon^5} \cdot \text{dist}^2(p, c_i)} \cdot \left[2(\text{dist}^2(p, c_i) + \text{dist}^2(c_i, k_l)) \right] \\
&\leq \epsilon^4 \cdot \text{cost}(C_i^{out}, c_i) + \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} \frac{\text{cost}(C_i^{out}, c_i)}{\frac{ck}{\epsilon^5} \cdot \text{dist}^2(p, c_i)} \cdot 2\text{dist}^2(c_i, k_l) \\
&\leq \epsilon^4 \cdot \text{cost}(C_i^{out}, c_i) + \sum_{p \in C_i^{out} \cap \mu_P^{-1}(K)} \frac{\text{cost}(C_i^{out}, c_i)}{\frac{ck}{\epsilon^5} \cdot r_i^2} \cdot 2(\frac{r_i(1+\epsilon)}{\epsilon})^2 \\
&\leq \epsilon^4 \cdot \text{cost}(C_i^{out}, c_i) + \epsilon^2 \cdot \text{cost}(C_i^{out}, c_i) \\
&\leq 2\epsilon^2 \cdot \text{cost}(C_i^{out}, c_i).
\end{aligned}$$

□

Finally, replacing ϵ by $\epsilon/4$ in equation (5.13) and combining it with equations (5.8)and (5.12) gives

$$\mathbf{Pr}[K \text{ is not well approximated} \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \leq \lambda \ .$$

Plugging this into equation (5.7) we get

$$\begin{aligned} & \mathbf{Pr}[\exists K \subseteq \mathcal{T}, |K| = k : K \text{ is not well approximated }] \\ \leq & \sum_{K \subseteq \mathcal{N}, |K|=k} \mathbf{Pr}[K \text{ is not well approximated} \mid \mu_P^{-1}(K) \subseteq \mathcal{S}] \cdot \mathbf{Pr}[\mu_P^{-1}(K) \subseteq \mathcal{S}] \\ \leq & \lambda \cdot \sum_{K \subseteq \mathcal{N}, |K|=k} \mathbf{Pr}[\mu_P^{-1}(K) \subseteq \mathcal{S}] \leq \lambda \cdot |\mathcal{S}|^{2k/\epsilon} \end{aligned}$$

It follows that for $\lambda \leq \delta/|\mathcal{S}|^{2k/\epsilon}$ we obtain the bound stated in the lemma. This is satisfied for

$$s_i^{in}, s_i^{out} \geq c \cdot \frac{k \ln(k/\delta)}{\epsilon^5} \cdot \ln(k/\epsilon \cdot \ln(1/\delta))$$

when c is a sufficiently large constant.

□

The coreset Finally, we put things together.

Theorem 49 *Given a set P of n points in $\mathbb{R}^d$ and parameters $\epsilon, \lambda > 0$ and an appropriate constant $c > 0$, if $\mathcal{S}$ is a weighted set of points obtained by our algorithm, i.e.,* Coreset*, using*

$$s_i^{in}, s_i^{out} \geq c \cdot \frac{k \ln(k/\delta)}{\epsilon^5} \cdot \ln(k/\epsilon \cdot \ln(1/\delta))$$

and $\mathcal{T}$ is the set of centroids of subsets of size $2/\epsilon$, then $(\mathcal{S}, \mathcal{T})$ is a weak (k, ϵ)-coreset for point set P with probability at least $1 - \delta$.

Proof. We apply Lemma 44 to show that the cost of an optimal set of centers is preserved upto a factor of $(1 \pm \epsilon)$. Then we apply Lemma 47 to show that this is true for all sets of centers from $\mathcal{T}$. From Lemma 43 is follows that $\mathcal{T}$ is a $(k, 6\epsilon)$-approximate centroid set with probability $1 - \delta + \lambda$. Replacing ϵ by $\epsilon/6$, δ by $\delta/2$ and λ by $\delta/2$ we obtain the theorem. □

5.2 A linear time $(1+\epsilon)$-approximation algorithm for the k-means

We obtain the following linear time $(1 + \epsilon)$-approximation algorithm for k-Means clustering. We first compute a weak (k, ϵ)-coreset $\mathcal{S}$. Then we do exhaustive search over all subsets of size k from $\mathcal{T}$. We can slightly improve the running time of this approach using dimensionality reduction. The idea is to use Johnson-Lindenstrauss Lemma (See Theorem 32) to map $\mathcal{S}$ to a lower dimensional space. Recall that Johnson-Lindenstrauss Lemma is

Johnson-Lindenstrauss Lemma: Theorem 32 Let $0 < \epsilon \leq 1/2$. Let $P = \{p_1, \cdots, p_n\}$ be a set of n points in $\mathbb{R}^d$. Let $t = O(\epsilon^{-2} \log n)$. Then there exists a linear mapping $T : \mathbb{R}^d \to \mathbb{R}^t$ such that with probability at least $1/2$ fulfills

$$(1 - \epsilon) \cdot \text{dist}(p_i, p_j) \leq \text{dist}(T(p_i), T(p_j)) \leq (1 + \epsilon) \cdot \text{dist}(p_i, p_j)$$

for all $i, j \in [n]$.

We choose the dimension of the target space in such a way that distances between the points in $\mathcal{S} \cup \mathcal{OPT} \cup \mathcal{T}$ are approximated within factor of $(1 + \epsilon)$. Thus, $t = O(\log |\mathcal{T}|/\epsilon^2)$. Since JL-transform is a linear mapping, we know that the centroid of points is mapped to the centroid of the mapped points. Since we may assume that the centroids of subsets of $\mathcal{T}$ of size k are disjoint in the original space they also will be disjoint in the target space (since their mutual distances are preserved upto a factor of $(1 + \epsilon)$). Thus, a centroid of $2/\epsilon$ points in the target space corresponds to a unique point (the centroid of the points in the original space) and so we can map a solution from the target space back to the original space. Finally, to obtain a solution we do exhaustive search in the set of all subsets of $\mathcal{T}$ of size k and evaluate the cost of each solution in the small target space of the JL-transform.

Linear Time $(1 + \epsilon)$-Approximation Algorithm (P)

(1) Let $\mathcal{S}$ denote just the sampled set of size $\frac{c \cdot k^2 \ln^2(k/(\delta\epsilon)}{\epsilon^5}$(without the centroid set) returned by Coreset.

(2) Map the set $\mathcal{S}$ to the set $\mathcal{S}'$ inside the target space $\mathbb{R}^t$ where $t = O(\log |\mathcal{T}|/\epsilon^2)$ using Theorem 32.

(3) $\mathcal{T} =$ The set of all centroids of combinations of $2/\epsilon$ points from $\mathcal{S}'$ (we allow repetition of points).

(4) Let $\mathcal{T}^k$ denote the set found by exhaustive search over all subsets of size k from $\mathcal{T}$.

(5) Let $\mathcal{O}'$ denote an arbitrary set of size k in the target space $\mathbb{R}^t$.

(6) For $i = 1$ to $|\mathcal{T}^k|$ do

  (a) If $\text{cost}(P, \mathcal{T}^k[i]) < \text{cost}(P, \mathcal{O}')$, where $\mathcal{T}^k[i]$ denote the ith subset of size k from $\mathcal{T}^k$,

  (b) Then put $\mathcal{O}' = \mathcal{T}^k[i]$.

(7) Apply the inverse of JL-transform to find a set $\mathcal{O}''$ of size k which mapped to $\mathcal{O}'$.

(8) return $\mathcal{O}''$.

Theorem 50 *Given a set P of n points in $\mathbb{R}^d$ and parameters $\epsilon, \lambda > 0$ and an appropriate constant $c > 0$, there exists a randomized algorithm that computes $(1 + \epsilon)$-approximate k-means clustering of P in time $O(nkd + d \cdot (k/\epsilon)^{O(1)} + 2^{\widetilde{O}(k/\epsilon)})$ with probability at least $1 - \lambda$.*

5.3 Insertion-only Streaming Algorithm

In this section we explain a merge step in level i w.l.o.g. at c_1^i and also set the parameters of the merge and reduce method that we explained in Section 2.5.1. Put $b = 2$ then the depth of the 2-ary tree T is $\log n$. Let $\mathcal{N}$ denote the set of centroids of all subsets from P of size $2/\epsilon$. For the merge step in level i w.l.o.g. at c_1^i, we compute a weak $(k, \epsilon_i = \epsilon/(ci^2))$-coreset $(c_1^i, \mathcal{N} \cup \mathcal{OPT})$ of its two children c_1^{i-1}, c_2^{i-1} via coreset construction of Lemma 44 and this should be a weak (k, ϵ)-coreset with probability $\geq 1 - \lambda_n$ for $\lambda_n = \lambda/n^{2k/\epsilon}$ and for the n points received so far.

We put $m_0 = k^2\epsilon^{-5}\log n$ and $m_i = ck\frac{\ln(k/\lambda_n)}{(\epsilon_i)^4}$ where $\lambda_n = \lambda/n^{2k/\epsilon}$ is the confidence parameter for the n points received so far, i is the level in which bucket size must be m_i, $\epsilon_i = \epsilon/(ci^2)$ is the error parameter at level i , and c is a large positive constant. Therefore the size of a bucket at level i is $m_i = ck\frac{\ln(k/\lambda_n)}{\epsilon_i} = ck^2\frac{i^8\log n}{\epsilon^5}\ln(k/\lambda)$ which is at most $m_{\log n} = ck^2\frac{\log^9 n}{\epsilon^5}\ln(k/\lambda)$ at level $\log n$. Since we need to keep coresets for at most $\log n$ levels therefore the space that we need to save all these coresets is at most $O(k^2\frac{\log^{10} n}{\epsilon^5}\ln(k/\lambda))$ points.

To analyze the update time for k-means, observe that the amortized time dealing with buckets at level 0 is constant; and for $i = 2, \ldots, \log_2 n$, c_1^i is constructed after every $2^{i-1}m_0$ insertions are made. Therefore the amortized time spent for an update is

$$\sum_{i=1}^{\log n} \frac{1}{2^{i-1}m_0} \cdot O(m_{i-1} \cdot dk \cdot \log(n^{2k/\epsilon}/\lambda)) = O(dk^2/\epsilon \log^2 n).$$

Chapter 6

Subspace Clustering

In this chapter we go further into the details of the idea that we explained in Section 3.2. In particular, in Section 6.1 we give a general dimensionality reduction for a broad class of clustering problems. We then apply this dimensionality reduction recursively to find an unbiased estimator for the j-subspace clustering problem in Section 6.2. The following sections give three applications of the unbiased estimator. In Section 6.3 we find the first strong coreset for this problem. Section 6.4 shows how to get a fairly fast linear time $(1+\epsilon)$-approximation algorithm using the estimator and a centorid set. In Section 6.5 we maintain the strong coreset using the merge and reduce method.

6.1 Dimensionality Reduction for Clustering Problems

In this section we present a general dimensionality reduction technique for problems that involve sums of distances as a quality measure. Our result is that for an arbitrary fixed subset $C \subseteq \mathbb{R}^d$, $\text{cost}(P, C)$ can be approximated by a small weighted sample and the projection of P onto a low dimensional subspace. This result can be immediately applied to obtain a dimensionality reduction method for a large class of clustering problems, where the cluster centers are objects contained in low-dimensional spaces. Examples include: k-median clustering, subspace problem under ℓ_1-error, variants of projective clustering and more specialized problems where cluster centers are, for example, discs or curved surfaces.

For these type of problems, we suggest an algorithm that computes a low dimensional weighted point set Q such that, with probability at least $1-\delta$, for any fixed query center C, $\text{cost}(Q, C)$ approximates $\text{cost}(P, C)$ to within a factor of $1 \pm \epsilon$. The algorithm is a generalization of a technique developed in Chapter 5 to compute coresets for the k-means clustering problem. Recall that originally the idea goes back to Example 1 where we do non-uniform sampling according to some prespecified probabilities.

The main new idea that allows us to handle any type of low dimensional center is the use of points that are associated with negative weights. To obtain this result, we first define a randomized algorithm DIMREDUCTION . For a given (low-dimensional) subspace C^* and a

parameter $\epsilon > 0$, the algorithm DIMREDUCTION computes a weighted point set Q, such that most of the points of Q lie on C^*, and for any fixed query center C we have $\mathbf{E}[\text{cost}(Q, C)] = \text{cost}(P, C)$. Then we show that, with probability at least $1 - \delta$, the $\text{cost}(Q, C)$ has an additive error of at most $\epsilon \cdot \text{cost}(P, C^*)$.

We can then apply this result to low dimensional clustering problems in two steps. First, we observe that, if each center is a low dimensional object, i.e. is contained in a low dimensional j-subspace, then k centers are contained in a (kj)-subspace and so clustering them is at least as expensive as $\text{cost}(P, C')$, where C' is a (kj)-subspace that minimizes $\text{cost}(P, C')$. Thus, if we compute an α-approximation C^* for the (kj)-dimensional subspace approximation problem, and replace ϵ by ϵ/α, we obtain the result outlined above.

DIMREDUCTION $(P, C^*, \delta, \epsilon)$

(1) Pick $r = \left\lceil \frac{2\log(2/\delta)}{\epsilon^2} \right\rceil$ points $s_1, \ldots, s_r$ i.i.d. from P, s.t. each $p \in P$ is chosen with probability $\mathbf{Pr}[p] = \frac{\text{dist}(p, C^*)}{\text{cost}\left(P, C^*\right)}$.

(2) For $i \leftarrow 1$ to r do $w(s_i) \leftarrow \dfrac{1}{r \cdot \mathbf{Pr}[s_i]}$

(3) Return the multiset $Q = \text{proj}(P, C^*) \cup \{s_1, \ldots, s_r\} \cup \{\text{proj}(s_1^-, C^*), \ldots, \text{proj}(s_r^-, C^*)\}$, where s_i^- is the point s_i with weight $-w(s_i)$ and $\text{proj}(s_i^-, C^*)$ is the orthogonal projection of s_i^- onto C^*.

Analysis of Algorithm DIMREDUCTION. Let us fix an arbitrary set C. Our first step will be the following technical lemma that shows that the expectation of the random variable $\text{cost}(Q, C)$ is $\text{cost}(P, C)$. Let X_i denote the random variable for the sum of contributions of the sample points s_i and $\text{proj}(s_i^-, C)$ to C, i.e.

$$\begin{aligned} X_i &= w(s_i) \cdot \text{dist}(s_i, C) + w(s_i^-) \cdot \text{dist}(\text{proj}(s_i^-, C)) \\ &= w(s_i) \cdot \big(\text{dist}(s_i, C) - \text{dist}(\text{proj}(s_i, C^*), C)\big). \end{aligned}$$

Lemma 51 *Let P be a set of points in $\mathbb{R}^d$. Let $\epsilon > 0$, $0 < \delta \leq 1$, and Q be the weighted set that is returned by the randomized algorithm* DIMREDUCTION $(P, C^*, \delta, \epsilon)$. *Then*

$$\mathbf{E}[\text{cost}(Q, C)] = \text{cost}(P, C) \ .$$

Proof. We have

$$\begin{aligned} \mathbf{E}[X_i] &= \sum_{p \in P} \mathbf{Pr}[p] \cdot w(p) \big(\text{dist}(p, C) - \text{dist}(\text{proj}(p, C^*), C)\big) \\ &= \sum_{p \in P} \frac{1}{r} \frac{1}{\mathbf{Pr}[p]} \cdot \mathbf{Pr}[p] \big(\text{dist}(p, C) - \text{dist}(\text{proj}(p, C^*), C)\big) \\ &= \frac{1}{r} \cdot \big(\text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C)\big) \ . \end{aligned}$$

By linearity of expectation we have

$$\mathbf{E}[\sum_{i=1}^{r} X_i] = \text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C) \ .$$

Since algorithm DIMREDUCTION computes the union of $\text{proj}(P, C^*)$ and the points s_i and $\bar{s_i}$, we obtain

$$\begin{aligned} \mathbf{E}[\text{cost}(Q, C)] &= \text{cost}(\text{proj}(P, C^*), C) + \mathbf{E}[\sum_{i=1}^{r} X_i] \\ &= \text{cost}(P, C). \end{aligned}$$

The lemma follows. □

Our next step is to show that $\text{cost}(Q, C)$ is sharply concentrated about its mean.

Theorem 52 *Let P be a set of n points in $\mathbb{R}^d$, and let C^* be a j-subspace. Let $0 < \delta, \epsilon \leq 1$, and Q be the weighted point set that is returned by the algorithm* DIMREDUCTION *$(P, C^*, \delta, \epsilon)$. Then for a fixed query set $C \subseteq \mathbb{R}^d$ we have*

$$|\text{cost}\big(P, C\big) - \text{cost}\big(Q, C\big)| \leq \epsilon \cdot \text{cost}(P, C^*),$$

with probability at least $1 - \delta$. Moreover, only

$$r = O\left(\frac{\log(1/\delta)}{\epsilon^2}\right)$$

points of Q are not contained in $\text{proj}(P, C^*)$. This algorithm runs in $O(ndj + r)$ time.

Proof. Let $P = \{p_1, \ldots, p_n\}$ be a set of n points in $\mathbb{R}^d$. We first prove the concentration bound and then discuss the running time.

In order to apply additive Hoeffding inequality we need to determine the range of values X_i can attain. By the triangle inequality we have

$$\text{dist}(s_i, C) \leq \text{dist}(s_i, C^*) + \text{dist}(\text{proj}(s_i, C^*), C)$$

and

$$\text{dist}(\text{proj}(s_i, C^*), C) \leq \text{dist}(s_i, C) + \text{dist}(s_i, C^*).$$

This implies

$$|\text{dist}(s_i, C) - \text{dist}(\text{proj}(s_i, C^*), C)| \leq \text{dist}(s_i, C^*).$$

We then have

$$\begin{aligned} |X_i| &= \left|w(s_i) \cdot \big(\text{dist}(s_i, C) - \text{dist}(\text{proj}(s_i, C^*), C)\big)\right| \\ &\leq w(s_i) \cdot \text{dist}(s_i, C^*) = \frac{\text{cost}(P, C^*)}{r}. \end{aligned}$$

Thus, $-\text{cost}(P, C^*)/r \leq X_i \leq \text{cost}(P, C^*)/r$. Using additive Hoeffding inequality and similar to Example 1 we set $M = \text{cost}(P, C^*), t = \epsilon \cdot \text{cost}(P, C^*), m = r$ to get the result.

In order to achieve the stated running time, we proceed as follows. We first compute in $O(ndj)$ time for each point $p \in P$ its distance $\text{dist}(p, C^*)$ to C^* and store it. This can easily be done by first computing an orthonormal basis of C^*. We sum these distances in order to obtain $\text{cost}(P, C^*)$ in $O(n)$ time. From this we can also compute $\mathbf{Pr}[p]$ and $w(p)$ for each $p \in P$, in $O(n)$ overall time. We let P be the array of probabilities $p_1, \ldots, p_n$. It is well known that one can obtain a set of r samples according to a distribution given as a length-n array in $O(n + r)$ time, see [Vos91]. □

6.2 The Unbiased Estimator

In this section we show how to use Theorem 52 to obtain a small weighted set S that, with probability at least $1 - \delta$, approximates the cost to an arbitrary fixed j-subspace, i.e., we show that $\text{cost}(S, C^*)$ is an unbiased estimator for $\text{cost}(P, C^*)$. The first step of the algorithm is to apply our dimensionality reduction procedure with a j-subspace C_j^* that is, with probability at least $2/3$ an $O(j^{j+1})$-approximation to the optimal j-dimensional linear subspace with respect to the ℓ_1-error. Such an approximation can be computed in $O(ndj)$ time using the algorithm APPROXIMATEVOLUMESAMPLING by Deshpande and Varadarajan [DV07]. Note that the success probability can be amplified to $1 - \delta$ in time $O(ndj \log(1/\delta))$. Once we have projected all the points on C_j^*, we apply the same procedure using a $(j-1)$-dimensional linear subspace C_{j-1}^*. We continue this process until all the points are projected onto a 0-dimensional linear subspace, i.e. the origin. As we will see, this procedure can be used to approximate the cost of a fixed j-subspace C.

ADAPTIVESAMPLING (P, j, δ, ϵ)

(1) $P_{j+1} \leftarrow P$.

(2) **For** $i = j$ **Downto** 0

  (a) $C_i^* \leftarrow$ APPROXIMATEVOLUMESAMPLING(P_{i+1}, i).

  (b) $Q_i \leftarrow$ DIMREDUCTION $(P_{i+1}, C_i^*, \delta, \epsilon)$.

  (c) $P_i \leftarrow \text{proj}(P_{i+1}, C_i^*)$.

  (d) $S_i \leftarrow Q_i \setminus P_i$, where S_i consists of the positively and negatively weighted sample points.

(3) Return $S = \bigcup_{i=0}^{j} S_i$.

Note that P_0 is the origin, and so $\text{cost}(P_0, C) = 0$ for any j-subspace C. Let C_i^* be an arbitrary but fixed sequence of linear subspaces as used in the algorithm.

Theorem 53 *Let P be a set of n points in $\mathbb{R}^d$, and $\epsilon', \delta' > 0$. Let C be an arbitrary j-dimensional linear subspace. If we call algorithm* ADAPTIVESAMPLING *with the parameters $\delta = O(\delta'/(j+1))$ and $\epsilon = \epsilon'/j^{c \cdot j^2}$ for a large enough constant c, then we get*

$$(1-\epsilon') \cdot \mathrm{cost}(P, C) \leq \mathrm{cost}(S, C) \leq (1+\epsilon') \cdot \mathrm{cost}(P, C),$$

with probability at least $1 - \delta'$. The running time of the algorithm is

$$O(ndj(j + \log(1/\delta)) + \frac{j^{O(j^2)} \log(1/\delta')}{\epsilon'^2}).$$

Proof. Let C be an arbitrary j-subspace. We split the proof of Theorem 53 into two parts. The first and easy part is to show that $\mathrm{cost}(S, C)$ is an unbiased estimator of $\mathrm{cost}(P, C)$. The hard part is to prove that $\mathrm{cost}(S, C)$ is sharply concentrated.

We can apply Lemma 51 with $C^* = C_i^*$ to obtain that for any $1 \leq i \leq j$ we have $\mathbf{E}[\mathrm{cost}(Q_i, C)] = \mathrm{cost}(P_{i+1}, C)$ and hence

$$\mathbf{E}[\mathrm{cost}(S_i, C)] = \mathrm{cost}(P_{i+1}, C) - \mathrm{cost}(P_i, C) \ .$$

Therefore,

$$\begin{aligned} \mathbf{E}[\mathrm{cost}(S, C)] &= \sum_{i=0}^{j} \mathbf{E}[\mathrm{cost}(S_i, C)] \\ &= \mathrm{cost}(P_{j+1}, C) - \mathrm{cost}(P_0, C) \\ &= \mathrm{cost}(P, C) \ , \end{aligned}$$

where the last equality follows from $P_{j+1} = P$ and P_0 being a set of n points at the origin.

Now we show that $\mathrm{cost}(S, C)$ is sharply concentrated. We have

$$\left|\mathbf{E}[\mathrm{cost}(S, C)] - \mathrm{cost}(S, C)\right| \leq \sum_{i=0}^{j} \left|\mathbf{E}[\mathrm{cost}(S_i, C)] - \mathrm{cost}(S_i, C)\right| \ .$$

The following observation was used in [FFS06] for $j = 1$, and generalized later in [FL07].

Lemma 54 *Let C be a j-subspace, and L be an $(i+1)$-subspace, such that $i + 1 \leq j$. Then there exists an i-subspace C_i, and a constant $0 < \nu_L \leq 1$, such that for any $p \in L$ we have $\mathrm{dist}(p, C) = \nu_L \cdot \mathrm{dist}(p, C_i)$.*

Let $0 \leq i \leq j$. By substituting $L = \mathrm{Span}\{P_{i+1}\}$ in Lemma 54, there is an i-subspace C_i and a constant ν_L, such that

$$\begin{aligned} &\left|\mathbf{E}[\mathrm{cost}(S_i, C)] - \mathrm{cost}(S_i, C)\right| \\ &= \left|\mathrm{cost}(P_{i+1}, C) - \mathrm{cost}(P_i, C) - \mathrm{cost}(S_i, C)\right| \\ &= \nu_L \cdot |\mathrm{cost}(P_{i+1}, C_i) - \mathrm{cost}(P_i, C_i) - \mathrm{cost}(S_i, C_i)| \\ &= \nu_L \cdot |\mathrm{cost}(P_{i+1}, C_i) - \mathrm{cost}(Q_i, C_i)| \ . \end{aligned}$$

Here, the second equality follows from the fact that the solution computed by approximate volume sampling is spanned by input points and so $P_i, S_i \subseteq \mathrm{Span}\{P_{i+1}\}$. We apply Theorem 52 with $C = C_i$ and $C^* = C_i^*$ to obtain

$$|\mathrm{cost}(P_{i+1}, C_i) - \mathrm{cost}(Q_i, C_i)| \leq \epsilon \cdot \mathrm{cost}(P_{i+1}, C_i^*),$$

with probability at least $1 - \delta$. By our choice of C_i^*, we also have

$$\mathrm{cost}(P_{i+1}, C_i^*) \leq O(i^{i+1}) \cdot \mathrm{cost}(P_{i+1}, C_i).$$

Combining the last three inequalities yields

$$\begin{aligned} |\mathbf{E}[\mathrm{cost}(S_i, C)] - \mathrm{cost}(S_i, C)| &\leq \nu_L \cdot \epsilon \cdot \mathrm{cost}(P_{i+1}, C_i^*) \\ &\leq O(\nu_L \cdot \epsilon \cdot i^{i+1}) \cdot \mathrm{cost}(P_{i+1}, C_i) \\ &= O(\epsilon \cdot i^{i+1}) \cdot \mathrm{cost}(P_{i+1}, C) \ , \end{aligned}$$

with probability at least $1 - \delta$. Hence,

$$\begin{aligned} \left|\mathbf{E}[\mathrm{cost}(S, C)] - \mathrm{cost}(S, C)\right| &\leq O\left(\sum_{i=0}^{j-1} \epsilon \cdot i^{i+1} \cdot \mathrm{cost}(P_{i+1}, C)\right) \\ &\leq O\left(\sum_{i=0}^{j-1} \epsilon \cdot i^{i+1} \cdot (i+1)^{i+2} \cdot \mathrm{cost}(P_{i+2}, C)\right) \\ &\leq O\left(\sum_{i=0}^{j-1} \epsilon \cdot i^{i+1} \cdot (i+1)^{i+2} \cdots (j)^{j+1} \cdot \mathrm{cost}(P_{j+1}, C)\right) \\ &\leq \epsilon \cdot j^{O(j^2)} \cdot \mathrm{cost}(P, C), \end{aligned}$$

with probability at least $1 - j \cdot \delta$. Note that the second and the third inequality is because when we project P_{i+1} onto C_i^* to get P_i the sum of distances between a point in P_{i+1} and its corresponding point in P_i is an $O(i^{i+1})$-approximation of the optimal i-dimensional subspace of P_{i+1} inside the $(i+1)$-subspace C_{i+1}^* since we are using the algorithm APPROXIMATEVOLUMESAMPLING(P_{i+1}, i) for this projection. Therefore, for our choice of δ and ϵ with probability at least $1 - j \cdot \delta$ we get

$$\left|\mathbf{E}[\mathrm{cost}(S, C)] - \mathrm{cost}(S, C)\right| \leq \epsilon \cdot j^{O(j^2)} \cdot \mathrm{cost}(P, C).$$

Now we analyze the running time. Inside the i-th step of the loop in Step 2 we have the following operations:

(1) We envoke the algorithm APPROXIMATEVOLUMESAMPLING(P_{i+1}, i) in Step 2a which takes $O(ndi)$. We repeat $O(\log(1/\delta))$ times the process to amplify the sucess probability.

(2) Similar to Theorem 52 the random sampling in Step 2b is done in time $O(n + r)$ for $r = \left\lceil \frac{2\log(2/\delta)}{\epsilon^2} \right\rceil$ where $\delta = O(\delta'/(j+1))$ and $\epsilon = \epsilon'/j^{c \cdot j^2}$.

(3) The projection in Step 2c takes $O(ndi)$.

The summation over j steps of the loop in Step 2 gives the stated running time. □

6.3 Coresets

As the first application of the unbiased estimator that we developed in the previous section, we get a strong coreset in this section. In order to construct a coreset, we only have to run the algorithm ADAPTIVESAMPLING using small enough δ. One can compute δ by discretizing the space near the input points using a sufficiently fine grid. Then snapping a given subspace to the nearest grid points will not change the cost of the subspace significantly. If a subspace does not intersect the space near the input points, its cost will be high and the overall error can be easily charged.

Theorem 55 *Let P denote a set of n points in $\mathbb{R}^d$, $j \geq 0$, and $1 > \epsilon', \delta' > 0$, $d \leq n$. Let Q be the weighted set that is returned by the algorithm* ADAPTIVESAMPLING *with the parameters $\delta = \frac{1}{j} \cdot \delta'/(10nd)^{10dj}$ and $\epsilon = \epsilon'/(j+1)^{c \cdot j^2}$ for a large enough constant c. Then, with probability at least $1 - \delta' - 1/n^2$, Q is a strong ϵ-coreset. The size of the coreset in terms of the number of (weighted) points saved is $O(dj^{O(j^2)} \cdot \epsilon'^{-2} \log n)$.*

The following algorithm returns a grid $G \subseteq \text{Span}\{P\}$ and we first prove some auxiliary lemmas for this grid. The algorithm ADAPTIVESAMPLING gives an unbiased estimator for one fixed j-subspace spanned by j-points of G. We replace δ' by $\delta = \frac{1}{j} \cdot \delta'/(10nd)^{10dj}$ to prove this estimator is an unbiased estimator for all j-subspaces spanned by j points in G. This in turn gives the strong ϵ-coreset that we claimed in Theorem 55 when we invoke the algorithm $\text{NET}(P \cup \text{proj}(P, C^*), M, \epsilon')$, using $M = 10 \cdot \text{cost}(P, C^*)/\epsilon$, and $\epsilon' = \epsilon \cdot \text{cost}(P, C^*)/n^{10}$ as we will see in the proof of Proposition 60.

NET (P, M, ϵ)

(1) $G \leftarrow \emptyset$.

(2) **For** each $p \in P$ **Do**

  (a) $G_p \leftarrow$ vertex set of a d-dimensional grid that is centered at p. The side length of the grid is $2M$, and the side length of each cell is $\epsilon/(2\sqrt{d})$.

  (b) $G \leftarrow G \cup G_p$.

(3) **Return** G.

Lemma 56 *Let P be a set of points in a subspace A of $\mathbb{R}^d$. Let $\epsilon > 0$, $M > \epsilon$ denote two parameters. Let $G \subseteq A$ denote the grid returned by Algorithm* $\text{NET}(P, M, \epsilon)$ *such that for every point $c \in A$, if $\text{dist}(c, P) \leq 2M$ then $\text{dist}(c, G) \leq \epsilon/2$. Let $C \subseteq A$ be a 1-subspace (i.e, a line that intersects the origin of $\mathbb{R}^d$), such that $\text{dist}(p, C) \leq M$ for every $p \in P$. Then there is a 1-subspace D that is spanned by a point in G, such that,*

$$|\text{dist}(p, C) - \text{dist}(p, D)| \leq \epsilon,$$

for every $p \in P$.

Proof. Let g be a point such that the angle between the lines C and $\mathrm{Span}\{g\}$ is minimized over $g \in G$. Let $D = \mathrm{Span}\{g\}$, and $p \in P$. We prove the lemma using the following case analysis: **(i)** $\mathrm{dist}(p, D) \geq \mathrm{dist}(p, C)$, and **(ii)** $\mathrm{dist}(p, D) < \mathrm{dist}(p, C)$.

(i) $\mathrm{dist}(p, D) \geq \mathrm{dist}(p, C)$: Let $c = \mathrm{proj}(p, C)$. We have $\mathrm{dist}(c, P) \leq \|c - p\| = \mathrm{dist}(p, C) \leq M$. By the assumption of the lemma, we thus have $\mathrm{dist}(c, G) \leq \epsilon$. By the construction of D, we also have $\mathrm{dist}(c, D) \leq \mathrm{dist}(c, G)$. Combining the last two inequalities yields $\mathrm{dist}(c, D) \leq \epsilon$. Hence

$$\mathrm{dist}(p, D) \leq \|p - c\| + \mathrm{dist}(c, D) \leq \mathrm{dist}(p, C) + \epsilon.$$

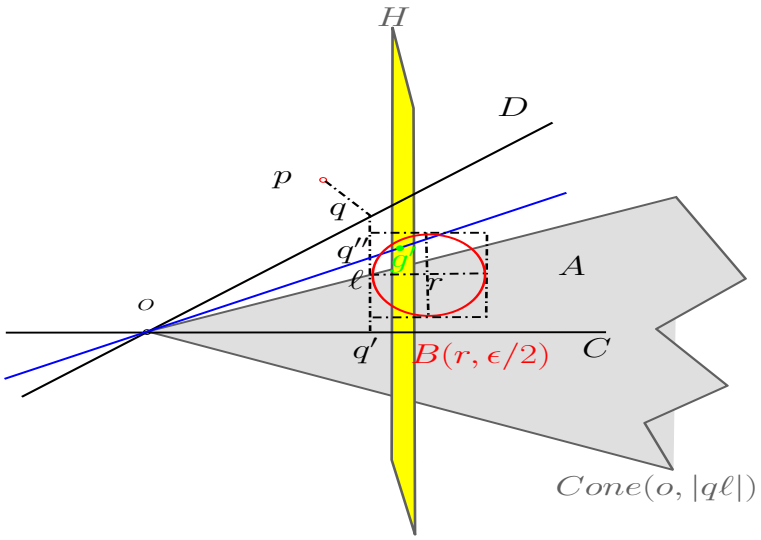


Fig. 6.1: Case ii. $\mathrm{dist}(p, D) < \mathrm{dist}(p, C)$

(ii) $\mathrm{dist}(p, D) < \mathrm{dist}(p, C)$: Let $q = \mathrm{proj}(p, D)$, and $q' = \mathrm{proj}(q, C)$. We can assume that $\mathrm{dist}(q, q') > \epsilon$ since otherwise by the triangle inequality, $\mathrm{dist}(p, C) \leq \mathrm{dist}(p, q) + \mathrm{dist}(q, q') \leq \mathrm{dist}(p, D) + \epsilon$, and we are done.

Define $\ell = \frac{q+q'}{2}$. Let C_ℓ denote the parallel line to C at ℓ. Consider the cylinder with the axis C_ℓ of radius $\epsilon/2$. Let H_1 denote a hyperplane at q' perpendicular to C. Let q'_ϵ denote a point which is at distance ϵ from q on C, in the halfspace of H_1 which does not contain O. Let H_2 denote a hyperplane at q'_ϵ and perpendicular to C. The H_1 and H_2 cut the cylinder at two surfaces and the volume in between these two hyperplanes and the cylinder can be packed inside a hypercube A of side length ϵ.

Let r denote the center of A and let $B(r, \epsilon/2)$ denote a ball centered at r of radius $\epsilon/2$. See Figure 6.1.

We observe that

$$\mathrm{dist}(p, r) \leq \mathrm{dist}(p, q) + \mathrm{dist}(q, \ell) + \mathrm{dist}(\ell, r).$$

We also have that $\mathrm{dist}(p, q) = \mathrm{dist}(p, D) < \mathrm{dist}(p, C) - \epsilon$ otherwise we are done and nothing left to prove. Therefore, $\mathrm{dist}(p, q) < M - \epsilon$. On the other hand $\mathrm{dist}(q, \ell) \leq \mathrm{dist}(p, C) \leq M$ and $\mathrm{dist}(\ell, r) = \epsilon/2$. Thus we get that

$$\mathrm{dist}(p, r) \leq M - \epsilon + M + \epsilon/2 \leq 2M.$$

Using $M > \epsilon$, the assumption of lemma implies $\text{dist}(r, G) < \epsilon/2$ which means that there is a grid point $g' \in G$ such that $\text{dist}(r, g') < \epsilon/2$ and therefore $g' \in B(r, \epsilon)$. Now let us consider a line which is perpendicular to C at q' and intersects $\text{Span}\{g'\}$ at q''. Such a line exists since C and $\text{Span}\{g'\}$ intersects at o. Consider a cone with apex at o and axis C that touches ℓ. Note that the extension of the segment $q'\bar{q}''$ intersects this cone at ℓ. Let H be the hyperplane which goes through g' and is perpendicular to C. Let ℓ' be the extension of ℓ that intersects H. Consider the triangles $\Delta o\ell q''$ and $\Delta o\ell' g'$. Now we use the intercept theorem for these two triangles. Recall that $g' \in B(r, \epsilon/2)$ and we have $|q''\ell| \leq |g'\ell'| \leq |g'r| \leq \epsilon/2$ which means that

$$|q'q''| \leq |q'\ell| + |q''\ell| \leq \frac{|qq'|}{2} + \frac{\epsilon}{2} < |qq'|.$$

Thus we obtain $\sin(\angle(\text{Span}\{g'\}, C)) = \frac{|q''q'|}{|oq'|} < \sin(\angle(\text{Span}\{g\}, C)) = \frac{|qq'|}{|oq'|}$, contradicting the choice of g. □

The following is a generalization of the last lemma for $j > 1$.

Lemma 57 *Let P be a set of points in a subspace A of $\mathbb{R}^d$. Let $\epsilon > 0$, $M > \epsilon$ denote two parameters. Let $G \subseteq A$ denote the grid returned by Algorithm $\text{NET}(P, M, \epsilon)$ such that for every point $c \in A$, if $\text{dist}(c, P) \leq 2M$ then $\text{dist}(c, G) \leq \epsilon/2$. Let C be a j-subspace, such that $\text{dist}(p, C) \leq M - (j-1)\epsilon$ for every $p \in P$. Then there is a j-subspace D that is spanned by j points from G, such that*

$$|\text{dist}(p, C) - \text{dist}(p, D)| \leq j\epsilon,$$

for every $p \in P$.

Proof. The proof is by induction on j. The base case of $j = 1$ is furnished by substituting $A = \mathbb{R}^d$ in Lemma 56. We now give a proof for the case $j \geq 2$. Let $e_1, \cdots, e_j, e_{j+1}, \cdots, e_d$ denote an orthonormal set of $\mathbb{R}^d$ in which the first j of them are orthogonal unit vectors of C. Let E denote the subspace that is spanned by $e_1, \cdots, e_{j-1}$ and let $E^\perp$ be the orthogonal complement of E.

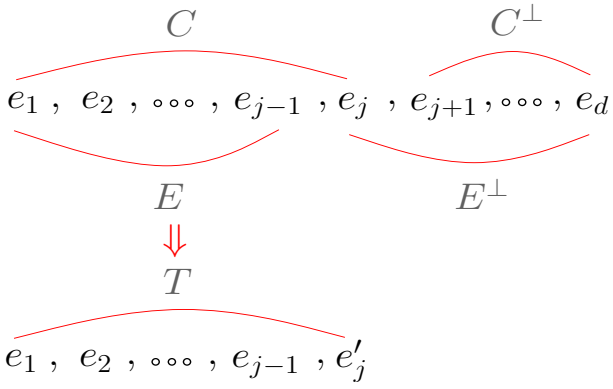


Fig. 6.2: Orthonormal basis of $\mathbb{R}^d$ and its subspaces shown by their orthonormal bases

Fix $p \in P$. Assume that in this coordinate system the point p is shown as $p = (p_1 = p^T \cdot e_1, \cdots, p_d = p^T \cdot e_d)$. The key observation is that for any j-subspace T in $\mathbb{R}^d$ that contains

$e_1, \cdots, e_{j-1}$, we have

$$\mathrm{dist}(p, T) = \mathrm{dist}(\mathrm{proj}(p, E^{\perp}), \mathrm{proj}(T, E^{\perp})).$$

In fact in this new coordinate system, we have $\mathrm{proj}(p, E^{\perp}) = (p_j, \cdots, p_d)^T$ and for the j-subspace T, $\mathrm{proj}(T, E^{\perp})$ is a 1-subspace $o\nu_T$ for $\nu_T = (e_j^T e_j', \cdots, e_d^T e_j')^T \in E^{\perp}$.

Let $P' = \mathrm{proj}(P, E^{\perp})$, and let $c = (c_1, \cdots, c_d) \in \mathbb{R}^d$ be such that $\mathrm{dist}(c, P) \leq 2M$. Hence, there is a point $q = (q_1, \cdots, q_d) \in P$ such that

$$\|q - c\| = \mathrm{dist}(c, P) \leq 2M. \tag{6.1}$$

Then by the assumption of this lemma we have that $\mathrm{dist}(c, G) \leq \epsilon/2$. Assume that a point $r \in G$ is the nearest point in G to c, so we have $\|c - r\| \leq \epsilon/2$.

Let $\mathrm{proj}(q, E^{\perp}) = q' = (q_j, \cdots, q_d)$, $\mathrm{proj}(c, E^{\perp}) = c' = (c_j, \cdots, c_d)$, and $\mathrm{proj}(r, E^{\perp}) = r' = (r_j, \cdots, r_d)$.

Hence,

$$\|q' - c'\| \leq \|q - c\| \leq 2M,$$

$$\|c' - r'\| \leq \|c - r\| \leq \epsilon/2,$$

which means $\mathrm{dist}\big(c', \mathrm{proj}(G, E^{\perp})\big) \leq \|c' - r'\| \leq \epsilon/2$.

From the previous paragraph, we conclude that for every $c' \in E^{\perp}$, if $\mathrm{dist}(c', P') \leq 2M$ then $\mathrm{dist}\big(c', \mathrm{proj}(G, C^{\perp})\big) \leq \epsilon/2$. Clearly, we also have $\mathrm{dist}(\mathrm{proj}(p, E^{\perp}), \mathrm{proj}(C, E^{\perp})) = \mathrm{dist}(p, C) \leq M$. Using this, we apply Lemma 56 while replacing A with $E^{\perp}$, P with P', C with $\mathrm{proj}(C, E^{\perp})$ and G with $\mathrm{proj}(G, E^{\perp})$.

We obtain that there is a 1-subspace $D \subseteq E^{\perp}$ that is spanned by a point from $\mathrm{proj}(G, E^{\perp})$, such that

$$|\mathrm{dist}\big(\mathrm{proj}(p, E^{\perp}), \mathrm{proj}(C, E^{\perp})\big) - \mathrm{dist}\big(\mathrm{proj}(p, E^{\perp}), D\big)| \leq \epsilon.$$

Since $\mathrm{dist}\big(\mathrm{proj}(p, E^{\perp}), \mathrm{proj}(C, E^{\perp})\big) = \mathrm{dist}(p, C)$ by the definition of $E^{\perp}$, the last inequality implies

$$|\mathrm{dist}(p, C) - \mathrm{dist}\big(\mathrm{proj}(p, E^{\perp}), D\big)| \leq \epsilon. \tag{6.2}$$

Let F be the j-subspace of $\mathbb{R}^d$ that is spanned by D and $e_1, \cdots, e_{j-1}$. Let $D^{\perp}$ be the $(d-1)$-subspace that is the orthogonal complement of D in $\mathbb{R}^d$. Since $D \subseteq F$, we have that $\mathrm{proj}(F, D^{\perp})$ is a $(j-1)$-subspace of $\mathbb{R}^d$. We thus have

$$\begin{aligned} \mathrm{dist}(\mathrm{proj}(p, E^{\perp}), D) &= \mathrm{dist}(\mathrm{proj}(p, D^{\perp}), \mathrm{proj}(F, D^{\perp})) \\ &= \mathrm{dist}(p, F). \end{aligned} \tag{6.3}$$

Using (6.2), with the assumption of this lemma that $\mathrm{dist}(p, C) \leq M - (j-1)\epsilon$, yields

$$\begin{aligned} \mathrm{dist}(\mathrm{proj}(p, E^{\perp}), D) &\leq \mathrm{dist}(p, C) + \epsilon \\ &\leq M - (j-2)\epsilon. \end{aligned}$$

By the last inequality and (6.3), we get

$$\text{dist}(\text{proj}(p, D^{\perp}), \text{proj}(F, D^{\perp})) \leq M - (j-2)\epsilon. \tag{6.4}$$

Similar to the case $P' = \text{proj}(P, C^{\perp})$ that was already proven, we can prove that for $P'' = \text{proj}(P, D^{\perp})$ and $c'' \in D^{\perp}$, we have that if $\text{dist}(c'', P'') \leq 2M$ then $\text{dist}(c'', \text{proj}(G, D^{\perp}) \leq \epsilon/2$. Using this and (6.4), we apply this lemma inductively with C as $\text{proj}(F, D^{\perp})$, G as $\text{proj}(G, D^{\perp})$ and P as $P'' = \text{proj}(P, D^{\perp})$, to obtain a $(j-1)$-subspace U that is spanned by $j-1$ points from $\text{proj}(G, D^{\perp})$, such that

$$|\text{dist}(\text{proj}(p, D^{\perp}), \text{proj}(F, D^{\perp})) - \text{dist}(\text{proj}(p, D^{\perp}), U)| \leq (j-1)\epsilon.$$

Hence,

$$\begin{aligned} &|\text{dist}(p, F) - \text{dist}(\text{proj}(p, D^{\perp}), U)| = \\ &|\text{dist}(\text{proj}(p, D^{\perp}), \text{proj}(F, D^{\perp})) - \text{dist}(\text{proj}(p, D^{\perp}), U)| \\ &\leq (j-1)\epsilon. \end{aligned} \tag{6.5}$$

Let R be the j-subspace of $\mathbb{R}^d$ that is spanned by D and U. Hence, R is spanned by j points of G. We have

$$\begin{aligned} &|\text{dist}(p, C) - \text{dist}(p, R)| \\ &= |\text{dist}(p, C) - \text{dist}(\text{proj}(p, D^{\perp}), U)| \\ &\leq |\text{dist}(p, C) - \text{dist}(p, F)| + |\text{dist}(p, F) - \text{dist}(\text{proj}(p, D^{\perp}), U)|. \end{aligned}$$

By (6.3), we have $\text{dist}(p, F) = \text{dist}(\text{proj}(p, E^{\perp}), D)$. Together with the previous inequality, we obtain

$$\begin{aligned} &|\text{dist}(p, C) - \text{dist}(p, R)| \\ &\leq |\text{dist}(p, C) - \text{dist}(\text{proj}(p, E^{\perp}), D)| + |\text{dist}(p, F) - \text{dist}(\text{proj}(p, D^{\perp}), U)|. \end{aligned}$$

Combining (6.2) and (6.5) in the last inequality proves the lemma. □

Lemma 58 *Let $0 < \epsilon, \delta' < 1$, and P be a set of n points in $\mathbb{R}^d$ with $d \leq n$. Let C^* be a j-subspace that is, with probability at least $2/3$ an $O(j^{j+1})$-approximation to the optimal j-dimensional linear subspace with respect to the ℓ_1-error. Let Q be the weighted set that is returned by the algorithm* DIMREDUCTION$(P, C^*, \delta, \epsilon)$ *with the parameter $\delta = \delta'/(10nd)^{10jd}$. Then, with probability at least $1 - \delta' - 1/n^2$, for every j-subspace $C \subseteq \mathbb{R}^d$ we have (simultaneously)*

$$|\text{cost}(P, C) - \text{cost}(Q, C)| \leq \epsilon \cdot \text{cost}(P, C^*) + \epsilon \cdot \text{cost}(P, C).$$

The following two propositions prove the lemma.

Proposition 59 *For every j-subspace C of $\mathbb{R}^d$ such that*

$$\text{cost}(P, C) > 2\text{cost}(P, C^*)/\epsilon,$$

we have

$$|\text{cost}(P, C) - \text{cost}(Q, C)| \leq \epsilon \cdot \text{cost}(P, C).$$

Proof. Let C be a j-subspace such that

$$\text{cost}(P, C) > 2\text{cost}(P, C^*)/\epsilon.$$

Let $S = Q \setminus \text{proj}(P, C^*)$. Hence,

$$\begin{aligned} |\text{cost}(P, C) - \text{cost}(Q, C)| &= |\text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C) - \text{cost}(S, C)| \\ &\leq |\text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C)| + |\text{cost}(S, C)|. \end{aligned} \tag{6.6}$$

We now bound each term in the right hand side of (6.6).

Let s_i denote the ith point of S, $1 \leq i \leq |S|$. By the triangle inequality,

$$|\text{dist}(s_i, C) - \text{dist}(\text{proj}(s_i, C^*), C)| \leq \text{dist}(s_i, C^*),$$

for every $1 \leq i \leq |S|$. Hence,

$$\begin{aligned} |\text{cost}(S, C)| &= \left| \sum_{1 \leq i \leq |S|} w(s_i)(\text{dist}(s_i, C) - \text{dist}(\text{proj}(s_i, C^*), C)) \right| \\ &\leq \sum_{1 \leq i \leq |S|} w(s_i)|\text{dist}(s_i, C^*)| = \text{cost}(P, C^*). \end{aligned}$$

Similarly,

$$\begin{aligned} |\text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C)| &= \left| \sum_{p \in P} \text{dist}(p, C) - \sum_{p \in P} \text{dist}(\text{proj}(p, C^*), C) \right| \\ &\leq \sum_{p \in P} \text{dist}(p, C^*) \\ &= \text{cost}(P, C^*). \end{aligned}$$

Combining the last two inequalities in (6.6) yields

$$\begin{aligned} |\text{cost}(P, C) - \text{cost}(Q, C)| &\leq |\text{cost}(P, C) - \text{cost}(\text{proj}(P, C^*), C)| + |\text{cost}(S, C)| \\ &\leq 2\text{cost}(P, C^*) \leq \epsilon \cdot \text{cost}(P, C). \end{aligned}$$

□

Proposition 60 *Let $0 < \epsilon < 1$ and $d \leq n$. With probability at least*

$$1 - \delta' - 1/n^2,$$

for every j-subspace C such that

$$\text{cost}(P, C) \leq 2\text{cost}(P, C^*)/\epsilon,$$

we have (simultaneously)

$$|\text{cost}(P, C) - \text{cost}(Q, C)| \leq \epsilon \cdot \text{cost}(P, C) + \epsilon\text{cost}(P, C^*).$$

Proof. Let G denote the set that is returned by the algorithm $\text{NET}(P \cup \text{proj}(P, C^*), M, \epsilon')$, where $M = 10\text{cost}(P, C^*)/\epsilon$, and $\epsilon' = \epsilon\text{cost}(P, C^*)/n^{10}$. Note that G is used only for the proof of this proposition.

By Theorem 52, for a fixed center $D \in G$ we have

$$\begin{aligned} |\text{cost}(P, D) - \text{cost}(Q, D)| &\leq \epsilon \cdot \text{cost}(P, D) \\ &\leq \epsilon \cdot \text{cost}(P, C) + \epsilon \cdot |\text{cost}(P, C) - \text{cost}(P, D)|, \end{aligned} \tag{6.7}$$

with probability at least

$$1 - \delta \geq 1 - \frac{\delta'}{(10nd)^{10jd}} \geq 1 - \frac{\delta'}{|G|^j}.$$

Using the union bound, (6.7) holds simultaneously for every j-subspace D that is spanned by j points from G, with probability at least $1 - \delta'$.

Let $p \in P$. By the assumption of this claim, we have

$$\text{dist}(p, C) \leq \text{cost}(P, C) \leq 2\text{cost}(P, C^*)/\epsilon,$$

and also

$$\begin{aligned} \text{dist}(\text{proj}(p, C^*), C) &\leq \|\text{proj}(p, C^*) - p\| + \text{dist}(p, C) \\ &\leq \text{dist}(p, C^*) + \frac{2\text{cost}(P, C^*)}{\epsilon} \\ &\leq \frac{3\text{cost}(P, C^*)}{\epsilon}. \end{aligned}$$

By the last two inequalities, for every $p \in P \cup \text{proj}(P, C^*)$ we have

$$\begin{aligned} \text{dist}(p, C) \leq \frac{3\text{cost}(P, C^*)}{\epsilon} &\leq \frac{10\text{cost}(P, C^*)}{\epsilon} - \frac{\text{cost}(P, C^*)}{\epsilon} \\ &\leq M - (j-1)\epsilon', \end{aligned}$$

where in the last derivation we used the assumption $j \leq d \leq n$ and $0 \leq \epsilon \leq 1$. By the construction of G, for every $c \in \mathbb{R}^d$, if $\text{dist}(c, P) \leq 2M$, then $\text{dist}(c, G) \leq \epsilon'/2$. Using this, applying Lemma 57 with $P \cup \text{proj}(P, C^*)$ yields that there is a j-subspace D that is spanned by j points from G, such that

$$|\text{dist}(p, C) - \text{dist}(p, D)| \leq j \cdot \epsilon',$$

for every $p \in P \cup \text{proj}(P, C^*)$. Using the last equation with (6.7) yields

$$\begin{aligned} &|\text{cost}(P, C) - \text{cost}(Q, C)| \\ &\leq |\text{cost}(P, C) - \text{cost}(P, D)| + |\text{cost}(P, D) - \text{cost}(Q, D)| + |\text{cost}(Q, D) - \text{cost}(Q, C)| \\ &\leq (1 + \epsilon)|\text{cost}(P, C) - \text{cost}(P, D)| + \epsilon\text{cost}(P, C) + |\text{cost}(Q, D) - \text{cost}(Q, C)| \\ &\leq \epsilon\text{cost}(P, C) + 3 \sum_{p \in P \cup Q} |w(p)| \cdot |\text{dist}(p, C) - \text{dist}(p, D)| \\ &\leq \epsilon\text{cost}(P, C) + 3j\epsilon' \sum_{p \in P \cup Q} |w(p)|, \end{aligned} \tag{6.8}$$

with probability at least $1 - \delta'$.

Let $s \in S$ be such that $w(s) > 0$. By the construction of S, we have

$$\text{dist}(s, C^*) \geq \text{cost}(P, C^*)/(n^2|S|)$$

with probability at least $1 - 1/(n^2|S|)$. Hence, with probability at least $1 - 1/n^2$, for every $s \in S$ we have

$$|w(s)| = \frac{\text{cost}(P, C^*)}{|S|\text{dist}(s, C^*)} \leq n^2.$$

Combining the last two equations with (6.8) yields

$$\begin{aligned} |\text{cost}(P, C) - \text{cost}(Q, C)| &\leq \epsilon \text{cost}(P, C) + 3j\epsilon' \sum_{p \in P \cup Q} |w(p)| \\ &\leq \epsilon \text{cost}(P, C) + \epsilon \text{cost}(P, C^*), \end{aligned}$$

with probability at least $1 - 1/n^2 - \delta'$, as desired. □

Proof. [of Theorem 55] Let P_i, S_i, Q_i and C_i^* denote the set that are defined in the ith iteration of ADAPTIVESAMPLING, for every $0 \leq i \leq j$. For every i, $0 \leq i \leq j$, we have $|S_i| = O(\log(1/\delta)/\epsilon^2)$. Hence,

$$|Q| = \bigcup_{0 \leq i \leq j} S_i = O\left(\frac{j \log(1/\delta)}{\epsilon^2}\right) \leq j^{O(j^2)} \cdot \frac{\log(1/\delta')}{\epsilon'^2}.$$

This bounds the size of Q. For the correctness, let $0 \leq i \leq j$.

Fix $0 \leq i \leq j$. By the previous lemma and our choice of δ, we conclude that, with probability at least $1 - \delta'/j - 1/n^2$, for any j-subspace C we have for our choice of ϵ (assuming c' large enough)

$$\begin{aligned} |\text{cost}(P_{i+1}, C) - \text{cost}(Q_i, C)| &\leq \epsilon \text{cost}(P_{i+1}, C) + \epsilon \text{cost}(P_{i+1}, C_i^*) \\ &\leq O(\frac{\epsilon'}{j^{j+1}}) \text{cost}(P_{i+1}, C) + O(\frac{\epsilon'}{j^{j+1}}) \text{cost}(P_{i+1}, C_i^*). \end{aligned}$$

By construction of C_i^*, we have

$$\begin{aligned} \text{cost}(P_{i+1}, C_i^*) &\leq O(j^{j+1}) \min_{C'} \text{cost}(P_{i+1}, C') \\ &\leq O(j^{j+1}) \text{cost}(P_{i+1}, C). \end{aligned}$$

Combining the last two inequalities yields

$$|\text{cost}(P_{i+1}, C) - \text{cost}(Q_i, C)| \leq O(\frac{\epsilon'}{j^{j+1}}) \cdot \text{cost}(P_{i+1}, C),$$

with probability at least $1 - \delta'/j - 1/n^2$.

Summing the last equation over all the j iterations of ADAPTIVESAMPLING yields

$$\begin{aligned}
|\mathrm{cost}(P,C) - \mathrm{cost}(Q,C)| &= |\mathrm{cost}(P,C) - \mathrm{cost}(\bigcup_{0\le i\le j} S_i, C)| \\
&= |\sum_{0\le i\le j} \big(\mathrm{cost}(P_{i+1},C) - \mathrm{cost}(P_i,C) - \mathrm{cost}(S_i,C)\big)| \\
&= |\sum_{0\le i\le j} \big(\mathrm{cost}(P_{i+1},C) - \mathrm{cost}(Q_i,C)\big)| \\
&\le O(\frac{\epsilon'}{j^{j+1}}) \sum_{0\le i\le j} \mathrm{cost}(P_{i+1},C),
\end{aligned}$$

with probability at least $1-\delta'-1/n^2$.

By Lemma 54, there is an i-subspace C_i and a constant $0 < \nu_L \le 1$, such that for any $p \in L$ we have $\mathrm{dist}(p,C) = \nu_L \cdot \mathrm{dist}(p,C_i)$. Hence,

$$|\mathrm{cost}(P_i,C) - \mathrm{cost}(P_{i+1},C)| = \nu_L \cdot |\mathrm{cost}(P_i,C_i) - \mathrm{cost}(P_{i+1},C_i)|.$$

We thus have

$$\begin{aligned}
\mathrm{cost}(P_i,C) &\le \mathrm{cost}(P_{i+1},C) + \mathrm{cost}(P_i,C) - \mathrm{cost}(P_{i+1},C) \\
&\le \mathrm{cost}(P_{i+1},C) + |\mathrm{cost}(P_i,C) - \mathrm{cost}(P_{i+1},C)| \\
&= \mathrm{cost}(P_{i+1},C) + \nu_L \cdot |\mathrm{cost}(P_i,C_i) - \mathrm{cost}(P_{i+1},C_i)| \\
&\le \mathrm{cost}(P_{i+1},C) + \nu_L \cdot \mathrm{cost}(P_{i+1},C_i^*) \\
&\le \mathrm{cost}(P_{i+1},C) + \nu_L \cdot O(i^{i+1}) \cdot \mathrm{cost}(P_{i+1},C_i) \\
&= \mathrm{cost}(P_{i+1},C) + O(i^{i+1}) \cdot \mathrm{cost}(P_{i+1},C) \\
&= O(i^{i+1}) \cdot \mathrm{cost}(P_{i+1},C)
\end{aligned}$$

Hence,

$$\mathrm{cost}(P_{i+1},C) \le O(j^{j+1})\mathrm{cost}(P,C)$$

for every $0 \le i \le j$.
Combining the last inequalities together yields,

$$\mathbf{Pr}[|\mathrm{cost}(P,C) - \mathrm{cost}(Q,C)| \le \epsilon'\mathrm{cost}(P,C)] \ge 1-\delta'-1/n^2.$$

□

6.4 A linear time $(1+\epsilon)$-approximation algorithm for the j-subspace problem

In this section we show how to construct in $O(nd \cdot \mathrm{poly}(j/\epsilon) + (n+d) \cdot 2^{\mathrm{poly}(j/\epsilon)})$ time, a small set $\mathfrak{C}$ of centroid solutions (i.e., j-subspaces) such that $\mathfrak{C}$ contains a $(1+\epsilon/3)$-approximation to the j-subspace problem, i.e., for the point set P, one of the j-subspaces in $\mathfrak{C}$ is a $(1+\epsilon/3)$-approximation to the optimal j-subspace. Given such a centroid set $\mathfrak{C}$, we run the algorithm

Notation	Meaning
$\mathfrak{C}$	The set of centroid solutions (i.e., j-subspaces)
A	The poly(j/ϵ)-subspace that contains a $(1+\epsilon/6)$-approximation
$\mathcal{OPT}$	The cost of an optimal subspace (not necessarily contained in A)
C_i	A j-subspace which is a $(1+\gamma)^i$-approximation to the optimal j-subspace of P
H_i	An i-subspace which is the span of i points from $G_{\le i}$ where $H_i \subseteq C_i$
$H_i^\perp$	The orthogonal complement of the linear subspace H_i in $\mathbb{R}^d$
C_i^*	The projection of C_i on $H_i^\perp$
N_i	$\{p \in P_{i+1} : \text{dist}(p, C_i) \le 2 \cdot \text{cost}(P, C_i) \cdot \mathbf{Pr}[p]\}$
r_l	A point in $N_i \subseteq H_i^\perp$ that has $\mathbf{Pr}[r_l] > 0$
q in case 1	$\text{proj}(r_l, C_i^*)$
q in case 2	A point in $C_i^* \cap B(\text{proj}(r_l, C_i^*), 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp)$ s.t. $\text{dist}(q, 0) \ge 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$
q'	A point in $\mathcal{N}(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp, \gamma/20)$ s.t $\text{dist}(q, q') \le \frac{\gamma}{2} \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$
ℓ	$\text{Span}\{q\}$
ℓ'	$\text{Span}\{q'\}$
$C_i^\perp$	The orthogonal complement of ℓ in C_i
L_i	The orthogonal complement of $C_i^\perp$ in $\mathbb{R}^d$
C_{i+1}	A j-subspace which is the span of $C_i^\perp$ and ℓ'
$\mathcal{N}(p, R, A, \gamma)$	A γ-net of a ball $B(p, R, A)$ in the subspace A with radius R centered at p

Table 6.1: Notation in Section 6.4.

ADAPTIVESAMPLING with parameters $\delta/|\mathfrak{C}|$ and $\epsilon/6$. By the union bound it follows that every $C \in \mathfrak{C}$ is approximated by a factor of $(1 \pm \epsilon/6)$ with probability at least $1 - \delta$. It follows that the cost of the optimal centroid solution in $\mathfrak{C}$ is a $1 + O(\epsilon)$-approximation to the cost of the optimal j-subspace of the original set of points P.

The intuition behind the algorithm and the analysis. The first step of the algorithm is to invoke approximate volume sampling due to Deshpande and Varadarajan [DV07] to obtain in $O(nd \cdot \text{poly}(j/\epsilon))$ time, an $\widetilde{O}(j^4 + (j/\epsilon)^3)$-dimensional subspace A that contains a $(1+\epsilon/6)$-approximation j-subspace. We use C_0 to denote a linear j-dimensional subspace of A with

$$\text{cost}(P, C_0) \le (1 + \epsilon/6) \cdot \mathcal{OPT}.$$

Our centroid set $\mathfrak{C}$ will consist of subspaces of A. Then the algorithm proceeds in j phases. In phase i, the algorithm computes a set G_i of points in A. We define $G_{\le i} = \bigcup_{1 \le l \le i} G_l$. The algorithm maintains, with probability at least $1 - \frac{i \cdot \delta}{j}$, the invariant that i points from $G_{\le i}$ span an i-subspace H_i such that there exists another j-subspace C_i, $H_i \subseteq C_i \subseteq A$, with

$$\begin{aligned} \text{cost}(P, C_i) &\le (1 + \epsilon/6) \cdot (1+\gamma)^i \cdot \mathcal{OPT} \\ &\le (1 + \epsilon/3) \cdot \mathcal{OPT}, \end{aligned}$$

where $\mathcal{OPT}$ is the cost of an optimal subspace (not necessarily contained in A) and $\gamma = \epsilon/(12j)$ is an approximation parameter. The centroid set $\mathfrak{C}$ will be the set of every j-tuples (i.e., subsets of size j) of $G_{\le j}$.

6.4.1 The algorithm.

In the following, we present our algorithm to compute the centroid set. We use $H_i^{\perp}$ to denote the orthogonal complement of a linear subspace H_i in $\mathbb{R}^d$. We use $\mathcal{N}(p, R, A, \gamma)$ to denote a γ-net of a ball $B(p, R, A)$ in the subspace A with radius R centered at p, i.e. a set of points such that for every point $t \in B(p, R, A)$ there exists a point q in $\mathcal{N}(p, R, A, \gamma)$ with $\mathrm{dist}(t, q) \leq \gamma R$. It is easy to see that a γ-net of a ball $B(p, R, A)$ of size $O(\sqrt{d'}/\gamma^{d'})$ (See [AS00]) exists, where d' is the dimension of A. The input to the algorithm is the point set $P' = \mathrm{proj}(P, A)$ in the space A, an i-dimensional linear subspace H_i and the parameters i and γ. The algorithm is invoked with $i = 0$ and $H_i = C_0$ and j being the dimension of the subspace that is sought. Notice that the algorithm can be carried out in the space A since $H_i \subseteq A$ and so the projection of P' to $H_i^{\perp}$ will be inside A. Note, that although the algorithm doesn't know the cost $\mathcal{OPT}$ of an optimal solution, it is easy to compute the cost of an $O(j^{j+1})$-approximation using approximate volume sampling. From this approximation we can generate $O(j \log j)$ guesses for $\mathcal{OPT}$, one of which includes a constant factor approximation.

CENTROIDSET (P', H_i, i, j, γ)

(1) **if** $i = j$ **then return** H_i.

(2) $P_{i+1} \leftarrow \mathrm{proj}(P', H_i^{\perp})$.

(3) Sample $s = \lceil \log(j/\delta) \rceil$ points $r_1, \ldots, r_s$ i.i.d. from P_{i+1} s.t. each $p \in P_{i+1}$ is chosen with probability $\mathbf{Pr}[p] = \mathrm{dist}(p, 0) / \sum_{q \in P_{i+1}} \mathrm{dist}(q, 0)$

(4) $G_{i+1} \leftarrow \bigcup_{l=1}^{s} \mathcal{N}(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^{\perp}, \gamma/20)$.

(5) **return** $\bigcup_{q \in G_{i+1}}$ CENTROIDSET$(P', \mathrm{Span}\{H_i \cup q\}, i+1, j, \gamma)$.

6.4.2 Invariant of algorithm CENTROIDSET.

We will prove that the algorithm satisfies the following lemma.

Lemma 61 *Let $C_i \subseteq A$ be a subspace that contains H_i. Assume that C_i is a $(1+\gamma)^i$-approximation to the optimal j-subspace of P. Then, with probability at least $1 - \delta/j$, there is a j-subspace $C_{i+1} \subseteq A$ containing H_i and a point from G_{i+1}, such that C_{i+1} is a $(1+\gamma)^{i+1}$ approximation to the optimal j-subspace of P.*

Once the lemma is proved, we can apply it inductively to show that with probability at least $1-\delta$ we have a subspace C_j that is spanned by j points from $G_{\leq j}$ and that has

$$\begin{aligned}
\mathrm{cost}(P, C_j) &\leq (1+\gamma)^j \cdot \mathrm{cost}(P, C_0) \\
&\leq (1+\epsilon/6) \cdot (1+\gamma)^j \cdot \mathcal{OPT} \\
&\leq (1+\epsilon/6) \cdot (1+\epsilon/12) \cdot \mathcal{OPT} \\
&\leq (1+\epsilon/3) \cdot \mathcal{OPT}.
\end{aligned}$$

The running time of the algorithm is dominated by the projections in line 2, j of which are carried out for each element of the centroid set. Since the input P' to the algorithm is in the subspace A, its running time is $n \cdot 2^{\text{poly}(j/\epsilon)}$. To initialize the algorithm, we have to compute space A and project all points on A. This can be done in $O(nd \cdot \text{poly}(j/\epsilon))$ time [DV07].

Finally, we run algorithm ADAPTIVESAMPLING to approximate the cost for every centroid solution generated by algorithm CENTROIDSET. For each centroid solution, we have to project all points on its span. This can be done in $O(d \cdot 2^{\text{poly}(j/\epsilon)})$ time, since the number of centroid solutions is $2^{\text{poly}(j/\epsilon)}$ and the size of the sample is $\text{poly}(j/\epsilon)$. Thus we can summarize the result in the following theorem setting $\delta = 1/6$ in the approximate volume sampling and in our algorithm.

Theorem 62 *Let P be a set of n points in $\mathbb{R}^d$, $0 < \epsilon < 1$ and $1 \leq j \leq d$. A $(1+\epsilon)$-approximation for the j-subspace approximation problem can be computed, with probability at least $2/3$, in time*

$$O(nd \cdot \text{poly}(j/\epsilon) + (n+d) \cdot 2^{\text{poly}(j/\epsilon)}).$$

6.4.3 Overview of the proof of Lemma 61.

The basic idea of the proof follows earlier results of [SV07]. We show that by sampling with probability proportional to the distance from the origin, we can find a point p whose distance to the optimal solution is only a constant factor more than the weighted average distance (where the weighting is done according to the distance from the origin). If we then consider a ball with radius a constant times the average weighted distance and that is centered at p, then this ball must intersect the projection of the current space C_i on $H_i^{\perp}$. If we now place a sufficiently fine net on this ball, then there must be a point q of this net that is close to the projection. We can then define a certain rotation of the current subspace to contain q to obtain the new subspace C_{i+1}. This rotation increases the cost only slightly and C_{i+1} contains $\text{Span}\{H_i \cup \{q\}\}$.

6.4.4 The complete proof of Lemma 61.

We assume that there is a j-subspace C_i, $H_i \subseteq C_i \subseteq A$, with

$$\text{cost}(P, C_i) \leq (1+\gamma)^i \cdot \text{cost}(P, C_0) \leq (1+\epsilon/3) \cdot \mathcal{OPT}.$$

We use C_i^* to denote the projection of C_i on $H_i^{\perp}$. Note that C_i^* has $j-i$ dimensions as $H_i \subseteq C_i$. The idea is to find a point q from $G_{i+1} \subseteq H_i^{\perp} \cap A$ such that we can rotate C_i^* in a certain way to contain q and this rotation will not change the cost with respect to P significantly. Let

$$N_i = \{p \in P_{i+1} : \text{dist}(p, C_i) \leq 2 \cdot \text{cost}(P, C_i) \cdot \mathbf{Pr}[p]\}.$$

N_i contains all points that are close to the subspace C_i, where closeness is defined relative to the distance from the origin. We will first show that by sampling points with probability proportional to their distance from the origin, we are likely to find a point from N_i.

Proposition 63

$$\mathbf{Pr}[\exists r_l, 1 \leq l \leq s : r_l \in N_i] \geq 1 - \delta/j \ .$$

Proof. We first prove by contradiction that the probability to sample a point from N_i is at least $1/2$. Assume that

$$\sum_{p \in P_{i+1} \setminus N_i} \mathbf{Pr}[p] > 1/2.$$

Observe that $\text{cost}(P, C_i) \geq \text{cost}(P', C_i)$ since $C_i \subseteq A$ and $P' = \text{proj}(P, A)$. Further, $\text{cost}(P', C_i) = \text{cost}(P_{i+1}, C_i)$ since $P_{i+1} = \text{proj}(P', H_i^{\perp})$ and $H_i \subseteq C_i$. It follows that

$$\begin{aligned}
\text{cost}(P, C_i) &\geq \text{cost}(P', C_i) = \text{cost}(P_{i+1}, C_i) \\
&\geq \sum_{p \in P_{i+1} \setminus N_i} \text{dist}(p, C_i) \\
&> 2 \cdot \text{cost}(P, C_i) \cdot \sum_{p \in P_{i+1} \setminus N_i} \mathbf{Pr}[p] \\
&> \text{cost}(P, C_i),
\end{aligned}$$

which is a contradiction. Hence,

$$\mathbf{Pr}[r_l \in N_i] = \sum_{p \in N_i} \mathbf{Pr}[p] \geq 1/2.$$

It follows that

$$\begin{aligned}
\mathbf{Pr}[\exists l, 1 \leq l \leq s : r_l \in N_i] &\geq 1 - (1 - 1/2)^s \\
&\geq 1 - \delta/j.
\end{aligned}$$

□

We now make a case distinction in order to prove Lemma 61.

Case 1: Points are on average much closer to C_i than to the origin.
We first consider the case that

$$\sum_{p \in P_{i+1}} \text{dist}(p, 0) \geq 4 \sum_{p \in P} \text{dist}(p, C_i).$$

In this case, the points in N_i are much closer to C_i than to the origin. Note that this inequality is defined in terms of $\text{cost}(P, C_i)$ which is an upper bound for $\text{cost}(P_{i+1}, C_i)$ since

$$\text{cost}(P, C_i) \geq \text{cost}(P', C_i) = \text{cost}(P_{i+1}, C_i).$$

Now let r_l be a point from $N_i \subseteq H_i^{\perp}$ that has $\mathbf{Pr}[r_l] > 0$. Since $C_i \subseteq A$ and

$$\begin{aligned}
\text{dist}(r_l, C_i^*) = \text{dist}(r_l, C_i) &\leq 2 \cdot \text{cost}(P, C_i) \cdot \mathbf{Pr}[r_l] \\
&\leq 2 \cdot \text{cost}(P, C_i) \cdot \mathbf{Pr}[r_l] \\
&\leq 2 \cdot (1 + \epsilon/6) \cdot (1 + \gamma)^i \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l] \\
&\leq 4 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]
\end{aligned}$$

for $\gamma = \epsilon/(12j)$ we get that $B(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^{\perp})$ intersects C_i^*. This also implies that $q := \text{proj}(r_l, C_i^*)$ lies in $B(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^{\perp})$. Hence, there is a point

$$q' \in \mathcal{N}(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^{\perp}, \gamma/20)$$

with $\text{dist}(q, q') \leq \frac{\gamma}{2} \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$.

Let ℓ be the line through q and let ℓ' be the line through q'. Let $C_i^{\perp}$ denote the orthogonal complement of ℓ in C_i. Define the subspace C_{i+1} as the span of $C_i^{\perp}$ and ℓ'. Since q lies in C_i^* (and hence in $H_i^{\perp}$) we have that $C_i^{\perp}$ contains H_i. Hence, C_{i+1} also contains H_i. It remains to show that

$$\text{cost}(P, C_{i+1}) \leq (1+\gamma) \cdot \text{cost}(P, C_i).$$

We have

$$\begin{aligned}
\text{cost}(P, C_{i+1}) - \text{cost}(P, C_i) &\leq \sum_{p \in P} \text{dist}(\text{proj}(p, C_i), C_{i+1}) && (6.9)\\
&= \sum_{p \in P} \text{dist}(\text{proj}(\text{proj}(p, A), C_i), C_{i+1}) && (6.10)\\
&= \sum_{p \in P'} \text{dist}(\text{proj}(p, C_i), C_{i+1}) && (6.11)\\
&= \sum_{p \in P_{i+1}} \text{dist}(\text{proj}(p, C_i), C_{i+1}) && (6.12)
\end{aligned}$$

where Step 6.10 follows from the fact that $C_i \subseteq A$ and so $\text{proj}(\text{proj}(p, A), C_i) = \text{proj}(p, C_i)$ for all $p \in \mathbb{R}^d$. In fact assume that the orthonormal of A is $v_1, \cdots, v_{|A|}$ and the orthonormal of C_i is $v_1, \cdots, v_j$. Then $\text{proj}(p, A)$ is the first $|A|$ coordinates $p_1, \cdots, p_{|A|}$ of p; therefore $\text{proj}(\text{proj}(p, A), C_i)$ the same as $\text{proj}(p, C_i)$ is just the first j coordinates $p_1 \cdots, p_j$. Step 6.12 follows from $H_i \subseteq C_i, C_{i+1}$.

Now define L_i to be the orthogonal complement of $C_i^{\perp}$ in $\mathbb{R}^d$. Note that for any $p \in \mathbb{R}^d$ and its projection $p' = \text{proj}(p, L_i)$ we have $\text{dist}(p, C_i) = \text{dist}(p', C_i)$ and $\text{dist}(p, C_{i+1}) = \text{dist}(p', C_{i+1})$. Further observe that C_i corresponds to the line ℓ in L_i and C_{i+1} corresponds to a line $\ell'' = \text{proj}(\ell', L_i)$. Define α to be the angle between ℓ and ℓ' and β the angle between ℓ and ℓ''. Since the projection of q' onto L_i shrinks the distance between this projection and q we have $\alpha \geq \beta$. Then

$$\begin{aligned}
\text{dist}(\text{proj}(p, C_i), C_{i+1}) &= \text{dist}(\text{proj}(\text{proj}(p, C_i), L_i), \ell'')\\
&= \text{dist}(\text{proj}(p, \ell), \ell'').
\end{aligned}$$

This implies

$$\begin{aligned}
\text{dist}(\text{proj}(p, \ell), \ell'') &= \text{dist}(\text{proj}(p, \ell), 0) \cdot \sin\beta\\
&\leq \text{dist}(p, 0) \cdot \sin\alpha.
\end{aligned}$$

We need the following claim that the distance of q to the origin is not much smaller than the distance of r_l to the origin.

Proposition 64 *If*

$$\sum_{p \in P_{i+1}} \text{dist}(p, 0) \geq 4 \sum_{p \in P} \text{dist}(p, C_i)$$

then

$$\text{dist}(q, 0) \geq \frac{1}{2}\text{dist}(r_l, 0).$$

Proof. Since $r_l \in N_i$ we have

$$\mathrm{dist}(r_l, C_i) \leq 2\mathrm{cost}(P, C_i)\frac{\mathrm{dist}(r_l, 0)}{\sum_{p\in P_{i+1}} \mathrm{dist}(p, 0)}.$$

By our assumption we have

$$\sum_{p\in P_{i+1}} \mathrm{dist}(p, 0) \geq 4 \sum_{p\in P} \mathrm{dist}(p, C_i),$$

which implies $\mathrm{dist}(r_l, C_i) \leq \frac{1}{2}\mathrm{dist}(r_l, 0)$ by plugging in into the previous inequality. We further have $\mathrm{dist}(r_l, C_i) = \mathrm{dist}(r_l, C_i^*)$ and so

$$\mathrm{dist}(q, 0) \geq \mathrm{dist}(r_l, 0) - \mathrm{dist}(r_l, C_i^*) \geq \frac{1}{2}\mathrm{dist}(r_l, 0)$$

by the triangle inequality. □

We get

$$\begin{aligned}
\sin\alpha &\leq \frac{\mathrm{dist}(q, q')}{\mathrm{dist}(q, 0)} \\
&\leq \frac{1/2 \cdot \gamma \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]}{1/2 \cdot \mathrm{dist}(r_l, 0)} \\
&= \frac{\gamma \cdot \mathcal{OPT} \cdot \mathrm{dist}(r_l, 0)}{\mathrm{dist}(r_l, 0) \cdot \sum_{p\in P_{i+1}} \mathrm{dist}(p, 0)} \\
&= \frac{\gamma \cdot \mathcal{OPT}}{\sum_{p\in P_{i+1}} \mathrm{dist}(p, 0)}.
\end{aligned}$$

The latter implies

$$\begin{aligned}
\mathrm{cost}(P, C_{i+1}) - \mathrm{cost}(P, C_i) &\leq \sum_{p\in P_{i+1}} \mathrm{dist}(p, 0) \cdot \sin\alpha \\
&\leq \gamma \cdot \mathcal{OPT} \\
&\leq \gamma \cdot \mathrm{cost}(P, C_i)
\end{aligned}$$

which implies the lemma in Case 1.

Case 2: Points are on average much closer to the origin than to C_i.
Now we consider the case that

$$\sum_{p\in P_{i+1}} \mathrm{dist}(p, 0) < 4 \sum_{p\in P} \mathrm{dist}(p, C_i).$$

Let r_l be a point from $P_{i+1} \subseteq H_i^\perp$ that is in N_i and that has $\mathbf{Pr}[r_l] > 0$. Since $C_i \subseteq A$ and

$$\mathrm{dist}(r_l, C_i^*) = \mathrm{dist}(r_l, C_i) \leq 2 \cdot \mathrm{cost}(P, C_i) \cdot \mathbf{Pr}[r_l],$$

we know that $B(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp)$ intersects C_i^*. This implies that $\mathrm{proj}(r_l, C_i^*)$ lies also in $B(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp)$.

In fact,

$$2 \cdot \text{cost}(P, C_i) \cdot \mathbf{Pr}[r_l] \leq 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$$

implies that

$$B(\text{proj}(r_l, C_i^*), 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp) \subseteq B(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp).$$

Since $C_i^* \subseteq A \cap H_i^\perp$ we also have that there is a point

$$q \in C_i^* \cap B(\text{proj}(r_l, C_i^*), 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp)$$

with $\text{dist}(q, 0) \geq 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$.

Now consider the set which is the intersection of

$$\mathcal{N}(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp, \gamma/20)$$

with

$$B(\text{proj}(r_l, C_i), 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp),$$

which is a $(\gamma/10)$-net of

$$B(\text{proj}(r_l, C_i), 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp).$$

Hence, there is a point

$$q' \in \mathcal{N}(r_l, 10 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l], A \cap H_i^\perp, \gamma/20)$$

with $\text{dist}(q, q') \leq \frac{\gamma}{10} \cdot 5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l] \leq \gamma \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]$.

Let ℓ be the line through q and let ℓ' be the line through q'. Let $C_i^\perp$ denote the orthogonal complement of ℓ in C_i. Define the subspace C_{i+1} as the span of $C_i^\perp$ and ℓ'. Since q lies in C_i^* we have that $C_i^\perp$ contains H_i. Hence, C_{i+1} also contains H_i.

It remains to show that

$$\text{cost}(P, C_{i+1}) \leq (1 + \gamma) \cdot \text{cost}(P, C_i).$$

Now define L_i to be the orthogonal complement of $C_i^\perp$. Note that for any $p \in \mathbb{R}^d$ and its projection $p' = \text{proj}(p, L_i)$ we have $\text{dist}(p, C_i) = \text{dist}(p', C_i)$ and $\text{dist}(p, C_{i+1}) = \text{dist}(p', C_{i+1})$. Further observe that C_i corresponds to the line ℓ in L_i and C_{i+1} corresponds to a line $\ell'' = \text{proj}(\ell', L_i)$.

Define α to be the angle between ℓ and ℓ' and β the angle between ℓ and ℓ''. Note that $\alpha \geq \beta$. Then

$$\begin{aligned} \text{dist}(\text{proj}(p, C_i), C_{i+1}) &= \text{dist}(\text{proj}(\text{proj}(p, C_i), L_i), \ell'') \\ &= \text{dist}(\text{proj}(p, \ell), \ell''). \end{aligned}$$

This implies

$$\begin{aligned} \text{dist}(\text{proj}(p, \ell), \ell'') &= \text{dist}(\text{proj}(p, \ell), 0) \cdot \sin\beta \\ &\leq \text{dist}(p, 0) \cdot \sin\alpha. \end{aligned}$$

We have

$$\sin\alpha \leq \frac{\gamma \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]}{5 \cdot \mathcal{OPT} \cdot \mathbf{Pr}[r_l]} \leq \frac{\gamma}{5}.$$

Similar to the first case it follows that

$$\begin{aligned} \mathrm{cost}(P, C_{i+1}) - \mathrm{cost}(P, C_i) &\leq \sum_{p \in P_{i+1}} \mathrm{dist}(p, 0) \cdot \sin\alpha \\ &\leq \frac{\gamma}{5} \cdot \sum_{p \in P_{i+1}} \mathrm{dist}(p, 0). \end{aligned}$$

Since we are in Case 2 we have

$$\sum_{p \in P_{i+1}} \mathrm{dist}(p, 0) < 4 \cdot \mathrm{cost}(P, C_i),$$

which implies

$$\begin{aligned} \mathrm{cost}(P, C_{i+1}) - \mathrm{cost}(P, C_i) &\leq \frac{\gamma}{5} \cdot \sum_{p \in P_{i+1}} \mathrm{dist}(p, 0) \\ &\leq \gamma \cdot \mathrm{cost}(P, C_i). \end{aligned}$$

This concludes the proof of Lemma 61.

6.5 Insertion-only Streaming Algorithm

Here we assume that we have the coreset procedure of Lemma 65 below that given a subset $A \subseteq P$ of points and a j-subspace C generates a weighted point set B so that

$$\mathbf{Pr}[|\mathrm{cost}(A, C) - \mathrm{cost}(B, C| \leq \epsilon \cdot \mathrm{cost}(A, C)] \geq 1 - \delta.$$

We call this coreset procedure a *merge operator*. Let $b, m_0, m_1, \cdots, m_{\log_b n}$ denote parameters which will be determined later.

Recall that the merge and reduce method works as follows: For the offline variant of the merge and reduce method we assume that we are given a point set $P \subset \mathbb{R}^d$ of size n, where n is known in advance. We construct a b-ary tree T of depth $dep = \log_b n$. Let us assume level zero is the leaf level and level dep is the root of T.

Starting from level zero we have leaves $c_1^0, \cdots, c_{n/m_0}^0$ each has a bucket that can store points. We put the first m_0 points in c_1^0, the second m_0 points in c_2^0 and so on up to the n/m_0-th m_0 points that we store them in c_{n/m_0}^0. We then apply the merge operator on buckets (or leaves) $c_{(r-1)b+1}^0$ up to c_{rb}^0 for $1 \leq r \leq n/(m_0 b)$ and put new coreset points in c_r^1 of size m_1 at level 1. Recursively and at level i, for $1 \leq i \leq \log_b n$ we merge buckets (or children) $c_{(r-1)b+1}^{i-1}$ up to c_{rb}^{i-1} for $1 \leq r \leq n/(m_0 b^{i-1})$ (of node c_r^i) and put new coreset points in c_r^i of size m_i at level i.

Now the streaming variant of this algorithm is in this way. Assume the points are coming one by one and the result of the streaming algorithm should be correct coreset with probability $\geq 1-\delta$.

We put the first m_0 points in c_1^0, the second m_0 points in c_2^0 and so on up to the b-th chunk of m_0 points which we store in c_b^0. At this time and in level zero we have $m_0 \times b$ points in b full leaves and so we merge c_1^0 up to c_b^0 and put new points in c_1^1 of size m_1 at level 1 and deallocate the space assigned to $c_1^0, \cdots, c_b^0$ for new points. If we repeat this process for b times then we will have b full buckets at level 1 and so we must merge c_1^1 up to c_b^1 and put new points in c_1^2 of size m_2 at level 2 and deallocate the space assigned to $c_1^1, \cdots, c_b^1$ for new points.

As long as we see new points we do the same iteration, store them into b buckets at level 0 and then merge them into one of the b buckets at level 1, and deallocate the space assigned to the b buckets and so on. Therefore in each level i for $1 \leq i \leq \log_b n$ we have at most b buckets each one of size m_i and at each instance of time a union of all these buckets in all levels will be a snapshot of the stream up to that time.

If n is not known in advance, we start with some constant guess for n. If there are more points in the stream, compute a new value for n' such that the coreset size doubles. Use the streaming algorithm for the next n' points and keep the coreset for the first n points. Continue until no more points are coming. Since the coreset size doubles in each step, the space for the smaller coresets is at most the space for the largest one (by geometric progression). In order to make sure that everything works with good probability, one chooses confidence probability of $\frac{\delta}{bi^2}$ for the i-th step. Since the sum $\frac{1}{bi^2}$ is constant, we get an overall error of $O(\delta)$.

Now we explain a merge step in level i w.l.o.g. at c_1^i. We also set the parameters of the merge and reduce method. First of all we know that for the subtree rooted at c_1^i the original points are in buckets or leaves $c_1^0, \cdots, c_{b^i}^0$. For the simplicity we drop the index in c_1^i and show it with c^i.

Let $P(c^i) = c_1^0 \cup \cdots \cup c_{b^i}^0$. Assume that the optimal j-subspace to $P(c^i)$ is $\mathcal{OPT}(P(c^i))$. Here for the simplicity of the representation we replace $\text{proj}(p^-, C)$ by p^- where C is some j-subspace in $\mathbb{R}^d$. Therefore we can denote a weighted pair $(p, \text{proj}(p^-, C))$ by a triple (p, w_p, p^-) such that p has a weight of w_p and p^- has a weight of $-w_p$. For a set S of triples and a j-subspace C we define

$$\text{cost}(S, C) = \sum_{(p, w_p, p^-) \in S} w_p \cdot (\text{dist}(p, C) - \text{dist}(p^-, C)).$$

Our main theorem in this section is

Theorem 65 *Let P be a set of n points in $\mathbb{R}^d$, $j \geq 0$, and $\epsilon, \delta > 0$. In the read-only streaming model we can maintain a set S'' of triples and a set Q of positively weighted points using $\tilde{O}(d(j\epsilon^{-2} \cdot 2^{\sqrt{\log n}})^{\text{poly}(j)})$ weighted points such that, with probability at least $1-\delta$,*

$$\left|\text{cost}(P, C) - \text{cost}(S'', C) - \text{cost}(Q, C)\right| \leq \epsilon \cdot \text{cost}(P, C),$$

for every j-subspace in $\mathbb{R}^d$.

Proof. **[Overview of the proof of Theorem 65]**
One merge step is done as follows: Assume we have a set A of positive points and a set B of triples (p, w_p, p^-) all in $\mathbb{R}^d$. For A, we apply the algorithm of Lemma 52 to generate projected positive points E and new triples F where the points in E and the points with negative weights in F are on one $O(j^4)$-subspace. Now we need to reduce the size of sets B, E, F. For the triples in $B \cup F$ we use Lemma 67 (i.e., coreset of coreset) to sample few of them. For the points in E since E lies on a low-dimensional space we use the known low-dimensional coreset of Lemma 66 to reduce the size of E. □

Lemma 66 (Low-dimensional Coreset) *[FFS06] Given a point set P of size n in $O(j^4)$-subspace C^*, one can construct a strong coreset Q of size $O((\frac{j \cdot \log n}{\epsilon^2})^{j^2})$, where each point in Q has a positive weight, such that for any j-dimensional subspace $C \subseteq \mathbb{R}^d$ we have*

$$|\text{cost}(P, C) - \text{cost}(Q, C)| \leq \epsilon \cdot \text{cost}(P, C).$$

Lemma 67 (Coreset of Coreset) *Let C denote a fixed subspace of dimension j in $\mathbb{R}^d$. Let $S'(c^i)$ denote a set of triples for which we have $|S'(c^i)| > \frac{2^{(\tilde{O}(i \cdot j))^2}}{\epsilon^2} \log(2/\delta)$. Assume*

$$\sum_{(p, w_p, p^-) \in S'(c^i)} |w_p \cdot \text{dist}(p, p^-)| \leq 2^{\tilde{O}(i \cdot j)} \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))). \tag{6.13}$$

Then one can find a sample set $S''(c^i)$ of triples of size $s = \frac{2^{(\tilde{O}(i \cdot j))^2}}{\epsilon^2} \log(2/\delta)$ such that

$$|\text{cost}(S'(c^i), C) - \text{cost}(S''(c^i), C)| \leq \epsilon \cdot \text{cost}(P(c^i), C).$$

Proof. Let $U = \sum_{(p, w_p, p^-) \in S'(c^i)} w_p \cdot \text{dist}(p, p^-)$ and $V = \text{cost}(P(c^i), \mathcal{OPT}(P(c^i)))$. So by the assumption of the lemma $U \leq 2^{\tilde{O}(i \cdot j)} \cdot V$.

We use the framework of Lemma 24 for the problem of approximating a sum without computing all the summands that we introduced in Chapter 3.

Assume we number the triples in $S'(c^i)$ from 1 to $a = |S'(c^i)|$, i.e.,

$$S'(c^i) = \{(p_1, w_{p_1}, p_1^-), \cdots, (p_a, w_{p_a}, p_a^-)\}.$$

Put $Z = \text{cost}(S'(c^i), C)$, $Z_k = w_{p_k}(\text{dist}(p_k, C) - \text{dist}(p_k^-, C))$, $q_k = \frac{Z_k}{Z}$, and $r_k = \frac{w_{p_k} \cdot \text{dist}(p_k, p_k^-)}{U}$ for $1 \leq k \leq a$. We take a sample set $A = \{a_1, \cdots, a_\ell, \cdots, a_s\} \subseteq [a]$ of indexes according to the probabilities r_k and assign a weight of $w(a_\ell) = \frac{w_{p_{a_\ell}}}{s \cdot r_{a_\ell}}$ to a sampled triple $(p_{a_\ell}, w(a_\ell), p_{a_\ell}^-)$ for $1 \leq \ell \leq s$. Note that p_{a_ℓ} gets the weight of $w(a_\ell)$ and $p_{a_\ell}^-$ gets the weight of $-w(a_\ell)$.

Corresponding to the sampled triple $(p_{a_\ell}, w(a_\ell), p_{a_\ell}^-)$ we define a random variable

$$X_\ell = \frac{Z_{a_\ell}}{s \cdot r_{a_\ell}} = \frac{w_{p_{a_\ell}}}{s \cdot r_{a_\ell}} \cdot \left(\text{dist}(p_{a_\ell}, C) - \text{dist}(p_{a_\ell}^-, C)\right).$$

Let $X = \sum_{\ell=1}^{s} X_\ell$. Note that $\mathbf{E}[X_\ell] = \frac{Z}{s}$ and $\mathbf{E}[X] = Z$. By the triangle inequality we get

$$|\mathrm{dist}(p_{a_\ell}, C) - \mathrm{dist}(p_{a_\ell}^-, C)| \leq \mathrm{dist}(p_{a_\ell}, p_{a_\ell}^-).$$

So we can get an upper bound of

$$\begin{aligned}
|X_\ell| &\leq \frac{w_{p_{a_\ell}}}{s \cdot r_{a_\ell}} \cdot \left|\mathrm{dist}(p_{a_\ell}, C) - \mathrm{dist}(p_{a_\ell}^-, C)\right| \\
&\leq \frac{w_{p_{a_\ell}}}{s \cdot r_{a_\ell}} \cdot \mathrm{dist}(p_{a_\ell}, p_{a_\ell}^-) \\
&\leq \frac{w_{p_{a_\ell}}}{s \cdot \frac{w_{p_\ell}\mathrm{dist}(p_{a_\ell}, p_{a_\ell}^-)}{U}} \mathrm{dist}(p_{a_\ell}, p_{a_\ell}^-) \\
&\leq \frac{U}{s} \\
&\leq \frac{2^{\tilde{O}(i\cdot j)}V}{s},
\end{aligned}$$

for the random variable $|X_\ell|$.

Now similar to Lemma 24 we use the additive Hoeffding 4 with $M = 2^{\tilde{O}(i\cdot j)}V$, $t = \epsilon_i \cdot U$, and $m = s$ to show that for the sample set $S''(c^i) = \{(p_{a_\ell}, w(a_\ell), p_{a_\ell}^-) | a_\ell \in A\}$ of size $O(\frac{\log(2/\delta)}{\epsilon_i^2}) = \frac{2^{(\tilde{O}(i\cdot j))^2}}{\epsilon^2}\log(2/\delta)$ we get that with probability at least $1-\delta$ and for $\epsilon_i = \frac{\epsilon}{2^{\tilde{O}(i\cdot j)}}$

$$|\mathrm{cost}(S'(c^i), C) - \mathrm{cost}(S''(c^i), C)| \leq \epsilon_i \cdot U \leq \epsilon_i 2^{\tilde{O}(i\cdot j)} \cdot V \leq \epsilon \cdot \mathrm{cost}(P(c^i), C).$$

□

Proof. [**The complete proof of Theorem 65**]
Let C denote a fixed subspace of dimension j in $\mathbb{R}^d$.

First, we prove the theorem for $i = 1$. In order to merge the b buckets c_1^0 up to c_b^0 we run the algorithm of Lemma 52 on $P(c^1) = \cup_{r=1}^{b} c_r^0$. For C^* in Lemma 52 we use Lemma 8 in [DV07] which gives a $O(j^4)$-subspace with 2-approximation guarantee. We call this subspace $C^*(c^1)$. Lemma 52 returns $\mathrm{proj}(P(c^1), C^*(c^1))$ and $S''(c^1)$ which is a set of triples $(p, w_p, p^- = \mathrm{proj}(p^-, C^*(c^1)))$ of size $O(\log(1/\delta)\cdot\epsilon_1^{-2})$ such that with probability at least $1-\delta$ we get

$$\begin{aligned}
|\mathrm{cost}(P(c^1), C) - \mathrm{cost}(S''(c^1), C) - \mathrm{cost}(\mathrm{proj}(P(c^1), C^*(c^1)), C)| &\leq \epsilon_1 \cdot \mathrm{cost}(P(c^1), C^*(c^1)) \\
&\leq 2\epsilon_1 \cdot \mathrm{cost}(P(c^1), C).
\end{aligned}$$

Then we run the algorithm of Theorem 66 on $\mathrm{proj}(P(c^1), C^*(c^1))$ to obtain a strong coreset $Q(c^1)$ of size $O((\frac{j\cdot\log n}{\epsilon_1})^{j^2})$ such that we have

$$\begin{aligned}
|\mathrm{cost}(\mathrm{proj}(P(c^1), C^*(c^1)), C) - \mathrm{cost}(Q(c^1), C)| &\leq \epsilon_1 \cdot \mathrm{cost}(\mathrm{proj}(P(c^1), C^*(c^1)), C) \\
&\leq 3\epsilon_1 \cdot \mathrm{cost}(P(c^1), C),
\end{aligned}$$

where the last inequality is because

$$\text{cost}(\text{proj}(P(c^1), C^*(c^1)), C) \leq \text{cost}(P(c^1), C^*(c^1))) + \text{cost}(P(c^1), C) \leq 3 \cdot \text{cost}(P(c^1), C). \quad (6.14)$$

Applying the triangle inequality with probability at least $1 - \delta$ we get

$$|\text{cost}(P(c^1), C) - \text{cost}(S''(c^1), C) - \text{cost}(Q(c^1), C)| \leq 5\epsilon_1 \cdot \text{cost}(P(c^1), C).$$

From Equation 6.14 we get

$$\text{cost}(Q(c^1), C) \leq 3(1 + \epsilon) \cdot \text{cost}(P(c^1), C).$$

Since $Q(c^1)$ is a coreset for an arbitrary j-subspace C it also applies to $\mathcal{OPT}(P(c^1))$ the optimal j-subspace of $P(c^1)$ for which we get

$$\text{cost}(Q(c^1), \mathcal{OPT}(P(c^1))) \leq 3(1 + \epsilon) \cdot \text{cost}(P(c^1), \mathcal{OPT}(P(c^1))).$$

On the other hand, for the optimal j-subspace of $Q(c^1)$, i.e., $\mathcal{OPT}(Q(c^1))$ we have

$$\text{cost}(Q(c^1), \mathcal{OPT}(Q(c^1))) \leq \text{cost}(Q(c^1), \mathcal{OPT}(P(c^1))) \leq 3(1 + \epsilon) \cdot \text{cost}(P(c^1), \mathcal{OPT}(P(c^1))).$$

Next, we prove the theorem for $i > 1$.

For the merge step assume that the buckets c_r^{i-1} for $1 \leq r \leq b$ consists of two sets $Q(c_r^{i-1})$ and $S''(c_r^{i-1})$ where $Q(c_r^{i-1})$ is a positively weighted point set and $S''(c_r^{i-1})$ is a set of triples. Also assume that

$$\text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), C) \leq (3(1 + \epsilon))^{i-1} \cdot \text{cost}(\cup_{r=1}^{b} P(c_r^{i-1}), C) = (3(1 + \epsilon))^{i-1} \cdot \text{cost}(P(c^i), C)$$

as for the induction hypothesis.

Once again, since $\cup_{r=1}^{b} Q(c_r^{i-1})$ is a coreset for an arbitrary j-subspace C it also applies to $\mathcal{OPT}(\cup_{r=1}^{b} P(c_r^{i-1})) = \mathcal{OPT}(P(c^i))$ the optimal j-subspace of $P(c^i)$ for which we get

$$\text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), \mathcal{OPT}(P(c^i))) \leq (3(1 + \epsilon))^{i-1} \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))).$$

On the other hand, for the optimal j-subspace of $\cup_{r=1}^{b} Q(c_r^{i-1})$, i.e., $\mathcal{OPT}(\cup_{r=1}^{b} Q(c_r^{i-1}))$ we have

$$\begin{aligned}\text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), \mathcal{OPT}(\cup_{r=1}^{b} Q(c_r^{i-1}))) &\leq \text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), \mathcal{OPT}(P(c^i))) \\ &\leq (3(1 + \epsilon))^{i-1} \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))).\end{aligned}$$

We run the algorithm of Lemma 52 on $\cup_{r=1}^{b} Q(c_r^{i-1})$ and for C^* in Lemma 52 again we use Lemma 8 in [DV07] which gives a $O(j^4)$-subspace with 2-approximation guarantee of the optimal j-subspace of $\cup_{r=1}^{b} Q(c_r^{i-1})$, i.e., $\mathcal{OPT}(\cup_{r=1}^{b} Q(c_r^{i-1}))$. We call this subspace $C^*(c^i)$. Lemma 52 returns $\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i))$ and $S(c^i)$ which is a set of triples $(p, w_p, p^- = \text{proj}(p^-, C^*(c^i)))$ of size $O(\log(1/\delta) \cdot \epsilon_i^{-2})$ for which we have

$$\begin{aligned}
&\text{cost}(\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)), C) \\
&\leq \text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)) + \text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), C) \\
&\leq 2(3(1+\epsilon))^{i-1} \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))) + (3(1+\epsilon))^{i-1} \cdot \text{cost}(P(c^i), C) \\
&\leq 3^i(1+\epsilon)^{i-1} \cdot \text{cost}(P(c^i), C).
\end{aligned}$$

Therefore with probability at least $1-\delta$ we get

$$\begin{aligned}
&|\text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), C) - \text{cost}(S(c^i), C) - \text{cost}(\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)), C)| \\
&\leq \epsilon_i \cdot \text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)) \\
&\leq 2\epsilon_i \cdot \text{cost}(\cup_{r=1}^{b} Q(c_r^{i-1}), \mathcal{OPT}(\cup_{r=1}^{b} Q(c_r^{i-1}))) \\
&\leq 2\epsilon_i \cdot (3(1+\epsilon))^{i-1} \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))) \\
&\leq \epsilon_i \cdot O(2^{\tilde{O}(i)}) \cdot \text{cost}(P(c^i), C).
\end{aligned}$$

Then we run the algorithm of the theorem 66 on $\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i))$ to obtain a strong coreset $Q(c^i)$ of size $O((\frac{j \cdot \log n}{\epsilon_i})^{j^2})$ such that

$$\begin{aligned}
&|\text{cost}(\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)), C) - \text{cost}(Q(c^i), C)| \\
&\leq \epsilon_i \cdot \text{cost}(\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)), C) \\
&\leq \epsilon_i \cdot 3^i(1+\epsilon)^{i-1} \cdot \text{cost}(P(c^i), C) \\
&\leq \epsilon_i \cdot O(2^{\tilde{O}(i)}) \cdot \text{cost}(P(c^i), C).
\end{aligned}$$

As for the next step of the induction we also obtain a bound on as follows $\text{cost}(Q(c^i), C)$ assuming $\epsilon_i \leq \epsilon$

$$\begin{aligned}
\text{cost}(Q(c^i), C) &\leq \text{cost}(\text{proj}(\cup_{r=1}^{b} Q(c_r^{i-1}), C^*(c^i)), C) + \epsilon_i \cdot O(2^{\tilde{O}(i)}) \cdot \text{cost}(P(c^i), C) \\
&\leq 3^i(1+\epsilon)^{i-1} \cdot \text{cost}(P(c^i), C) + \epsilon_i \cdot 3^i(1+\epsilon)^{i-1} \cdot \text{cost}(P(c^i), C) \\
&\leq 3^i(1+\epsilon)^{i-1}(1+\epsilon_i) \cdot \text{cost}(P(c^i), C) \\
&\leq (3(1+\epsilon))^i \cdot \text{cost}(P(c^i), C).
\end{aligned}$$

Therefore we can apply the whole machinary for the next step of the induction.

Applying the triangle inequality with probaility at least $1-\delta$, we get

$$|\text{cost}(\cup_{m=1}^{b} Q(c_m^{i-1}), C) - \text{cost}(S(c^i), C) - \text{cost}(Q(c^i), C)| \leq \epsilon_i \cdot O(2^{\tilde{O}(i)}) \cdot \text{cost}(P(c^i), C).$$

Let $S'(c^i)$ denote the union of $\cup_{r=1}^{b} S''(c_r^{i-1})$ and $S(c^i)$ which is in fact a union set of old triples and new triples. Here also the same as in the above one can inductively prove that

$$\begin{aligned}
\text{cost}(S'(c^i), C) &= \sum_{(p, w_p, p^-) \in S'(c^i)} w_p \cdot (\text{dist}(p, C) - \text{dist}(p^-, C)) \leq \sum_{(p, w_p, p^-) \in S'(c^i)} w_p \cdot (\text{dist}(p, p^-)) \\
&\leq O(2^{\tilde{O}(i)}) \cdot \text{cost}(P(c^i), \mathcal{OPT}(P(c^i))).
\end{aligned}$$

We put $\epsilon_i = \frac{\epsilon}{O(2^{\tilde{O}(i \cdot j)})}$ and use Lemma 67 for $S'(c^i)$ to sample a set $S''(c^i)$ of triples of size $t = O(\frac{2^{(\tilde{O}(i \cdot j))^2}}{\epsilon^2} \cdot \log(2/\delta))$.

Thus the cost of two sets $Q(c^i)$ and $S''(c^i)$ to an arbitrary j-subspace C is an unbiased estimator for $\text{cost}(P(c^i), C)$. We should note that during $\log_b n$ level of tree T we generate new points, but overall number of generated points is at most $\log_b^{\log_b n}(n) \cdot n$. Now let Z of size $|Z| = O((\log_b^{\log_b n} n \cdot n)^{10dj})$ denote all possible j-subspaces in $\mathbb{R}^d$ generated by Algorithm NET, in Section 6.3. We replace δ by $\delta'/|Z|$ in sample size t to show that a strong coreset can be maintained in data stream.

At $c^{\log_b n}$, we set $Q = Q(c^{\log_b n})$ and $S'' = S''(c^{\log_b n})$ to get the sets claimed in the theorem.

Now we explain the space complexity of the algorithm. Put $b = 2^{\frac{j^2}{\gamma}}$. In this way, the depth of the b-ary tree T is $\log_{2^{j^2/\gamma}} n = \gamma/j^2 \cdot \log_2 n$. We put $m_0 = \tilde{O}((j \cdot \log n/\epsilon^2)^{j^2})$ and $m_i = \tilde{O}((\frac{j \cdot \log n}{\epsilon_i^2})^{j^2} \log(2/\delta_n))$ for $1 \leq i \leq dep = \lceil \log_b n \rceil$ where $\delta_n = \delta/(\log_b^{\log_b n}(n) \cdot n)^{10dj}$ is the confidence parameter for the n points received so far, i is the level in which bucket size must be m_i, $\epsilon_i = \frac{\epsilon}{O(2^{\tilde{O}(i \cdot j)})}$ is the error parameter at level i.

We can have at most b buckets in each level for at most $\gamma/j^2 \cdot \log_2 n$ levels therefore the space complexity of our streaming algorithm will be $\tilde{O}(d \cdot \gamma/j^2 \cdot \log_2 n \cdot 2^{\frac{j^2}{\gamma}} \cdot (\frac{j \cdot n^\gamma}{\epsilon^2})^{j^2})$ weighted points. We set $\gamma = \frac{1}{j^3 \cdot \sqrt{\log n}}$ to have the space complexity of $\tilde{O}(d \cdot 2^{O(j^5 \cdot \sqrt{\log n})}(\frac{j \cdot n^\gamma}{\epsilon^2})^{j^2})) = \tilde{O}(d \cdot 2^{O(j^5 \cdot \sqrt{\log n})}(j/\epsilon^2 \cdot 2^{\frac{\log n}{j^4 \cdot \sqrt{\log n}}})^{j^2}) = \tilde{O}(d \cdot (j/\epsilon^2)^{j^2} \cdot 2^{O(j^5 \cdot \sqrt{\log n})})$ weighted points. □

Chapter 7

$O(\log n)$-Pass L_p-Sampler

In this chapter we give the details of the $O(\log n)$-Pass L_p-Sampler that we introduced in Section 3.3. The following is our main theorem.

Theorem 68 *Let C be an arbitrary constant. Let $0 < \epsilon \leq \frac{1}{12}$. Let $\delta = O(n^{-C})$ and $\eta = \Theta(\frac{\epsilon}{\log n})$. For any $p \in [0, 2]$, there is an $O(\log n)$-pass L_p-sampler that uses $O(\epsilon^{-2} \log^4 n)$ bits of space. The probability of outputting* FAIL *is less than n^{-C} for any constant $C > 0$.*

The Algorithm. Our algorithm is given below.

$O(\log n)$-Pass-L_p-Sampler (Stream $\mathcal{S}, \epsilon$)

(1) Initialize $a = [n]$, $\delta = O(n^{-C})$, $\eta = \Theta(\frac{\epsilon}{\log n})$, and $\beta = 1$.

(2) In the first pass, compute $\tilde{F}_p(a) = F_p$-Estimation$(\mathcal{S}, a, 2\epsilon, \delta)$.

(3) If $\tilde{F}_p(a) = 0$, Output FAIL.

(4) For $j = 1, 2, \ldots, \log_2 n$, in $(j+1)$-st pass do:

  (a) Let a_L be the first $|a|/2$ items of a, and let $a_U = a \setminus a_L$.

  (b) Let $\mathcal{S}_L$ and $\mathcal{S}_U$ be the interleaved streams consisting of the subsequence of updates to a_L and a_U respectively.

  (c) $L \leftarrow F_p$-Estimation$(\mathcal{S}_L, a_L, \eta, \delta)$,
  $U \leftarrow F_p$-Estimation$(\mathcal{S}_U, a_U, \eta, \delta)$

  (d) If $L = U = 0$, output FAIL.

  (e) With probability $\frac{L}{L+U}$, assign $a \leftarrow a_L$, $\mathcal{S} \leftarrow \mathcal{S}_L$, and $\beta \leftarrow \beta \cdot \frac{L}{L+U}$, else assign $a \leftarrow a_U$, $\mathcal{S} \leftarrow \mathcal{S}_U$ and $\beta \leftarrow \beta \cdot \frac{U}{L+U}$.

(5) Let i be such that $a = \{i\}$. Compute a_i in an extra pass. Let $q = (1 + 2\epsilon)|a_i|^p / \tilde{F}_p(a)$.

(6) If $|\beta - q| > 12\epsilon \cdot q$, then output FAIL otherwise output (i, a_i).

The Proof. In this algorithm, the routine F_p-Estimation $\tilde{F}_p(a) = F_p$-Estimation$(\mathcal{S}, a, \epsilon, \delta)$ is the optimal-space algorithm for estimating the F_p-value of a vector due to Kane *et al* [KNW10] in which the stream $\mathcal{S}$ corresponds to the vector a with error parameter ϵ and failure probability δ, i.e, it returns a value $\tilde{F}_p(a)$ for which

$$\mathbf{Pr}[|\tilde{F}_p(a) - F_p(a)| \leq \epsilon \cdot F_p(a)] \geq 1 - \delta,$$

where $F_p(a)$ is the p-th frequency moment of a.

Let SuccessfulSubroutines be the event that none of the invocations of F_p-Estimation in the above algorithm is failed in Steps 3 and 4d.

Assume that SuccessfulSubroutines occurs. Let GoodNormEstimation be the event that $F_p(a)$ is well-approximated by $\tilde{F}_p(a)$, i.e,

$$(1 - 2\epsilon)F_p(a) \leq \tilde{F}_p(a) \leq (1 + 2\epsilon)F_p(a).$$

By our choice of δ and a union bound, with probability at least $1 - n^{-C}$, the events SuccessfulSubroutines and GoodNormEstimation occur.

Lemma 69 *Suppose event SuccessfulSubroutines occurs. Let* $0 < \epsilon \leq 1/9$ *and* $\eta = \Theta(\frac{\epsilon}{\log n})$. *Then for all* i*, the probability that coordinate* i *is chosen in Step 5 is* $(1 \pm 4\epsilon)|a_i|^p/F_p(a)$*, i.e,*

$$(1 - 4\epsilon)|a_i|^p/F_p(a) \leq \beta = \mathbf{Pr}[i \textit{ chosen in Step 5 }] \leq (1 + 4\epsilon)|a_i|^p/F_p(a).$$

Proof. Fix an $i \in [n]$. Then there is a unique sequence of assignments

$$a^0 = [n], a^1, a^2, \ldots, a^{\log_2 n} = \{i\}$$

to the loop variable a, for which $a = a^j$ after iteration j of Step 4, that cause coordinate i to be chosen in Step 5. For $j \in \{0, 1, \ldots, \log_2 n\}$, let $\mathcal{E}_j$ be the event that $a = a^j$ after iteration j of Step 4. Then,

$$\begin{aligned}\mathbf{Pr}[i \text{ chosen in step 4}] &= \cap_{j=0}^{\log_2 n} \mathbf{Pr}[\mathcal{E}_j \mid \mathcal{E}_1, \ldots, \mathcal{E}_{j-1}] \\ &= \cap_{j=1}^{\log_2 n} \mathbf{Pr}[\mathcal{E}_j \mid \mathcal{E}_{j-1}].\end{aligned}$$

For any j assuming that $\eta \leq 1/2$, we get

$$\begin{aligned}\mathbf{Pr}[\mathcal{E}_j \mid \mathcal{E}_{j-1}] &\leq \frac{(1+\eta)F_p(a^j)}{(1-\eta)F_p(a_L^{j-1}) + (1-\eta)F_p(a_U^{j-1})} \\ &= \frac{(1+\eta)}{(1-\eta)} \cdot \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1 + \frac{2\eta}{(1-\eta)}) \cdot \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1 + 3\eta) \cdot \frac{F_p(a^j)}{F_p(a^{j-1})},\end{aligned}$$

and

$$\begin{aligned}\mathbf{Pr}[\mathcal{E}_j \mid \mathcal{E}_{j-1}] &\geq \frac{(1-\eta)F_p(a^j)}{(1+\eta)F_p(a_L^{j-1}) + (1+\eta)F_p(a_U^{j-1})} \\ &= \frac{(1-\eta)}{(1+\eta)} \cdot \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1 - \frac{2\eta}{(1+\eta)}) \cdot \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1-3\eta) \cdot \frac{F_p(a^j)}{F_p(a^{j-1})}.\end{aligned}$$

Hence,

$$\begin{aligned}\mathbf{Pr}[i \text{ chosen in Step 5}] &\leq (1+3\eta)^{\log_2 n} \prod_{j=1}^{\log_2 n} \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1+3\eta)^{\log_2 n} \frac{|a_i|^p}{F_p(a)},\end{aligned}$$

and

$$\begin{aligned}\mathbf{Pr}[i \text{ chosen in Step 5}] &\leq (1-3\eta)^{\log_2 n} \prod_{j=1}^{\log_2 n} \frac{F_p(a^j)}{F_p(a^{j-1})} \\ &= (1-3\eta)^{\log_2 n} \frac{|a_i|^p}{F_p(a)}.\end{aligned}$$

Using the well known fact that $(1+t) \leq e^t$ for all $t \in \mathbb{R}$, we have

$$(1+3\eta)^{\log_2 n} \leq e^{3\eta \log_2 n} \leq e^{3\epsilon} \leq 1 + 4\epsilon,$$

where the last inequality follows by a Taylor expansion, i.e,

$$e^{3\epsilon} = 1 + \frac{3\epsilon}{1!} + \frac{(3\epsilon)^2}{2!} + \frac{(3\epsilon)^3}{3!} + \cdots \leq 1 + 3\epsilon + \frac{\epsilon}{2} + \frac{\epsilon}{2} \leq 1 + 4\epsilon,$$

for $\epsilon \leq 1/9$.

For the other direction, we appeal to Proposition B.3, part 2, of [MR95], which states that for all $t, n \in \mathbb{R}$ such that $r \geq 1$ and $|t| \leq r$,

$$e^t(1 - t^2/r) \leq (1 + t/r)^r.$$

Then

$$(1-3\eta)^{\log_2 n} = (1 - (3\eta \log_2 n)/\log_2 n)^{\log_2 n}.$$

Thus, setting $t = -3\eta \log_2 n$, and $r = \log_2 n$, applying this inequality for $\epsilon \leq 1/9$ and $\eta = \frac{\epsilon}{\log n}$ we have,

$$\begin{aligned}(1-3\eta)^{\log_2 n} &\geq e^{-3\eta \log_2 n}(1 - (-3\eta \log_2 n)^2/(\log_2 n)) \\ &\geq (1-3\epsilon)(1-9\epsilon^2) = 1 - 3\epsilon - 9\epsilon^2 + 27\epsilon^3 \geq 1 - 4\epsilon.\end{aligned}$$

Thus,

$$(1-4\epsilon)|a_i|^p/F_p(a) \leq \beta = \mathbf{Pr}[\text{i chosen in Step 5 }] \leq (1+4\epsilon)|a_i|^p/F_p(a).$$

□

Proof. **[of Theorem (68)]** Let C be an arbitrary constant that we have it in Theorem 68. Recall that $0 < \epsilon \leq \frac{1}{12}$. Event SuccessfulSubroutines occurs with probability at least $1-1/(n^C)$. In this case, by Lemma 69, $O(\log n)$-Pass-L_p-Sampler never outputs FAIL in Step 4d, and for each $i \in [n]$, the probability that coordinate i is chosen in Step 5 is $(1 \pm 4\epsilon)|a_i|^p/F_p(a)$.

Event GoodNormEstimation occurs with probability at least $1 - n^{-C}$. We condition on both SuccessfulSubroutines and GoodNormEstimation occurring, which, by a union bound and our choice of $\delta = O(n^{-C})$, happens with probability at least $1 - n^{-C}$.

Since GoodNormEstimation occurs and $F_p(a) > 0$, the algorithm does not output FAIL in Step 3 and

$$(1-2\epsilon)F_p(a) \leq \tilde{F}_p(a) \leq (1+2\epsilon)F_p(a).$$

Notice that $q = \frac{(1+2\epsilon)|a_i|^p}{\tilde{F}_p(a)}$, and we have that

$$\frac{(1+2\epsilon)|a_i|^p}{(1-2\epsilon)F_p(a)} \geq q \geq \frac{(1+2\epsilon)|a_i|^p}{(1+2\epsilon)F_p(a)}$$

which is

$$(1+8\epsilon)\frac{|a_i|^p}{F_p(a)} = \frac{(1-2\epsilon)(1+8\epsilon)}{(1-2\epsilon)}\frac{|a_i|^p}{F_p(a)} \geq q \geq \frac{|a_i|^p}{F_p(a)}.$$

Hence

$$|\beta - q| \leq (4\epsilon + 8\epsilon)\frac{|a_i|^p}{F_p(a)} \leq 12\epsilon \cdot q.$$

So the algorithm does not output FAIL in step 6.

For the efficiency, the number of passes is always $O(\log n)$. The space used is dominated by that of F_p-Estimation on a set of size at most n with the failure probability parameter $\delta = O(n^{-C})$ and the relative error parameter $\eta = \Theta(\frac{\epsilon}{\log n})$ for C which is an arbitrary constant that we have it in Theorem 68. The space is $O(\epsilon^{-2}\log^4 n)$ since the space of F_p-Estimation is $O(\eta^{-2}\log n \cdot \log(1/\delta))$ bits. □

Chapter 8

1-Pass L_p-Sampler

Our goal in this chapter is to reduce the number of passes of the L_p-sampler from $O(\log n)$ pass in the previous chapter to only one pass. The following is our main theorem.

Theorem 70 *Let $0 < \epsilon \leq 1$ and let $C > 0$ be an arbitrarily large constant. For any $p \in [0, 2]$, there is a 1-pass L_p-sampler that uses $\text{poly}(\epsilon^{-1} \log n)$ bits of space. Ignoring an initial data structure initialization stage (which doesn't require looking at the stream), the update time of the L_p-sampler is $\text{poly}(\epsilon^{-1} \log n)$. There is also an n^{-C} probability of failure of the algorithm, in which case it can output anything.*

We first describe the solution for $p \in (0, 2]$ which is overviewed in Section 3.4. At the end of this chapter in Section 8.2 we describe how to sample when $p = 0$, which is in fact simpler.

8.1 The Proof of Theorem 70 for $p \in (0, 2]$

Throughout we use an algorithm that we call HeavyHitters, which is given by the following theorem. The algorithm has its roots in the CountSketch algorithm of Charikar *et al* [CCFC02], but that algorithm did not have fast reporting time for streams in the turnstile model. This property was later achieved by Ganguly *et al* [GSS08], building upon work of Cormode and Muthukrishnan [CM05a].

Theorem 71 ([CCFC02, CM05a, GSS08]) *Let $0 < \delta < 1$. Let a be a vector of length n initialized to zero. Let $\mathcal{S}$ be a stream of m updates (i, x) to a, where $i \in [n]$ and $x \in \{-M, \cdots, +M\}$. There is an algorithm* HeavyHitters$(\mathcal{S}, B, \delta, \epsilon)$ *that, with probability at least $1 - \delta$, returns all coordinates i (plus potentially some more coordinates) for which $|a_i|^2 \geq \frac{F_2(a)}{B}$, together with an approximation $\tilde{a}_i$ such that $|a_i| \leq |\tilde{a}_i| \leq (1 + \epsilon)|a_i|$ and sign$(\tilde{a}_i) =$ sign(a_i). Here B is an input parameter. The space complexity of* HeavyHitters *is $B \cdot \text{poly}(\epsilon^{-1} \log n) \log 1/\delta$. The update time is $\text{poly}(\epsilon^{-1} \log n)$, and the reporting time is $\text{poly}(B\epsilon^{-1} \log n)$.*

Corollary 72 *For any $p \in (0,2]$, with probability at least $1-\delta$, the output of* HeavyHitters *also contains all items i for which $|a_i|^p \geq \frac{F_p(a)}{B^{p/2}}$.*

Proof. For $p \in (0,2]$, if $|a_i|^p \geq \frac{F_p(a)}{B^{p/2}}$, then $|a_i|^2 \geq \frac{(F_p(a))^{2/p}}{B}$, and, by monotonicity of norms (see, e.g., [Fri04]), $(F_p(a))^{2/p} \geq F_2(a)$. Hence, $|a_i|^2 \geq \frac{F_2(a)}{B}$, as needed in order to be found by HeavyHitters. □

We start with the following definition of level sets, modified from that of [IW05, JW09]. We should mention that another definition of level sets was also developed by Frahling and Sohler in [FS05] for the problem of finding coresets for k-median and k-means problem in dynamic geometric data streams which is a variant of the turnstile model.

Fix an $\eta = 1 + \Theta(\epsilon)$. Let C' be a sufficiently large constant. We shall make the simplifying assumption that all values $|a_i|$, if non-zero, are integers of absolute value at least $\tau = C'\epsilon^{-1}$. This is w.l.o.g., since for each update (i,x) in the input stream $\mathcal{S}$, we can replace it with update $(i,\tau x)$. This will also change F_p by a factor of τ^p, but will not affect the distribution we are trying to sample from. In the continuation we drop the vector (a) from $F_p(a)$ and show it with F_p whenever it is clear from context.

Definition 73 *Let $B \geq 1$ be a parameter. For $1 \leq t \leq \Theta(\epsilon^{-1}\log\epsilon^{-1})$, define $S_t = \{i \in [n] : |a_i| = t\}$, and for $\Omega(\epsilon^{-1}\log\epsilon^{-1}) \leq t \leq C_\eta \log n$, for some constant C_η that depends on η, define*

$$S_t = \{i \in [n] : |a_i| \in [\eta^{t-1}, \eta^t)\}.$$

Call a level t $(1/B)$-contributing if $|S_t| \cdot \eta^{pt} \geq \frac{F_p}{B}$. For a $(1/B)$-contributing level t, coordinates in S_t will also be called $(1/B)$-contributing coordinates.

In the description of our algorithms, we assume that we have a value F_p' with $F_p \leq F_p' \leq (1+\Theta(\epsilon))F_p$. This is w.l.o.g., since the algorithm can guess each value in a set of $O(\epsilon^{-1}\log n)$ possible values, and use this as the estimate F_p'. In parallel, the algorithm runs the $O(\epsilon^{-2}\log n)$ space algorithm of [KNW10] to obtain a $(1+\Theta(\epsilon))$-approximation to F_p, and after processing the stream knows which of its guesses is correct. This only changes the space by a $\mathrm{poly}(\epsilon^{-1}\log n)$ factor.

We can assume that $\|a\|_p^p \geq \epsilon^{-2}$. Notice that if this were not the case, i.e., we had $\|a\|_p^p < \epsilon^{-2}$, then since the entries of a are integers, the number $\|a\|_0$ of non-zero coordinates is at most ϵ^{-2}. In this case it is easy to obtain all non-zero coordinates i together with the exact values a_i in $\mathrm{poly}(\epsilon^{-1}\log n)$ bits of space using our L_0-sampler in Section 8.2.

Indeed, if we run the L_0-sampler $O(\epsilon^{-2}\log n)$ times, the probability we obtain every non-zero coordinate i together with its value a_i is $\geq 1 - n^{-C}$. As the space complexity of the L_0-sampler is $\mathrm{poly}(\epsilon^{-1}\log n)$, our resulting space is also $\mathrm{poly}(\epsilon^{-1}\log n)$. Given all the non-zero items together with their values a_i, it is then trivial to do L_p-sampling.

Put $T = C_\eta \log n$. We will perform the following transformation to the stream $\mathcal{S}$, creating a new input stream $\mathcal{S}'$. Say a $t \in [T]$ is *growing* if

$$\eta^{pt} \leq \frac{\epsilon^4 F'_p}{5T^3 \log^2 n}.$$

StreamTransformation $(\mathcal{S}, p)$

- $\mathcal{S}' \leftarrow \mathcal{S}$
- For each $t \in [T]$ for which t is growing:
  (1) Allocate $\left\lceil \frac{\epsilon F'_p}{5T\eta^{pt}} \right\rceil$ new coordinates.
  (2) For each new coordinate j, prepend the pair $(j, \lfloor \eta^{t-1/2} \rfloor)$ to the stream $\mathcal{S}'$.
- Output the stream $\mathcal{S}'$.

Let a' denote the underlying vector of $\mathcal{S}'$, which is of length $\mathrm{poly}(n)$. We refer to each new coordinate added by StreamTransformation as an *injected coordinate* (that is, a coordinate that appears in a' but not in a is an injected coordinate). Notice that $\lfloor \eta^{t-1/2} \rfloor = \lfloor \eta^{t-1} \cdot \eta^{1/2} \rfloor$.

Lemma 74 *For growing t, the number of injected coordinates added is at least $\epsilon^{-3} T^2 \log^2 n$.*

Proof. For growing t, the number of coordinates added is

$$\left\lceil \frac{\epsilon F'_p}{5T\eta^{pt}} \right\rceil \geq \frac{\epsilon F'_p}{5T\epsilon^4 F'_p/(5T^3 \log^2 n)} = \epsilon^{-3} T^2 \log^2 n.$$

□

Lemma 75 $F_p(a) \leq F_p(a') \leq (1 + \epsilon/2) F_p(a)$*, for ϵ less than a sufficiently small constant.*

Proof. The left inequality is obvious. For the right inequality, fix a growing t. Then the added contribution of this level set to F_p is at most

$$\lceil \epsilon F'_p/(5T\eta^{pt}) \rceil \cdot \eta^{p(t-1/2)}.$$

By Lemma 74, $\lceil \epsilon F'_p/(5T\eta^{pt}) \rceil \geq \epsilon^{-3} T^2 \log^2 n$, which in particular implies $\epsilon F'_p/(5T\eta^{pt}) \geq 1$. Hence,

$$\lceil \epsilon F'_p/(5T\eta^{pt}) \rceil \cdot \eta^{p(t-1/2)} \leq 2\epsilon \eta^{pt-p/2} F'_p/(5T\eta^{pt}) \leq 2\epsilon F'_p/(5T)$$

Hence, the added contribution is at most

$$2\epsilon F'_p/(5T) \leq \epsilon F_p(a)/(2T).$$

The lemma follows by summing over all t. □

Lemma 76 *With respect to a', each growing t is $\epsilon/(40T)$-contributing.*

Proof. Each growing t contributes at least $\lfloor \eta^{t-1/2} \rfloor^p \cdot \frac{\epsilon F'_p}{5T\eta^{pt}} \geq \frac{\eta^{pt-p/2}}{2^p} \cdot \frac{\epsilon F'_p}{5T\eta^{pt}}$ to $F_p(a')$, which is at least

$$\frac{\eta^{-p/2}\epsilon F'_p}{5T2^p} \geq \frac{\eta^{-p/2}\epsilon F_p(a)}{5T2^p} \geq \frac{\eta^{-p/2}\epsilon F_p(a')}{5T2^p(1+\frac{\epsilon}{2})} \geq \frac{\epsilon F_p(a')}{40T},$$

where the second to last inequality follows from Lemma 75. □

Recall that Let a' denote the underlying vector of $\mathcal{S}'$, which is of length $\mathrm{poly}(n)$. Note that $\mathcal{S}'$ is the output of StreamTransformation. We start with the following assumption: for all i, if $i \in S_t$, then $|a'_i| \leq \eta^t/(1+\beta\epsilon)$ for a small constant $\beta > 0$. Intuitively, this is to ensure that the $|a'_i|$ are away from their upper boundaries, so that when HeavyHitters returns $(1+\Theta(\epsilon))$-approximations to the $|a'_i|$, which are guaranteed to be at least $|a'_i|$, the i can be classified correctly into the corresponding S_t. We note that for an injected coordinate j with value $\lfloor \eta^{t-1/2} \rfloor$ for some t, we have $\lfloor \eta^{t-1/2} \rfloor \leq \eta^{t-1/2} = \frac{\eta^t}{1+\Theta(\epsilon)}$, so this assumption already holds for the injected coordinates. Later we show that this assumption is unnecessary for all coordinates.

We can consider the operation on stream $\mathcal{S}'$ as opposed to $\mathcal{S}$, since StreamTransformation can be implemented in $\mathrm{poly}(\epsilon^{-1}\log n)$ space in a preprocessing phase, i.e., without looking at the pairs in $\mathcal{S}$. The following is our main sampling algorithm.

L_p-Sampler($\mathcal{S}'$, p)

(1) $L \leftarrow$ HeavyHitters($\mathcal{S}', A, n^{-C}, \epsilon/C$), where $A = (5T^3\epsilon^{-4}\log^2 n)^{2/p}$. Let the $(\tilde{a'})_i$ be estimates of the a'_i for all $i \in L$.

(2) $B = (6400 \cdot 1600\epsilon^{-3}\eta^p T^3 \log^3 n)^{2/p}$.

(3) Independently, for $z = 1, 2, \ldots, C\log n$,

  (a) Put $\{L_j^z \mid \text{all } j\} =$ ListHH($\mathcal{S}', B$).

  (b) Put $\{\tilde{s}_t^z \mid \text{growing } t\} =$Set-Estimation($\{L_j^z \mid \text{all } j\}$).

(4) For growing t, put $\tilde{s}_t = \mathrm{median}_z\, \tilde{s}_t^z$.

(5) $\{(x_t, (\tilde{a'})_{x_t}) \mid \text{growing } t\} =$Sample-Extraction($\mathcal{S}', B$).

(6) $G = \sum_{i\in L} |(\tilde{a'})_i|^p + \sum_{t=1}^T \tilde{s}_t \cdot |(\tilde{a'})_{x_t}|^p$.

(7) Choose a sample u from the distribution:
$\mathbf{Pr}[u = i] = \frac{|(\tilde{a'})_i|^p}{G}$ if $i \in L$;
$\mathbf{Pr}[u = x_t] = \frac{\tilde{s}_t \cdot |(\tilde{a'})_{x_t}|^p}{G}$.

(8) Repeat Steps (1-7) independently $C\log n$ times. If in all repetitions the sample u returned is an injected coordinate, then report FAIL; else return $(u, (\tilde{a'})_i)$ if $i \in L$, or the value $(u, (\tilde{a'})_{x_t})$. Do this from the first repetition for which the sample u obtained is not an injected coordinate.

ListHH($\mathcal{S}'$, B)

(1) For $j \in [\log |a'|]$, independently sample functions $h_j : [|a'|] \to [|a'|]$ from a set of pairwise independent hash functions.

(2) Let $\mathcal{S}_j$ be the restriction of $\mathcal{S}'$ to those pairs (i, x) for which $h_j(i) \leq |a'|2^{-j}$.

(3) Return, for each $j \in [\log |a'|]$, $L_j = \text{HeavyHitters}(\mathcal{S}_j, B, 1/(C \log |a'|), \epsilon/C)$.

Set-Estimation($\{L_j \mid \text{all } j\}$)

(1) For growing t,

  (a) Choose the largest j for which L_j contains at least $1600\epsilon^{-2}T^2 \log^2 n$ elements of S_t. If there is such a j,

  - $S'_t = S_t \cap L_j$ and $j(t) = j$.
  - $\tilde{s}_t = 2^{j(t)} \cdot |S'_t|$.

(2) Return $\tilde{s}_t$ for each t for which a j was found, and return 0 for the other growing t.

Sample-Extraction($\mathcal{S}'$, B)

(1) $D \leftarrow C(\epsilon^{-3}T^4 \log^2 n)^2$, and $E \leftarrow C \log n$.

(2) For $j \in [\log(C\epsilon^{-1}|a'|)]$,

  (a) Independently sample $g_j : [C\epsilon^{-1}|a'|] \to [C\epsilon^{-1}|a'|]$ from a $(C \log(\epsilon^{-1}|a'|))$-wise independent family of hash functions.

  (b) For $e \in [E]$, independently sample $f_{j,e} : [C\epsilon^{-1}|a'|] \to [D]$ from a D-wise independent family.

(3) For $j \in [\log(C\epsilon^{-1}|a'|)]$, $d \in [D]$, and $e \in [E]$,

  (a) Let $\mathcal{S}_{j,d,e}$ be the restriction of $\mathcal{S}'$ to those pairs (i, x) for which $g_j(i) \leq (C\epsilon^{-1}|a'|)2^{-j}$ and $f_{j,e}(i) = d$.

  (b) For $B' = O(T^4\epsilon^{-3} \log^2 n)^{2/p}$, $M_{j,d,e} = \text{HeavyHitters}(\mathcal{S}_{j,d,e}, B', n^{-C}, \epsilon/C)$.

(4) For growing t, choose the largest $j = j(t)$ for which $\cup_{d,e} M_{j,d,e}$ contains $\geq 1600\epsilon^{-2}T^2 \log^2 n$ elements of S_t. If there is such a j:

  - $S''_t = S_t \cap \left(\cup_{d,e} M_{j(t),d,e}\right)$.
  - Let w_t be the element of S''_t that minimizes $g_{j(t)}$.

(5) Return $(w_t, \tilde{a}_{w(t)})$ for those t for which a $j(t)$ was found.

One of the subroutines that algorithm L_p-Sampler invokes is ListHH. This is an algorithm which takes in the stream $\mathcal{S}'$ and a parameter B, sub-samples the stream at a logarithmic number of

different rates, using independent hash functions $h_j : [|a'|] \to [|a'|]$, and then invokes Heavy-Hitters on the substream with B as an input parameter. Let C be a sufficiently large constant. The ListHH algorithm is invoked $C \log n$ times in Step 3 of L_p-Sampler, once for each value of z. Moreover, the Set-Estimation algorithm is also invoked for each value of z.

Let us fix a value of z in Step 3, and analyze the output of ListHH and Set-Estimation. For this value of z, let V_j denote the set of coordinates i in a for which $h_j(i) \leq |a'|2^{-j}$ in ListHH.

The Random Events. Define the events:

- $\mathcal{E}$: for all growing t and all $j \in [\log |a'|]$, $|S_t \cap V_j| \leq 40|S_t|2^{-j}T \log n$.
- $\mathcal{F}$: for all growing t and all $j \in [\log |a'|]$, if $\mathbf{E}[|S_t \cap V_j|] \geq 40\epsilon^{-2}T \log n$, then $|S_t \cap V_j| \in [(1-\epsilon)|S_t|2^{-j}, (1+\epsilon)|S_t|2^{-j}]$.
- $\mathcal{G}$: for all $j \in [\log |a'|]$, $F_p(a'(j)) \leq 40 \log n \cdot 2^{-j} F_p(a')$, where $a'(j)$ is the restriction of a' to the coordinates in V_j.
- $\mathcal{H}$: all invocations of HeavyHitters by ListHH succeed.

Lemma 77

$$\mathbf{Pr}[\mathcal{E} \wedge \mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H}] \geq 9/10.$$

Proof. First for the event $\mathcal{E}$, observe that $\mathbf{E}[|S_t \cap V_j|] = |S_t|2^{-j}$, and so by a Markov and a union bound over all t and all j, we obtain

$$\mathbf{Pr}[\mathcal{E}] \geq 39/40.$$

To bound $\mathbf{Pr}[\mathcal{F}]$, fix a growing t and a j satisfying

$$\mathbf{E}[|S_t \cap V_j|] \geq 40\epsilon^{-2}T \log n.$$

Let $I_x^{t,j}$ be an indicator for the event that an $x \in S_t$ is also in V_j.

Put $I^{t,j} = \sum_{x \in S_t} I_x^{t,j}$. Then

$$\mathbf{E}[I^{t,j}] = |S_t|2^{-j} \geq 40\epsilon^{-2}T \log n.$$

Since h_j is drawn from a pairwise-independent family, $\mathbf{Var}[I^{t,j}] \leq \mathbf{E}[I^{t,j}]$. By Chebyshev's inequality,

$$\mathbf{Pr}[|I^{t,j} - \mathbf{E}[I^{t,j}]| \geq \epsilon \mathbf{E}[I^{t,j}]] \leq \frac{1}{\epsilon^2 \mathbf{E}[I^{t,j}]} \leq \frac{1}{\epsilon^2 \cdot 40\epsilon^{-2}T \log n} \leq \frac{1}{40T \log n}.$$

By a union bound over growing t and the j satisfying $\mathbf{E}[|S_t \cap V_j|] \geq 40\epsilon^{-2}T \log n$, $\mathbf{Pr}[\mathcal{F}] \geq 39/40$.

For the event $\mathcal{G}$, for a fixed j, we have $\mathbf{E}[F_p(a'(j))] = 2^{-j}F_p(a')$. By a Markov bound,

$$\mathbf{Pr}[F_p(a') \geq 40 \log n \cdot 2^{-j} F_p(a')] \leq \frac{1}{40 \log n}.$$

By a union bound,

$$\mathbf{Pr}[\mathcal{G}] \geq 39/40.$$

Finally for the event $\mathcal{H}$, as HeavyHitters is invoked $\log |a'|$ times with error probability $1/(C \log |a'|)$, for a sufficiently large constant C, we get that

$$\mathbf{Pr}[\mathcal{H}] \geq 39/40.$$

By a union bound,

$$\mathbf{Pr}[\mathcal{E} \wedge \mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H}] \geq 9/10$$

as we want. □

Lemma 78 *Fix a $z \in [C \log n]$. Then with probability at least $9/10$, for all growing t, there is a $j = j(t)$ assigned to t by* Set-Estimation*, and also*

$$\tilde{s}_t^z \in [(1-\epsilon)|S_t|, (1+\epsilon)|S_t|].$$

Proof. We condition on $\mathcal{E}, \mathcal{F}, \mathcal{G}$, and $\mathcal{H}$ jointly occuring. Fix a growing t. We first show there is a $j = j(t)$ assigned to t by Set-Estimation. By Lemma 74,

$$|S_t| \geq \epsilon^{-3} T^2 \log^2 n.$$

It follows that, for small enough ϵ, there is a unique $j^* \geq 0$ for which

$$3200\epsilon^{-2}T^2 \log^2 n \leq 2^{-j^*}|S_t| < 6400\epsilon^{-2}T^2 \log^2 n. \tag{8.1}$$

Since S_t is growing, by Lemma 76,

$$|S_t| \cdot \eta^{pt} \geq \frac{\epsilon F_p(a')}{40T}.$$

Hence,

$$2^{-j^*}|S_t|\eta^{pt} \geq \frac{\epsilon 2^{-j^*} F_p(a')}{40T}.$$

Since event $\mathcal{G}$ occurs,

$$F_p(a'(j^*)) \leq 40 \log n \cdot 2^{-j^*} F_p(a'),$$

and so

$$2^{-j^*}|S_t|\eta^{pt} \geq \frac{\epsilon F_p(a'(j^*))}{1600T \log n}. \tag{8.2}$$

Since $2^{-j^*}|S_t| < 6400\epsilon^{-2}T^2\log^2 n$ by (8.1), we have

$$\eta^{p(t-1)} \geq \frac{\epsilon^3 F_p(a'(j^*))}{6400 \cdot 1600\eta^p T^3 \log^3 n}.$$

Now we use the fact that event $\mathcal{H}$ occurs, and so HeavyHitters($\mathcal{S}_{j^*}, B, 1/(C\log|a'|), \epsilon/C$) succeeds in reporting a list $L^z_{j^*}$ containing all i for which $|a'_i|^p \geq \frac{F_p}{B^{p/2}}$, where

$$B = (6400 \cdot 1600\epsilon^{-3}\eta^p T^3 \log^3 n)^{2/p}.$$

In particular, $L^z_{j^*}$ contains $V_{j^*} \cap S_t$, and these coordinates i will be correctly classifed into S_t given our assumption that the $|a'_i|$ are away from their upper boundaries. Finally, notice that $\mathbf{E}[|S_t \cap V_{j^*}|] = 2^{-j^*}|S_t|$, which is at least $3200\epsilon^{-2}T^2\log^2 n$ by (8.1). Hence, since event $\mathcal{F}$ occurs,

$$|V_{j^*} \cap S_t| \geq 3200(1-\epsilon)\epsilon^{-2}T^2\log^2 n \geq 1600\epsilon^{-2}T^2\log^2 n$$

for $\epsilon \leq 1/2$. It follows that t is assigned a value $j(t)$ in Set-Estimation.

Now we show that

$$\tilde{s}^z_t \in [(1-\epsilon)|S_t|, (1+\epsilon)|S_t|].$$

We must show that

$$2^{j(t)}|S'_t| \in [(1-\epsilon)|S_t|, (1+\epsilon)|S_t|],$$

for the $j(t)$ assigned to t by Set-Estimation (which need not equal j^*), and where $S'_t = S_t \cap L_{j(t)}$. Now $j(t)$ is such that

$$|L^z_{j(t)} \cap S_t| \geq 1600\epsilon^{-2}T^2\log^2 n.$$

Since event $\mathcal{E}$ occurs,

$$|S_t \cap V_{j(t)}| \leq 40|S_t|2^{-j(t)}T\log n.$$

It follows that

$$|S_t|2^{-j(t)} = \mathbf{E}[|S_t \cap V_{j(t)}|] \geq 40\epsilon^{-2}T\log n.$$

But then since event $\mathcal{F}$ occurs, it follows that

$$|S_t \cap V_{j(t)}| \in [(1-\epsilon)|S_t|2^{-j(t)}, (1+\epsilon)|S_t|2^{-j(t)}].$$

The same analysis above for j^* in (8.2) shows that for $j(t)$,

$$2^{-j(t)}|S_t|\eta^{pt} \geq \frac{\epsilon F_p(a'(j(t)))}{1600T\log n}.$$

Since we have shown that $L^z_{j^*}$ contains at least $1600\epsilon^{-2}T^2\log^2 n$ elements of S_t, we must necessarily have $j(t) \geq j^*$ since the Set-Estimation algorithm chooses the largest value of j for which L^z_j contains at least $1600\epsilon^{-2}T^2\log^2 n$ elements of S_t. But then

$$2^{-j(t)}|S_t| \leq 2^{-j^*}|S_t| < 6400\epsilon^{-2}T^2\log^2 n,$$

and so

$$\eta^{p(t-1)} \geq \frac{\epsilon^3 F_p(a'(j(t)))}{6400 \cdot 1600 \eta^p T^3 \log^3 n}.$$

Since we are conditioning on event $\mathcal{H}$ occurring, HeavyHitters($\mathcal{S}_{j(t)}, B, 1/(C \log |a'|), \epsilon/C$) succeeds in reporting a list $L^z_{j(t)}$ containing $V_{j(t)} \cap S_t$, and these coordinates i will be correctly classified into S_t given that the $|a'_i|$ are away from their upper boundaries. It follows that

$$|S'_t| = |V_{j(t)} \cap S_t| \in [(1-\epsilon)|S_t|2^{-j(t)}, (1+\epsilon)|S_t|2^{-j(t)}].$$

Hence, $\tilde{s}_t \in [(1-\epsilon)|S_t|, (1+\epsilon)|S_t|]$, as desired.

□

Corollary 79 *With probability* $\geq 1 - n^{-\Theta(C)}$*, for an absolute constant hidden in the* $\Theta(\cdot)$*,*

$$\tilde{s}_t = \mathit{median}_z\, \tilde{s}^z_t \in [(1-\epsilon)|S_t|, (1+\epsilon)|S_t|].$$

We now turn to Steps 5 and beyond, which are based on the following Sample-Extraction algorithm. Recall that the underlying vector is a' (the output of StreamTransformation) with length $|a'| = \mathrm{poly}(n)$. Let $C > 0$ be a sufficiently large constant.

The algorithm, like the ListHH algorithm, performs sub-sampling at a logarithmic number of different rates. The first difference is that the functions g_j used to do the sub-sampling are now $C \log(\epsilon^{-1}|a'|)$-wise independent, and map $[C\epsilon^{-1}|a'|]$ to $[C\epsilon^{-1}|a'|]$. This will allow us to apply a theorem of Indyk [Ind99] which relates hash function families with limited independence to ϵ-min-wise independent families, which we shall define in the analysis.

The second difference is that the surviving coordinates, for a given level of sub-sampling, are further hashed into D buckets using a D-wise independent hash function. This second bucketing operation is repeated independently $E = C \log n$ times. The ultimate goal of this bucketization is to guarantee that in a sub-sampling level j for which items from S_t are extracted, all surviving items from S_t are reported with very high probability. This requires showing that the heavy items, i.e., those from $\cup_{t' \geq t} S_{t'}$ in the substreams, are perfectly hashed into the D buckets for some repetition $e \in [E]$, with high probability. Given all surviving items from S_t, which is the set of coordinates i satisfying $g_j(i) \leq C\epsilon^{-1}|a'|2^{-j}$, we can find the $i \in S_t$ which minimizes g_j. Given that g_j is from an ϵ-min-wise independent family, this i is a random element of S_t, up to a small relative error.

Lemma 80 *With probability* $1 - n^{-\Omega(C)}$*, for each growing* t *a pair* $(w_t, a_{w(t)})$ *is returned by* Sample-Extraction*. Moreover, conditioned on returning a pair for* S_t*, for all* $\alpha \in S_t$ *we have*

$$\mathbf{Pr}[w_t = \alpha] = (1 \pm \epsilon) \cdot \frac{1}{|S_t|} \pm n^{-\Omega(C)}.$$

Proof. We let U_j be the set of coordinates for which $g_j(i) \leq (C\epsilon^{-1}|a'|)2^{-j}$, and $a''(j)$ be the vector a restricted to coordinates in U_j.

Define the random events:

- $\mathcal{E}'$: for all growing t and $j \in [\log(C\epsilon^{-1}|a'|)]$,

$$|S_t \cap U_j| \leq 2|S_t|2^{-j} + O(\log n).$$

- $\mathcal{F}'$: for all growing t and $j \in [\log(C\epsilon^{-1}|a'|)]$, if

$$\mathbf{E}[|S_t \cap U_j|] \geq \epsilon^{-2} \log n,$$

then

$$|S_t \cap U_j| \in [(1-\epsilon)|S_t|2^{-j}, (1+\epsilon)|S_t|2^{-j}].$$

- $\mathcal{H}'$: all invocations of HeavyHitters by Sample-Extraction succeed.

We need the following proposition.

Proposition 81 $\mathbf{Pr}[\mathcal{E}' \wedge \mathcal{F}' \wedge \mathcal{H}'] \geq 1 - n^{-\Theta(C)}$*, for an absolute constant in the* $\Theta(\cdot)$*.*

Proof. Since g_j is $C\log(\epsilon^{-1}|a'|)$-wise independent for sufficiently large C, one can use the concentration bound of Theorem 7. We can assume $\log(\epsilon^{-1}|a'|) = \Theta(\log n)$, as otherwise ϵ is so small that we can just store the entire vector a' in $\mathrm{poly}(\epsilon^{-1}\log n)$ bits of space.

Suppose we set $A = |S_t|2^{-j} + \Theta(\log n)$. Using the fact that $\mathbf{E}[|S_t \cap U_j|] = |S_t|2^{-j}$ and Theorem 7 for $d = \log(\epsilon^{-1}|a'|) = \Theta(\log n)$,

$$\begin{aligned}\mathbf{Pr}\left[|S_t \cap U_j| - |S_t|2^{-j} > |S_t|2^{-j} + \Theta(\log n)\right] &\leq 8\left(\frac{d\mathbf{E}[|S_t \cap U_j|] + d^2}{A^2}\right)^{d/2} \\ &\leq 8\left(\frac{O(\log n)|S_t|2^{-j} + O(\log^2 n)}{|S_t|^2 2^{-2j} + \Theta(\log^2 n)}\right)^{C\cdot\Theta(\log n)}.\end{aligned}$$

Now, there are two cases: (1) $|S_t|2^{-j} \leq \log n$, and (2) $|S_t|2^{-j} > \log n$. In both cases the RHS has the form $8(C')^{C\cdot\Theta(\log n)}$ for a constant $C' > 0$, and so it can be bounded by $n^{-\Theta(C)}$, which upper bounds $\mathbf{Pr}[\neg\, \mathcal{E}']$.

To upper bound $\mathbf{Pr}[\neg\, \mathcal{F}']$, we set $A = \epsilon\mathbf{E}[|S_t \cap U_j|] = \epsilon|S_t|2^{-j}$.

Then by Theorem 7,

$$\mathbf{Pr}\left[||S_t \cap U_j| - |S_t|2^{-j}| \geq \epsilon|S_t|2^{-j}\right] \leq 8\left(\frac{O(\log n)|S_t|2^{-j} + O(\log^2 n)}{\epsilon^2|S_t|^2 2^{-2j}}\right)^{C\cdot\Omega(\log n)}.$$

Using the premise of event $\mathcal{F}'$, namely, that $|S_t|2^{-j} \geq \epsilon^{-2}\log n$, the RHS can be bounded by $n^{-\Theta(C)}$, which upper bounds $\mathbf{Pr}[\neg\, \mathcal{F}']$.

Since HeavyHitters is invoked $\mathrm{poly}(\epsilon^{-1}\log n)$ times by Sample-Extraction with error parameter n^{-C}, by a union bound,

$$\mathbf{Pr}[\mathcal{E}' \wedge \mathcal{F}' \wedge \mathcal{H}'] \geq 1 - n^{-\Theta(C)},$$

for an absolute constant in the $\Theta(\cdot)$. □

We condition on these three events in the remainder of the proof of Lemma 80.

Fix a t that is growing. Below we will show that a $j(t)$ is assigned to t in Step 4 of Sample-Extraction. Next, we show that Sample-Extraction ensures that with probability $\geq 1-n^{-\Theta(C)}$, all coordinates in $U_{j(t)} \cap S_t$ are in S''_t. This is stronger than the guarantee of Lemma 78, which could only guarantee this with constant probability (or by minor modifications, $1-\mathrm{poly}(\epsilon \log^{-1} n)$ probability). One obstacle is that there may be no concentration in the random variables $F_p(a''(j))$, and if they are too large, then we cannot collect the heavy hitters in the corresponding substream. In the proof of Lemma 78 it sufficed to bound the $F_p(a''(j))$ using Markov's inequality. Here, we further partition the streams $\mathcal{S}_j$ into D pieces $\mathcal{S}_{j,1}, \ldots, \mathcal{S}_{j,D}$. We do this independently E times, so that for all j,

$$\forall e \in [E],\ \mathcal{S}_j = \cup_{d \in [D]} \mathcal{S}_{j,d,e}.$$

Let $a''(j,d,e)$ denote the restriction of the vector $a''(j)$ to coordinates that go to the d-th bucket in the e-th independent repetition.

Proposition 82 *With probability* $\geq 1 - n^{-\Theta(C)}$*, a* $j(t)$ *is assigned to* t *in Step 4 of* Sample-Extraction *and all coordinates in* $U_{j(t)} \cap S_t$ *are in* S''_t*, i.e.,*

$$\mathbf{Pr}[S''_t = U_{j(t)} \cap S_t] \geq 1 - n^{-\Theta(C)}.$$

Proof. Fix a $j \in [\log(C\epsilon^{-1}|a'|)]$. By Lemma 74, the number of elements in level sets that are not growing is at most $O(\epsilon^{-3}T^3 \log^2 n)$ (recall there are T levels). Next, for a $t' \geq t$ that is growing, since event $\mathcal{E}'$ occurs,

$$|S_{t'} \cap U_j| \leq 2|S_{t'}|2^{-j} + O(\log n). \tag{8.3}$$

Moreover, $S_{t'}$ cannot be much larger than S_t, as otherwise S_t could not be $\epsilon/(40T)$-contributing, as per Lemma 76. That is,

$$|S_t|\eta^{pt} \geq \epsilon F_p(a')/(40T),$$

and since $t' \geq t$ all coordinates in $S_{t'}$ have absolute value at least η^{t-1}. Then necessarily,

$$|S'_t| \leq 40T\eta^p \epsilon^{-1}|S_t|.$$

Combining this inequality with that of (8.3),

$$|S_{t'} \cap U_j| = O(T\epsilon^{-1}|S_t|2^{-j} + \log n),$$

and so

$$|\cup_{t' \geq t} S_{t'} \cap U_j| = O(T^2\epsilon^{-1}|S_t|2^{-j} + T\log n).$$

By Lemma 74,

$$|S_t| \geq \epsilon^{-3} T^2 \log^2 n.$$

Hence, for small enough ϵ, there is a unique $j^* \geq 0$ for which

$$3200\epsilon^{-2} T^2 \log^2 n \leq |S_t| 2^{-j^*} < 6400\epsilon^{-2} T^2 \log^2 n.$$

In the remainder of the proof we restrict our attention to those j for which $j \geq j^*$. For such j,

$$|\cup_{t' \geq t} S_{t'} \cap U_j| = O(\epsilon^{-3} T^4 \log^2 n).$$

Now, fix an $e \in [E]$.

Since the $f_{j,e}$ are chosen from a D-wise independent family for $D = C(\epsilon^{-3} T^4 \log^2 n)^2$, it follows that for a sufficiently large constant $C > 0$, with probability $\geq 1/2$, none of the items in $\cup_{t' \geq t} S_{t'} \cap U_j$ go to the same bucket (i.e., agree on the function $f_{j,e}$).

We now show that conditioned on $\mathcal{E}' \wedge \mathcal{F}' \wedge \mathcal{H}'$ and assuming that the items in $\cup_{t' \geq t} S_{t'} \cap U_j$ are perfectly hashed, for each $i \in U_j \cap S_t$, the coordinate i is returned by HeavyHitters$(\mathcal{S}_{j,d,e}, B, n^{-C}, \epsilon/C)$.

For this, we need a bound on $F_p(a''(j,d,e))$ for each bucket $d \in [D]$ in an iteration $e \in [E]$ containing an element of S_t. Fix a $d \in [D]$. Since event $\mathcal{E}'$ occurs, the number $y_{t'}$ of items in a growing $t' < t$ that collide in the d-th bucket is $O(|S_{t'}| 2^{-j} + \log n)$.

Since S_t is $\epsilon/(40T)$-contributing,

$$|S_t| \eta^{pt} \geq \frac{\epsilon F_p(a')}{40T},$$

and since

$$|S_{t'}| \eta^{p(t'-1)} \leq F_p(a'),$$

we obtain the bound

$$|S_{t'}| \leq 40T\epsilon^{-1} \eta^p |S_t| \eta^{pt - pt'}.$$

Since $j \geq j^*$, we have

$$2^{-j} |S_t| \leq 2^{-j^*} |S_t| = O(\epsilon^{-2} T^2 \log^2 n),$$

and so we obtain $y_{t'} = O(T^3 \epsilon^{-3} \eta^{pt - pt'} \log^2 n)$. It follows that the contribution to $F_p(a''(j,d,e))$ is at most

$$\eta^{pt'} y_{t'} = O(T^3 \epsilon^{-3} \eta^{pt} \log^2 n).$$

Hence, the total contribution from all $t' < t$ to $F_p(a''(j,d,e))$ is at most

$$O(T^4 \epsilon^{-3} \eta^{pt} \log^2 n).$$

But the contribution of the single item in S_t in this bucket to $F_p(a''(j,d,e))$ is at least η^{pt-p}. Since HeavyHitters is invoked with parameter $B' = O(T^4 \epsilon^{-3} \log^2 n)^{2/p}$, the single item in S_t in this bucket will be returned, using the fact that event $\mathcal{H}'$ occurs, and thus HeavyHitters succeeds.

Since $E = C \log n$, it follows that with probability $\geq 1 - n^{-\Theta(C)}$, there is a value of e for which the items in $\cup_{t' \geq t} S_{t'} \cap U_j$ are perfectly hashed, and hence with this probability

$$\cup_{d,e} M_{j,d,e} = S_t \cap U_j,$$

for any $j \geq j^*$. Now, notice that

$$\mathbf{E}[|S_t \cap U_{j^*}|] = |S_t| 2^{-j^*} \geq 3200 \epsilon^{-2} T^2 \log^2 n.$$

Hence, since event $\mathcal{F}'$ occurs,

$$|S_t \cap U_{j^*}| \geq (1 - \epsilon) \mathbf{E}[|S_t \cap U_{j^*}|] \geq 1600 \epsilon^{-2} T^2 \log^2 n.$$

This implies that in Step 4 of Sample-Extraction the value j^* satisfies the criterion that $\cup_{d,e} M_{j^*,d,e}$ contains at least $1600 \epsilon^{-2} T^2 \log^2 n$ elements of S_t. Since Sample-Extraction sets $j(t)$ to be the largest such j, the value t will be assigned a value $j(t) \geq j^*$.

The above implies that

$$\mathbf{Pr}[S''_t = U_{j(t)} \cap S_t] \geq 1 - n^{-\Theta(C)}.$$

This concludes the proof of the proposition. □

Now if $S''_t = U_{j(t)} \cap S_t$, then all coordinates i in S_t for which $g_j(i) \leq (C \epsilon^{-1} |a'|) 2^{-j}$ are in S''_t, and there are at least $1600 \epsilon^{-2} T^2 \log^2 n \geq 1$ of them. In particular, we can find the coordinate i for which

$$g_j(i) = \min_{i' \in S_t} g_j(i').$$

Now we apply Theorem 9 which shows how to construct a family of (ϵ, s)-min-wise independent hash functions (see Definition 8 for the definition of (ϵ, s)-min-wise independent hash functions). In particular we apply this theorem with $N = C \epsilon^{-1} |a'|$ and $s = |a'|$. Our family of hash functions is also $C \log \epsilon^{-1} |a'|$-wise independent. Hence, for $C > 0$ a sufficiently large constant, we have for all $i \in S_t$,

$$\mathbf{Pr}[g_j(i) = \min_{i' \in S_t} g_j(i')] = (1 \pm \Theta(\epsilon)) \cdot \frac{1}{|S_t|}.$$

This finishes the proof of Lemma 80. □

Theorem 83 *For any $p \in [0, 2]$, the probability that L_p-Sampler$(\mathcal{S}', p)$ returns $(i, (1 \pm \Theta(\epsilon)) a_i)$ is*

$$(1 \pm \epsilon) \frac{|a_i|^p}{F_p(a)} \pm n^{-C}$$

for an arbitrarily large constant $C > 0$. The space complexity is $\mathrm{poly}(\epsilon^{-1} \log n)$ bits. Ignoring the time of StreamTransformation*, which can be performed without looking at the data stream, the update time is $\mathrm{poly}(\epsilon^{-1} \log n)$.*

Proof. For $p \in (0, 2]$, by Corollary 79 and Lemma 80, with probability at least $1 - n^{-\Theta(C)}$,

$$\tilde{s}_t \in [(1 - \epsilon)|S_t|, (1 + \epsilon)|S_t|]$$

and for each growing t a sample w_t is returned with probability $|S_t|^{-1} \pm n^{-\Theta(C)}$. We also condition on the event that HeavyHitters succeeds in returning all coordinates not in growing S_t in Step 1; this only adds an additional n^{-C} to the error probability. It now follows that in Step 7,

$$\mathbf{Pr}[u = i] = (1 \pm \Theta(\epsilon))\frac{|a_i|^p}{F_p} \pm n^{-\Theta(C)}.$$

By Lemma 75, the total contribution of injected coordinates to $F_p(a')$ is $O(\epsilon)F_p(a')$. Hence, in Step 8, the probability that in $C \log n$ repetitions all samples are injected coordinates is at most $n^{-\Theta(C)}$. The statement of the theorem now follows by adjusting C and ϵ by constant factors. It is also easy to see that the space of the overall algorithm is $\mathrm{poly}(\epsilon^{-1} \log n)$ bits. The update time is dominated by that of the HeavyHitters subroutines, and is $\mathrm{poly}(\epsilon^{-1} \log n)$. □

We now show how to remove the assumption that the $|a'_i|$ are away from their boundaries.

Removing the assumption The above algorithm holds under the assumption that for all i, if $i \in S_t$, then $|a'_i| \leq \eta^t/(1 + \beta\epsilon)$, for a small constant $\beta > 0$, allowing HeavyHitters to accurately classify the coordinates that it finds. This assumption is easy to remove given an additional pass, since one can compute the $|a_i|$ exactly and then perform classification. To achieve a 1-pass algorithm, we make the following observation.

We note that the guarantees of our algorithm do not change by more than a factor of $\eta^p = 1 + \Theta(\epsilon)$ provided the classification of coordinates i into the level sets S_t is *consistent*. That is, if coordinate i is returned by multiple HeavyHitters invocations, then each invocation must classify it into the same level set S_t. Notice that consistency is easily enforced since the total number of items returned, across all HeavyHitters invocations, is $\mathrm{poly}(\epsilon^{-1} \log n)$, and hence the algorithm can simply remember a table indicating how previous coordinates returned were classified.

This effectively takes the underlying vector a', and multiplies some coordinates by a value of at most η (those that are mis-classified into their neighboring level set). Sampling from the resulting vector is equivalent to sampling from a', up to a factor of η. We need consistency for estimating the set sizes $|S_t|$, since we do not want to count one coordinate towards multiple S_t. Notice that unlike the algorithm of Indyk and Woodruff [IW05], we do not have to worry about some level sets no longer contributing because of mis-classification. This is because, as argued earlier, all injected coordinates are not near their boundaries, by definition, so they will still be correctly classified, and so all growing sets will still be $\mathrm{poly}(\epsilon \log^{-1} n)$-contributing (items from the non-growing sets do not undergo classification).

8.2 The Proof of Theorem 70 for $p = 0$

At this time we can work directly on the vector a, without introducing injected coordinates as needed for L_p-sampling, $p > 0$. However, notice that to remove the assumption that $\|a\|_p^p > \epsilon^{-2}$ for the L_p-sampler of the revious section, the L_0-Sampler is invoked on the vector a' of

that section. We should mention that in [FIS05], Frahling, Indyk, and Sohler also develop L_0-sampler. They needed this subroutine to study geometric problems such as maintaining approximate range spaces and costs of Euclidean spanning trees in the dynamic geometric data streams. Although the space complexity of our algorithm is inferior to their algorithm, but it gives a different solution for this basic problem.

Theorem 84 *There exists a* 1*-pass algorithm* L_0-Sampler *which, given a stream* $\mathcal{S}$ *of an underlying vector* a*, outputs a random non-zero coordinate* i *together with the value* a_i*, such that for all non-zero* a_i*, the probability that* L_0-Sampler *outputs* (i, a_i) *is*

$$(1 \pm \epsilon) \cdot \frac{1}{\|a\|_0} \pm n^{-C}$$

for an arbitrarily large constant $C > 0$*. The space complexity is* $\mathrm{poly}(\epsilon^{-1} \log n)$ *bits, and the update time is* $\mathrm{poly}(\epsilon^{-1} \log n)$*.*

Proof. Let $C > 0$ be a sufficiently large constant. We first assume that $\|a\|_0 \geq C$. W.l.o.g., $C\epsilon^{-1}n$ is a power of 2. We choose $C \log(\epsilon^{-1}n)$ independent hash functions $h_j : [C\epsilon^{-1}n] \to [C\epsilon^{-1}n]$ from a family of $C \log(\epsilon^{-1}n)$-wise independent hash functions. For $j = 0, \ldots, \log(C\epsilon^{-1}n)$, we say that a coordinate i of a survives with respect to h_j if $h_j(i) \leq 2^{-j}(C\epsilon^{-1}n)$. Let $\mathcal{S}_j$ be the restriction of updates in $\mathcal{S}$ to coordinates that survive with respect to h_j. Let $a(j)$ denote the restriction of the underlying vector a to coordinates that survive with respect to h_j. On each $\mathcal{S}_j$, we run a $\mathrm{poly}(\log n)$-space L_0-estimation algorithm to estimate $a(j)$ up to a factor of $(1 \pm 1/3)$ in the general turnstile model with error probability $n^{-\Theta(C)}$, e.g., the algorithm of [KNW10]. Denote the resulting estimate E_j. Further, for each $\mathcal{S}_j$ choose $C \log n$ independent C^3-wise independent hash functions $\psi_{j,r} : [n] \to [C^3]$, for $r \in [C \log n]$. We maintain the counters:

$$c_{j,r,d} = \sum_{\ell \text{ s.t. } h_j(\ell) \leq 2^{-j}n \text{ and } \psi_{j,r}(\ell) = d} a_\ell.$$

We repeat the entire procedure in the previous paragraph $C \log n$ times in parallel. We find some repetition for which there is a j for which $E_j \in [C/16, C]$. If there is no such j, then we output the symbol FAIL. Otherwise, we arbitrarily choose one repetition for which such a j was found.

From the counters $c_{j,r,d}$, one can recover the list L_j of all coordinates i in $a(j)$, together with their values a_i. This follows since for each fixed value of r, the at most $\frac{3}{2} \cdot E_j = \frac{3C}{2}$ survivors with respect to h_j are perfectly hashed with probability $\geq 2/3$ (for large enough C). Hence, with probability $\geq 1 - n^{-\Theta(C)}$, for all survivors i we have,

$$a_i = \mathrm{median}_r \, c_{j,r,\psi(i)}.$$

If the list L_j does not contain at least $(3/4)C/16$ coordinates, then we output FAIL. Else, we find the coordinate i in L_j which minimizes h_j, and output (i, a_i).

We can assume that all invocations of the L_0-estimation succeed. Now, assuming that $\|a\|_0 \geq C$, for each independent repetition, we claim that there exists a value of j for which $E_j \in \left[\frac{C}{16}, C\right]$

with constant probability. To show this, by the definition of E_j, it suffices to show that there exists a value of j for which with constant probability,

$$\|a(j)\|_0 \in \left[\frac{4}{3}\cdot\frac{C}{16}, \frac{2}{3}\cdot C\right] = \left[\frac{C}{12}, \frac{8C}{12}\right]$$

To see the latter, consider the unique value of j for which $\frac{C}{6} \leq 2^{-j}\|a\|_0 < \frac{C}{3}$. For each non-zero coordinate i, let X_i be an indicator variable which is one iff i survives with respect to h_j. Let $X = \sum_i X_i$. Then $\mathbf{E}[X] = 2^{-j}\|a\|_0$, and by pairwise-independence, $\mathbf{Var}[X] \leq \mathbf{E}[X]$. By Chebyshev's inequality,

$$\mathbf{Pr}[|X - \mathbf{E}[X]| \geq \frac{\mathbf{E}[X]}{2}] \leq \frac{4\mathbf{Var}[X]}{\mathbf{E}^2[X]} \leq \frac{12}{C} < \frac{1}{3},$$

where the last inequality follows for large enough C. Hence, with probability $1 - n^{-\Omega(C)}$, in some repetition we will find a j for which $E_j \in [C/16, C]$.

Finally, we appeal to Theorem 9. We apply the theorem with $N = C\epsilon^{-1}n$ and $s = n$. Our family of hash functions is also $C\log\epsilon^{-1}n$-wise independent. Hence, for $C > 0$ a sufficiently large constant, we have for all i such that $a_i \neq 0$,

$$\mathbf{Pr}[h_j(i) = \min_{i' \text{ s.t. } h_j(i')\neq 0} h_j(i')] = (1 \pm \epsilon)\cdot\frac{1}{\|a\|_0}.$$

It remains to handle the case $\|a\|_0 < C$. We will in fact show how to solve the case $\|a\|_0 \leq 2C$. The idea is just to use the perfect hashing data structure described above. Namely, choose $C\log n$ independent C^3-wise independent hash functions $\psi_{j,r} : [n] \to [C^3]$, for $r \in [C\log n]$. We maintain the counters:

$$c_{r,d} = \sum_{\ell \text{ s.t. } \psi_{j,r}(\ell)=d} a_\ell.$$

As described above, with probability $\geq 1 - n^{-\Theta(C)}$, for all i we have,

$$a_i = \text{median}_r\, c_{r,\psi(i)}.$$

Hence, we can recover the vector a in this case, for which L_0-sampling is trivial. In parallel we run a $\text{poly}(\log n)$-space L_0-estimation algorithm which can distinguish between the cases (1) $\|a\|_0 \leq C$ and (2) $\|a\|_0 \geq 2C$ with probability $\geq 1 - n^{-\Theta(C)}$. In the former case we run the sampling algorithm based on perfect hashing just described. In the latter case we run the sampling algorithm described previously. If $C < \|a\|_0 < 2C$, we can use either sampling algorithm.

It can be easily checked that ur overall algorithm is 1-pass, the space is $\text{poly}(\epsilon^{-1}\log n)$ bits, and the time is $\text{poly}(\epsilon^{-1}\log n)$.

□

Chapter 9

Applications of the L_p-Sampler

$\mathcal{L}_p$-sampler can be seen as a basic primitive in the turnstile model and we are expecting that it has many applications. In this chapter we discuss some applications of L_p-sampler, including an $(1 \pm \epsilon)$-estimator for $Z = F_k(a) = \sum_{i=1}^{n} |a_i|^k$ for $k > 2$ in Section 9.1, weighted sampling with deletions in Section 9.2, cascaded norms in Section 9.3, and finally heavy hitters and block heavy hitters in Section 9.2.

9.1 Moment Estimation

Recall that the problem of approximating k-th frequency moment of a stream is as follows. Let $0 < \epsilon, \delta \leq 1$. Given a stream $\mathcal{S}$ as $m = \text{poly}(n, M)$ updates of the form (i, x), where $i \in [n]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$, find an $(1 \pm \epsilon)$-estimator $F'_k(a)$ for $F_k(a) = F_p(S) = \sum_{i=1}^{n} |a_i|^k$ such that $\mathbf{Pr}[|F_k(a) - F'_k(a)| \leq \epsilon \cdot F_k(a)] \geq 1 - \delta$.

In [CK04], Coppersmith and Kumar reduce the problem of estimating F_k of a vector (see Definition 19) to the problem of sampling according to F_2. In particular Proposition 4.1 of that paper states "If there is a black-box such that

(1) it uses $\tilde{O}(1)$ space,

(2) it makes $\tilde{O}(1)$ passes over the input,

(3) each invocation outputs a random variable Y such that $\mathbf{Pr}[Y = i] \propto a_i^2$,

then there is a data stream algorithm for approximating F_3 that uses $\tilde{O}(n^{1/3})$ space and makes $\tilde{O}(1)$ passes over the input." As the sampler of Theorem 70 satisfies these conditions, we immediately obtain an alternative F_3-estimation algorithm in optimal space (up to small factors). This generalizes to arbitrary F_k, $k > 2$, and can be implemented in a single pass, giving an alternative algorithm to that of Indyk and Woodruff [IW05]. Our algorithm is the first that does

not use Nisan's pseudorandom generator as a subroutine, potentially making it more practical. Moreover, if we consider the two-party communication complexity of L_k-estimation, $k > 2$, we can use an $O(\log n)$-pass version of our sampler in Theorem 68 to improve the dependence on ϵ and k of known algorithms [BGKS06, IW05].

For $k > 2$, the following is our F_k-estimation algorithm, which succeeds with probability at least $3/4$. To amplify the success probability to $1 - n^{-C}$, repeat the algorithm $O(C \log n)$ times and take the median of the outputs. Inside the F_k-Estimation, we can use either 1-pass L_2-Sampler of Theorem 70 which gives a 1-pass $n^{1-2/k}\text{poly}(\epsilon^{-1} \log n)$-space algorithm or use $O(\log n)$-pass L_2-sampler of Theorem 68 which gives an $O(\log n)$-pass algorithm with $O(n^{1-2/k}\epsilon^{-4} \log^4 n)$ bits of space. Depending whether we prefer the pass complexity or the space complexity in terms of $\epsilon, \log n$ we can choose the former algorithm or the latter one.

In what follows we will assume that the input parameter $\epsilon > 16/n$, as otherwise we can compute F_k exactly in $O(\log(n))/\epsilon)$ bits of space by keeping a counter for each coordinate.

F_k-Estimation:

(1) Initialize $\epsilon'' = \epsilon/8$, and $T = O(n^{1-2/k})/(\epsilon'')^2$.

(2) Run an ϵ''-relative-error L_2-Sampler algorithm $4T$ times in parallel, and let the first T output values be $a_{i_1}, a_{i_2}, \ldots, a_{i_T}$. If more than $3T$ of the outputs of L_2-Sampler are FAIL, then output FAIL.

(3) In parallel, run F_2-Estimation$([n], \epsilon'', 1/8)$ of [KNW10]. Denote the output by $\tilde{F}_2$.

(4) Output $\frac{\tilde{F}_2}{T} \cdot \sum_{j=1}^{T} |a_{i_j}|^{k-2}$.

Theorem 85 *For any $k > 2$ and $\epsilon < 1$, instantiating F_k-Estimation with our 1-pass L_2-Sampler algorithm results in a $(1 \pm \epsilon)$-approximation to F_k with probability at least $3/4$. The space complexity is $n^{1-2/k} \cdot \text{poly}(\epsilon^{-1} \log n)$ bits.*

Proof. We condition on the event $\mathcal{E}$ that F_2-Estimation succeeds, which occurs with probability at least $7/8$. We can thus assume that $F_k \neq 0$, since if $F_k = 0$ we will have $\tilde{F}_2 = 0$ and we will correctly output 0 in Step 4. In the remainder of the proof, assume that $F_k \geq 1$.

We also condition on the event $\mathcal{F}$ that at most $3T$ of the outputs of L_2-Sampler are FAIL. By Theorem 70 and a Chernoff bound, event $\mathcal{F}$ occurs with probability $1 - e^{-\Omega(T)}$.

For $i \in [n]$, let q_i be the probability that coordinate i is returned by an invocation of L_2-Sampler, given that L_2-Sampler does not output FAIL. By Theorem 70, $q_i = (1 \pm \epsilon'')|a_i|^2/F_2$. So for any

$j \in [T]$, we have

$$\begin{aligned}\mathbf{E}[|a_{i_j}|^{k-2}] &= \sum_{i=1}^{n} q_i |a_i|^{k-2} \\ &= \sum_{i=1}^{n} \left(1 \pm \epsilon''\right) \frac{|a_i|^2}{F_2} \cdot |a_i|^{k-2} \\ &= \left(1 \pm \epsilon''\right) \frac{F_k}{F_2}.\end{aligned}$$

Let $G_j = |a_{i_j}|^{k-2}$, and $G = \frac{\tilde{F}_2}{T} \cdot \sum_{j=1}^{T} G_j$. Then $\mathbf{E}[G_j] = (1 \pm \epsilon'')\frac{F_k}{F_2}$. Thus, since event $\mathcal{E}$ occurs, $\mathbf{E}[G] = T(1 \pm \epsilon'')(1 \pm \epsilon'')\frac{F_k F_2}{T F_2} = (1 \pm \epsilon/2)F_k$, for sufficiently small ϵ. By independence of the G_j and the fact that $\tilde{F}_2 \leq 2F_2$,

$$\begin{aligned}\mathbf{Var}[G] &\leq \frac{4F_2^2}{T^2} \sum_{j=1}^{T} \mathbf{Var}[G_j] \\ &= \frac{4F_2^2}{T^2} \cdot T \sum_{i=1}^{n} q_i |a_i|^{2k-4} \\ &\leq \frac{4F_2^2}{T} \sum_{i=1}^{n} \left(\frac{2|a_i|^2}{F_2}\right) |a_i|^{2k-4} \\ &= O\left(\frac{F_2 F_{2k-2}}{T}\right).\end{aligned}$$

To bound $F_2 F_{2k-2}$, we use Hölder's inequality in the same way as previous work [AMS96, CK04, IW05]. Namely,

$$\sum_{i=1}^{n} |a_i b_i| \leq \left(\sum_{i=1}^{n} |a_i|^p\right)^{1/p} \left(\sum_{i=1}^{n} |b_i|^q\right)^{1/q}$$

for any reals p, q with $p, q > 1$ and $1/p + 1/q = 1$. Taking $a_i = a_i^2$, $b_i = 1$, $p = k/2$, $q = k/(k-2)$, we have $F_2 \leq F_k^{2/k} n^{1-2/k}$. Moreover, $F_{2k-2}^{1/(2k-2)} \leq F_k^{1/k}$ using the fact that the L_k-norms of a vector are non-increasing in k, and $k \leq 2k-2$ for $k \geq 2$. So $F_{2k-2} \leq F_k^{2-2/k}$. Taking the product of the inequalities, $F_2 F_{2k-2} \leq n^{1-2/k} F_k^2$.

Thus, by our choice of T, $\mathbf{Var}[G] = O(\epsilon^2 \mathbf{E}^2[G])$. It follows by Chebyshev's inequality that

$$\mathbf{Pr}[|G - \mathbf{E}[G]| > \frac{\epsilon}{4}\mathbf{E}[G]] \leq 1/16,$$

for an appropriate choice of constant in the big-Oh defining T. In this case, for G we have

$$(1-\epsilon)F_k \leq (1-\epsilon/4)(1-\epsilon/2)F_k \leq G \leq (1+\epsilon/4)(1+\epsilon/2)F_k \leq (1+\epsilon)F_k,$$

for sufficiently small ϵ.

It follows that with probability at least $7/8 - e^{-\Theta(t)} - 1/16 \geq 3/4$, the output of the algorithm is a $(1 \pm \epsilon)$-approximation to F_k. The space complexity is dominated by Step 2 and is $n^{1-2/k} \cdot \mathrm{poly}(\epsilon^{-1} \log n)$. □

9.2 Weighted Sampling with Deletions

Cormode, Muthukrishnan, and Rozenbaum [CMI05] state that "A fundamental question that arises is to design algorithms to maintain a uniform sample of the *forward* distribution under *both* insertions and deletions or show that it is impossible." Here, by forward distribution, the authors mean to return a sample i with probability $|a_i|/\|a\|_1$, even if coordinate i undergoes deletions in the stream. Setting $p = 1$ in Theorem 70 therefore resolves the main open question of [CMI05] (up to a small relative error. As sampling in the presence of deletions is a useful primitive, we expect this to have many more applications. For instance, if the support of the underlying vector a is at most k, by applying Theorem 70 with $p = 0$ at most $O(k \log k)$ times, with constant probability all k non-zero items will be found.

Also, in [FIS05], Frahling, Indyk, and Sohler study geometric problems such as maintaining approximate range spaces and costs of Euclidean spanning trees. They need a routine which given a pointset P undergoing multiple insertions and deletions, maintains a random element from P. Applying Theorem 70 with $p = 0$ gives another solution to this problem. However, the space complexity of our algorithm is inferior to their algorithm.

9.3 Cascaded Norms

Recall that the problem of cascaded norms is defined as follows. Let $0 < \epsilon, \delta \leq 1$. Let $\mathcal{S}$ be a stream of $m = \text{poly}(n, M)$ updates of the form (i, j, x) to items in a $[n] \times [d]$ matrix A, where $i \in [n], j \in [d]$ and $x \in \{-M, -M+1, \ldots, M-1, M\}$. The problem of estimating cascaded norms is to find a $(1 \pm \epsilon)$-approximation to $F_k(F_p)(A)$, where $F_k(F_p)(a)$ means we first applying F_p to each row of A, and then apply F_k to the resulting vector of values, i.e,

$$F_k(F_p)(A) = \sum_{i \in [n]} (\sum_{j \in [d]} |A[i,j]|^p)^k.$$

In [JW09], Jayram and Woodruff give 1-pass algorithms for estimating cascaded moments of an $n \times d$ matrix A. Namely, they show that for $k \geq 1$ and $p \geq 2$, the 1-pass space complexity of outputting a $(1 \pm \epsilon)$-approximation to $F_k(F_p)(A)$ is $n^{1-2/(kp)} d^{1-2/p} \text{poly}(\epsilon^{-1} \log(nd))$.

They leave open the question of estimating $F_k(F_p)(A)$ for $k \geq 1$ and $p < 2$, though they prove an $\Omega(n^{1-1/k})$ space lower bound for this problem. The only other work in this regime is due to Cormode and Muthukrishnan [CM05b] who prove an $n^{1/2}\text{poly}(\epsilon^{-1} \log(nd))$ upper bound for estimating $F_2(F_0)(A)$ assuming that all stream updates are positive.

Theorem 70 implies a near-optimal $n^{1-1/k}\text{poly}(\epsilon^{-1} \log(nd))$-space 1-pass upper bound for any $k \geq 1$ and $p \in [0, 2]$, which, together with the results of [JW09], closes the problem for $k \geq 1$ and any $p \geq 0$, up to small factors. As our algorithm works in the general turnstile model, this also improves the algorithm of [CM05b] for estimating $F_2(F_0)$.

$F_k(F_p)$-Estimation(A, ϵ):

(1) Initialize $T = n^{1-1/k}\text{poly}(\epsilon^{-1}\log(nd))$.

(2) Run L_p-Sampler algorithm T times in parallel.

(3) Feed the row IDs of these samples into the 1-pass F_k-estimation algorithm given in [JW09].

The 1-pass F_k-estimation algorithm given in [JW09] is similar to the $n^{1-1/k}\text{poly}(\epsilon^{-1}\log(nd))$-space algorithm for estimating F_k of Lemma 25, but given a vector $(b_1, \ldots, b_n)$, it requires only sampling $n^{1-1/k}\text{poly}(\epsilon^{-1}\log(nd))$ coordinate IDs i, rather than approximations to their frequencies (the b_i), where i is sampled with probability $\frac{|b_i|}{\sum_{i=1}^n |b_i|}$. In our case, we run the sampler on the matrix A, obtaining an entry in a given row i with probability

$$(1 \pm \epsilon)\frac{\sum_{j=1}^d |A[i,j]|^p}{\sum_{i=1}^n \sum_{j=1}^d |A[i,j]|^p}.$$

Thus, b_i in the F_k subroutine is equal to

$$(1 \pm \epsilon)\sum_{j=1}^d |A[i,j]|^p.$$

Theorem 86 *For any $p \in [0,2], k \geq 1$, there is a 1-pass streaming algorithm which, with probability $\geq 1 - 1/\text{poly}(nd)$, outputs a $(1 \pm \epsilon)$-approximation to $F_k(F_p)(A)$ using space*

$$n^{1-1/k}\text{poly}(\epsilon^{-1}\log(nd)).$$

9.4 Heavy Hitters and Block Heavy Hitters

The classical heavy hitters problem is to report all coordinates i for which $|a_i| \geq \phi\|a\|_p$, where ϕ is an input parameter. For $p = 1$ this is solved by the CountMin data structure of Cormode and Muthukrishnan [CM05b], and for $p = 2$ by the CountSketch data structure by Charikar, Chen, and Farach-Colton [CCFC02]. Notice that Theorem 70 immediately implies an algorithm for every $p \in [0,2]$. Indeed, run L_p-sampler of Theorem 70 $O(\phi^{-1}\log\phi^{-1})$ times in parallel. Then with constant probability, the list of samples contains all heavy hitters, and using the probabilities returned, one can ensure that if $|a_i|^p \leq (\phi - \epsilon)\|a\|_p^p$, then i is not reported. Notice that our algorithm works in the turnstile model.

Another immediate application is that of computing block heavy hitters, which is the problem of reporting all rows a^i of an $n \times d$ matrix A for which $\|a^i\|_1$ is at least a ϕ fraction of the L_1-norm $\|A\|_1$ of A (i.e., $\|A\|_1 = \sum_{j=1}^n \|a^j\|_1$). These rows are called the block heavy hitters, and are a crucial building block in a streaming algorithm of Andoni, Indyk, and Kraughtgamer [AIK09] that

constructs a small-size sketch for the Ulam metric under the edit distance. In [ABI08], Andoni, DoBa, and Indyk devise a 1-pass algorithm for this problem using $\mathrm{poly}(\phi^{-1} \log n)$ bits of space.

Notice that if a^i is a block heavy hitter, then if we sample a random entry of A proportional to its absolute value, the probability it comes from row i is at least ϕ. Moreover, based on the number of times an item from row j is sampled, one can detect if

$$\|a^j\|_1 \leq (\phi - \epsilon)\|A\|_1.$$

Hence, Theorem 70 immediately implies the main result of [ABI08]. It should be noted that the proof of Theorem 70 does not rely on Nisan's pseudorandom generator or go through p-stable distributions, which could potentially make our block heavy hitters algorithm more practical than that of [ABI08]. Moreover, Theorem 70 immediately gives the analogous result for every L_p with $p \in [0, 2]$.

Bibliography

[AB02] R. Albert and A. L. Barabsi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, 2002.

[ABI08] A. Andoni, K. Do Ba, and P. Indyk. Block heavy hitters. *Manuscript*, 2008.

[AC09] N. Ailon and B. Chazelle. The fast johnson-lindenstrauss transform and approximate nearest neighbors. *SIAM Journal on Computing*, 39(1):302–322, 2009.

[Ach03] D. Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.

[AHPV04] P. K. Agarwal, S. Har-Peled, and K.R. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51(4):606–635, 2004.

[AIK09] A. Andoni, P. Indyk, , and R. Krauthgamer. Overcoming the l1 non-embeddability barrier: algorithms for product metrics. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 865–874, 2009.

[AMS96] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 20–29, 1996.

[AS00] N. Alon and J. Spencer. *The probabilistic method*. J. Wiley and Sons, 2000.

[BGKS06] L. Bhuvanagiri, S. Ganguly, D. Kesh, and C. Saha. Simpler algorithm for estimating frequency moments of data streams. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 708–713, 2006.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BN03] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[BN04] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1-3):209–239, 2004.

[BO04] A. L. Barabasi and Z.N. Oltvai. Network biology: Understanding the cells functional organization. *Nature Reviews : Genetics*, 5(1):101–113, 2004.

[BR94] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 276–287, 1994.

[BYJKS02] Z. Bar-Yossef, T.S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 209–218, 2002.

[CCFC02] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proceedings of the 29th Annual International Colloquium on Automata, Languages and Programming (ICALP)*, pages 693–703, 2002.

[Che52] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23(4):493 – 507, 1952.

[Che06] K. Chen. On coresets for k-clustering in metric and euclidean spaces and their applications. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1177–1185, 2006.

[CK04] D. Coppersmith and R. Kumar. An improved data stream algorithm for frequency moments. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 151–156, 2004.

[CKS03] A. Chakrabarti, S. Khot, and X. Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity (CCC)*, pages 107–117, 2003.

[CKVW06] D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. *ACM Transactions on Database Systems (TODS)*, 31(4):1499–1525, 2006.

[CM05a] G. Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. 55(1):58–75, 2005.

[CM05b] G. Cormode and S. Muthukrishnan. Space efficient mining of multigraph streams. In *Proceedings of the 11th ACM SIGMOD Symposium on Principles of Database Systems (PODS)*, pages 271–282, 2005.

[CMI05] G. Cormode, S. Muthukrishnan, and I.Rozenbaum. Summarizing and mining inverse distributions on data streams via dynamic inverse sampling. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB)*, pages 25–36, 2005.

[DDL+90] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[DFK+04] P. Drineas, A. M. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine Learning*, 56(1-3):9–33, 2004.

[DKR02] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 82–90, 2002.

[DKS10] A. Dasgupta, R. Kumar, and T. Sarlos. A sparse johnson-lindenstrauss transform. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 341–350, 2010.

[DRVW06] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(12):225–247, 2006.

[DV06] A. Deshpande and S. Vempala. Adaptive sampling and fast low-rank matrix approximation. In *Proceedings of the 10th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 292–303, 2006.

[DV07] A. Deshpande and K. Varadarajan. Sampling-based dimension reduction for subspace approximation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 641–650, 2007.

[FFS06] D. Feldman, A. Fiat, and M. Sharir. Coresets for weighted facilities and their applications. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 315–324, 2006.

[FIS05] G. Frahling, P. Indyk, and C. Sohler. Sampling in dynamic data streams and applications. In *Proceedings of the 21st Annual Symposium on Computational Geometry (SoCG)*, pages 142–149, 2005.

[FKK+00] R. Fagin, A. R. Karlin, J. M. Kleinberg, P. Raghavan, S. Rajagopalan, R. Rubinfeld, M. Sudan, and A. Tomkins. Random walks with "back buttons". In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 484–493, 2000.

[FKV04] A. Frieze, R. Kannan, and S. Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.

[FL07] D. Feldman and M. Landberg. Algorithms for approximating subspaces by subspaces. *Manuscript*, 2007.

[Fri04] E. Friedgut. Hypergraphs, entropy, and inequalities. *The American Mathematical Monthly*, 111(9):749–760, 2004.

[FS05] G. Frahling and C. Sohler. Coresets in dynamic geometric data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 209–217, 2005.

[GL96] G.H. Golub and C.F. Van Loan. *Matrix Computations*. 1996.

[GSS08] S. Ganguly, A.N. Singh, and S. Shankar. Finding frequent items over general update streams. In *Proceedings of the 31st International Conference on Scientific and Statistical Database Management (SSDBM)*, pages 204–221, 2008.

[Hau92] D. Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Journal Information and Computation*, 100(1):78–150, 1992.

[Hoe63] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[HP06] S. Har-Peled. Low-rank matrix approximation in linear time. *Manuscript*, 2006.

[HPK05] S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. In *Proceedings of the 21st Annual Symposium on Computational Geometry (SoCG)*, pages 126–134, 2005.

[HPM04] S. Har-Peled and S. Mazumdar. Coresets for k-means and k-median clustering and their applications. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 291–300, 2004.

[IKI94] M. Inaba, N. Katoh, and H. Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th Annual Symposium on Computational Geometry (SoCG)*, pages 332–339, 1994.

[IM98] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.

[Ind99] P. Indyk. A small approximately min-wise independent family of hash functions. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 454–456, 1999.

[Ind06] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *Journal of the ACM*, 53(3):307–323, 2006.

[IW05] P. Indyk and D. P. Woodruff. Optimal approximations of the frequency moments of data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 202–208, 2005.

[JL84] W.B. Johnson and J. Lindenstrauss. Extensions of lipschitz mappings into a hilbert sapce. *Contemporary Mathematics*, 54(2):129–138, 1984.

[JV01] K. Jain and V. V. Vazirani. Facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001.

[JW09] T.S. Jayram and D.P. Woodruff. The data stream space complexity of cascaded norms. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 765–774, 2009.

[KN10] D. M. Kane and J. Nelson. A derandomized sparse johnson-lindenstrauss transform. *Manuscript*, 2010.

[KNW10] D.M. Kane, J. Nelson, and D. P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1161–1178, 2010.

[KSS04] A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time $(1+\epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 454–462, 2004.

[KSS05] A. Kumar, Y. Sabharwal, and S. Sen. Linear time algorithms for clustering problems in any dimensions. In *Proceedings of the 32nd Annual International Colloquium on Automata, Languages and Programming (ICALP)*, pages 1374–1385, 2005.

[Li08] P. Li. Estimators and tail bounds for dimension reduction in l_p $(0 < p \leq 2)$ using stable random projections. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 10–19, 2008.

[Mat08] J. Matousek. On variants of the johnson-lindenstrauss lemma. *Random Structures and Algorithms*, 33(2):142–156, 2008.

[MP04] R.R. Mettu and C.G. Plaxton. Optimal time bounds for approximate clustering. *Machine Learning*, 56(1-3):35–60, 2004.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge Univ. Press, 1995.

[New04] M. Newman. Coauthorship networks and patterns of scientific collaboration. *National Academy of Sciences*, USA 101(016131):5200–5205, 2004.

[OR02] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric clustering problems. *Journal of the ACM*, 49(2):139–156, 2002.

[Pea01] Karl Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series*, 2(6):559–572, 1901.

[PRTV00] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, 61(2):217–235, 2000.

[Sar06] T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 143–152, 2006.

[SV07] N.D. Shyamalkumar and K. Varadrajan. Efficient subspace approximation algorithms. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 532–540, 2007.

[Vos91] M. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering.*, 17(9):972–975, 1991.