Beyond Worst-Case Analysis in Privacy and Clustering: Exploiting Explicit and Implicit Assumptions

Or Sheffet

CMU-CS-13-120

August 2013

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Avrim Blum, Chair
Anupam Gupta
Venkatesan Guruswami
Kunal Talwar, Microsoft Research

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by the National Science Foundation under grant numbers CCF0830540, CCF1101215, and CCF1116892; the U.S. Army Research Office under grant number W91NF0910273; and the Microsoft Research-Carnegie Mellon Center for Computational Thinking.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Non Worst-Case Analysis, Differential Privacy, Clustering, Algorithms.

להורי, באהבה.

זה הכל בזכותכם.

Abstract

This thesis can be viewed as a collection of work in differential privacy and in clustering. In its first part we discuss work aimed at preserving differential privacy in a social network, with respect to either the presence/absence of a single edge [41], or with respect to changing all edges adjacent to one node [42]. In its second part we discuss multiple clustering problems, focusing on the k-means and k-median problems. We show how to correctly cluster an instance whose optimal k-means solution either satisfies a certain stability condition [20], or is resilient to small constant metric perturbations [21], or with cluster centers that satisfy a particular separation condition [22].

Alternatively, this thesis can be viewed as an investigation of specific non-worst case analysis paradigms. The common theme among of all results composing this thesis is that they all introduce algorithms whose guarantees are meaningful only for a subset of inputs – for inputs that satisfy certain nice properties, or assumptions. These assumptions can be roughly divided into two types, which we refer to as *explicit* or *implicit*. Explicit assumptions give a very specific and quantifiable characterization of the input (e.g. clustering instances with the distance between any pair of cluster centers larger than a specific bound). On the other hand, implicit assumptions are harder to characterize. Implicit assumptions pose a certain property that the input should satisfy, due to some compelling "real-life" reasoning (e.g. justifying a particular value of k for a k-means clustering instance), and often give much leeway as to the particular structure of the input. In this thesis, we exhibit multiple examples of assumptions of both kinds, in differential privacy and clustering, and give algorithms that take advantage of these assumptions. In particular, we show how tasks that are provably hard become feasible under suitable assumptions; tasks like providing accurate answers for queries over graph while preserving privacy on the node level, or giving a c-approximation for the k-median objective for $c < 1 + e^{-1}$.

Acknowledgments

First and far most, I would like to thank my advisor, Avrim Blum. Avrim embodies everything one can ask for in an advisor: brilliant (as an understatement), quiet and calm, methodical and thorough, and extremely kind. Avrim gives his students the freedom to choose their own problems, and at the same time he serves as a prolific source of knowledge, concrete ideas and spot-on intuition. He is truly an inspiration.

I would like to thank all of my committee members for their advise as well: Anupam Gupta, Venkat Guruswami and Kunal Talwar. Many thanks also go to my co-authors: Nina Balcan, Jeremy Blocki (it took him a long while, but eventually he got my inappropriate sense of humor), Peter Bro-Miltersen (whose cynical humor was a refreshing European change), Anupam Datta (though he is soft spoken, his words of advice resonated with me for a long while), Samuel Ieong, Saranga Komanduri, Nina Mishra (who is the second kindest person I met, next to Avrim), Jamie Morgenstern (so cheerful it's eerie), Ariel Procaccia (another great source for lots of reassuring advice), and Santosh Vempala (it's a rare treat just to sit and listen to Santosh and Avrim talk).

CMU is a wonderful and friendly place, and I am extremely grateful to all the professors and my fellow students with whom I repeatedly had many helpful discussions. There are truly too many of you to mention... I would also like to thank CMU's always-helpful administrative staff, especially Deborah Cavlovich, Marilyn Walgora and Nicole Stenger.

Special thanks go to Pranjal Awasthi. As young researchers Avrim teamed us up, and much like the rest of Avrim's advice, this too was worth its weight in gold. Pranjal proved to be a superb collaborator and no less of a good friend, and our collaboration turned out to be extremely fruitful. So fruitful in fact, that Avrim himself started to worry we might write the same thesis twice...

Many thanks to all of my friends who constantly provide much needed emotional support. I love you and I am, as ever, indebted to you. Last but not least, to my family and especially to my parents – my two main role-models for pretty much everything in life.

And of course, to anyone who reads this thesis.

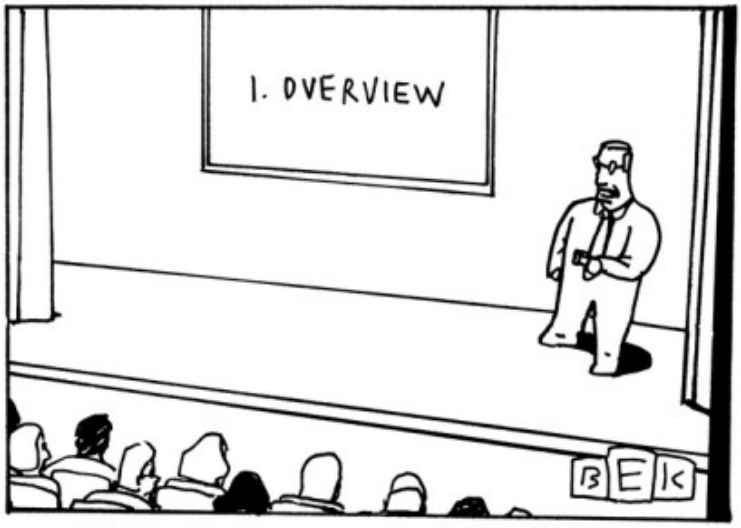
Contents

List of Figures

List of Tables

Part I

Overview



"First, I want to give you an overview of what I will tell you over and over again during the entire presentation."

Chapter 1

Introduction

This thesis details new results in differential privacy and in clustering that are obtained using non-worst case analysis. Whereas traditional approach requires us to devise algorithms that are suppose to work for any instance (algorithms which are tractable even in the worst-case), this thesis takes a complementary approach. In each of the works detailed in this thesis we assume the given instance has a certain property that makes it "nice", and leveraging on this assumption we devise tractable algorithms even for problems which are provably hard in general.

Indeed, one may view this thesis simply as an amalgamate of work in differential privacy and in clustering, and so it is partitioned into a part that deals with differential privacy (Chapters 5 and 6) and a part that deals with clustering (Chapters 7, 8 and 9). We prefer however to view this thesis as a study of non-worst case analysis, where the main theme of this thesis is to investigate assumptions: what explicit assumptions make certain algorithms useful, and what characterizes instances that adhere to certain implicit assumptions, often made in "real-life".

1.1 Problem Overview: Privacy and Clustering

1.1.1 Differential Privacy

It is no secret that privacy is a rising concern in today's world. Databases of enormous magnitude are kept essentially everywhere, from hospitals who keep the sensitive data of patients, to network providers who keep track of web-surfers, and to government held censuses. Naturally, participants in such datasets wish to keep their sensitive data private. In

other words, data curators, with access to participants' sensitive data, need to *guarantee* each participant that the data will be accessed only in a way that will not reveal (much about) her identity. In a series of works [61, 51, 44, 63, 62], the rigorous notion of differential privacy was established.

Definition 1.1. *A dataset D is a (multi-)set of elements from some predefined domain $\mathcal{X}$. Two datasets D and D' are called* neighbors *if they differ only on the details of a single individual. An algorithm ALG is said to preserve (ϵ, δ)-differential privacy if for any two neighboring datasets and any subset S of potential outputs it holds w.p. $\geq 1 - \delta$ that*

$$\mathbf{Pr}[\mathsf{ALG}(D) \in S] \leq e^{\epsilon}\mathbf{Pr}[\mathsf{ALG}(D') \in S]$$

By now, there are hundreds of works in differential privacy, detailing an abundance of privacy preserving algorithms aimed at a multitude of tasks. Yet the core technique that the majority of these algorithms apply is the same technique originally proposed by Dwork et al [63]. Given a query function f, the curator first estimates the global sensitivity of f, denoted $GS(f) = \max_{D,D'} f(D) - f(D')$, then outputs $f(D) + X$ where X is a random noise sampled from a suitable distribution, typically a Laplace or Gaussian, with 0 mean and variance proportional to $(GS(f)/\epsilon)^2$.

1.1.2 Clustering

Clustering is a "brand name" for a variety of tasks in computer science, which roughly translates to finding a good partition of a given collection of n datapoints. In this thesis we take the machine learning view of clustering – unbeknownst to us, there exists some true target partition, and the partition our algorithm retrieves should be as close as possible to this target clustering. In general, we assume the target partition has k clusters, but no datapoint is labeled. Instead, we are given access to a metric – the distances between any two datapoints, and we aim to retrieve a good partition of the dataset using the given metric. As such, we must assume a connection between the metric and the target clustering.

Indeed, multiple such connections were proposed. A common paradigm in clustering is to impose a quantitative objective, such as k-median or k-means, with the assumption that the target clustering identifies with (or is close to) the k-partition that minimizes this objective. Unfortunately, both the k-median and the k-means objective are NP-hard, and so much effort has been put into approximating these objectives [90, 98]. A separate paradigm investigates the notion that the distances reflect (dis)similarities between datapoints. So roughly speaking, based on the notion that points in different clusters have "large" distances and points from the same cluster have "small" distance, one should

be able to retrieve a good partition of the data [27]. The last paradigm this thesis discusses deals with points in Euclidean space, where one assumes a separation condition between the centers of respective clusters. This often reflects the case of a mixture model, where each cluster is characterized by a certain distribution, and the cluster points are iid samples from this distribution. In this model, the expected distance between two datapoints depends on the variance of the distribution. In contrast, the distance between two datapoints from different clusters is a function of the variances of the two distributions *and also the distance between their respective centers*. Therefore, numerous results (e.g. [56, 13, 138, 4]) are based on the assumption that the distance between the centers is sufficiently large – a fact which, combined with the iid assumption and the specific nature of the distribution, allows us to cluster correctly all (or most) points. Further details can be found in Chapter 4.

1.2 Results Overview: Privacy and Clustering

1.2.1 Differential Privacy in Social Networks

We study the notion of differential privacy in social networks. In the simplest of models, a social network is no more than a graph, and one of the most common types of queries regarding a graph is cut-queries: given a set of vertices S, how many edges have one endpoint in S and another in $\bar{S}$? In Chapter 5 we give an algorithm that answers cut-queries while preserving differential privacy w.r.t edge changes. Indeed, for a specific cut $(S, \bar{S})$, the presence or absence of a single edge changes the value of the cut by no more than 1, so the classical technique of adding a small random noise allows us to answer a single cut-query fairly accurately (while preserving differential privacy). The problem lies in answering multiple, and perhaps even all, cut-queries. Our technique essentially publishes a perturbed Laplacian of the graph, which w.h.p gives fairly accurate answers to many cut-queries simultaneously. This result is based on the work in [41].

In addition, we also study the notion of differential privacy w.r.t vertex-changes, i.e. a vertex can change its own attributes or the set of edges adjacent to it. Here, the problem lies in answering even a single query, as most queries have very high global sensitivity (as large as n). In Chapter 6 we propose the notion of restricted sensitivity, where we act as though the set of all possible inputs is restricted to a certain set, specified by the user. Our technique answers the query in a way that always preserves differential privacy, and should the network belong to a user-specified subset of potential networks then our answer is also fairly accurate – its noise is only proportional to the query's restricted sensitivity

rather than its global sensitivity. This result is based on the work in [42].

1.2.2 Clustering

In Chapter 7 we consider finite metric k-median instances and Euclidean k-means instances, and we improve on the notion of stability of Ostrovsky et al [121]. Ostrovsky et al study instances in which the ratio between the cost of the optimal $(k-1)$-means solution and the cost of the optimal k-means solution is at least $\max\{100, 1/\epsilon^2\}$, and give a $(1+O(\epsilon))$-approximation to the k-means optimum for such instances. Our work decouples the strength of the assumption from the quality of the result: we show that under the assumption that the above mentioned ratio is only $1+\alpha$, one can get a $1+\epsilon$ approximation to the k-means optimum, in time $\mathrm{poly}(n,k)\exp(1/\alpha, 1/\epsilon)$. This also holds for instances satisfying the same property w.r.t the k-median objective. We also build on the work of Balcan et al [25] that investigate the connection between point-wise approximations of the target clustering and a c-approximation for the k-median or k-means problem. In fact, our work unifies the Ostrovsky et al assumption and the Balcan et al assumption provided all clusters have sufficiently many datapoints. This result is based on the work in [20].

In Chapter 8 we consider clustering instances for which the optimal solution is optimal under the given metric, and also under any bounded multiplicative perturbation of the given distances. This model, proposed initially by Bilu and Linial [40], is motivated by the fact that in practice, distances between data points are typically the result of some heuristic measure (e.g., edit-distance between strings or Euclidean distance in some feature space) rather than true semantic distance between objects. Thus, the optimal solution should be correct or nearly so on small perturbations of the given distances as well. Bilu and Linial discuss Max-Cut instances satisfying this property up to a fairly large multiplicative error of $O(\sqrt{n})$. In [21] we show to correctly cluster k-median and k-means instances that are $O(1)$-perturbation resilient (specifically, 3-perturbation resilient). This result is based on the work in [21].

In Chapter 9 we improve on the work of Kumar and Kannan [104], that aims to unify many works in the mixture model. Kumar and Kannan pose a specific deterministic condition on a dataset, and show how to correctly cluster instances that satisfy this condition. Then they show that many of the previously studied mixture models, including the Planted Partition model [110] and the work of Ostrovsky et al [121] satisfy w.h.p. this condition. Our work improves on their condition and simplifies their analysis. We pose a condition only on the distances between any two clusters' centers, and using the simple triangle- and Markov-inequalities, we show how to cluster correctly most of the datapoints. This result is based on the work in [22].

1.3 Explicit vs Implicit Assumptions

As mentioned above, an alternative view of this thesis is the study of explicit and implicit assumptions as a specific case of non-worst case analysis. The works which compose this thesis do not give an algorithm that works for *any* graph or *any* clustering instance. Instead, in each chapter of this thesis we *assume* that the input satisfies some property which allows us to propose an algorithm that returns a meaningful output.

In this thesis we differentiate between two such assumptions: explicit and implicit assumptions. Of course, the distinction between the two types isn't formal or rigorous, but rather a helpful way of thinking about the different nature of assumptions. *Explicit assumptions* give an exact and quantifiable description of the input. They are best illustrated by examples: datapoints reside in a Euclidean space, points are sampled iid from a Gaussian, or matrices that have singular values greater than a specific threshold. In addition, explicit assumptions are often checkable – one can check the singular values of the given matrix, or find (approximate) cluster centers and see whether they are sufficiently far apart.

Implicit assumptions are different. They do not discuss the exact nature of the input, but rather propose a property that the input should satisfy. Such property is often related to some "real-life" or outside meaning, typically how the data was collected or how the solution will be used. Such properties are best illustrated in relation to clustering. The work of Ostrovsky et al [121] assumes a lower bound on the ratio between the optimal k-means solution and the optimal $(k-1)$-means solution, using the justification that "if a near-optimal k-clustering can be achieved by a partition into fewer than k clusters, then that smaller value of k should be used to cluster the data". The work of Balcan et al [25] assumes that applying a c-approximation for the k-means objective (for example) yields a clustering whose labeling errs on no more than a δ-fraction of the datapoints, where the error is measured w.r.t to some (unknown and predefined) target clustering, a property which they call (c, δ)-stability. Verifying whether such properties hold might not be tractable.

In this thesis, the works [41, 22] deal with explicit assumptions: in [41] we assume a lower bound on the singular values of the input, and in [22] we assume an exact separation bound between cluster centers. In contrast, the works [20, 21] deal with implicit assumptions: in [20] we unify both the assumption of Ostrovsky et al [121] and of Balcan et al [25], by a property we call *weak-deletion stability*, and in [20] we examine clustering instances that are perturbation resilient, like in Bilu and Linial [40]. The work of [42] lies somewhere in the middle, between explicit and implicit: we allow the querier to specify *any* property she believe the input satisfies, thus converting the querier's implicit assumption to an explicit one for our algorithm. A schematic view of the devision of works in this thesis into subject as well as whether they are based on implicit and explicit assumptions

is given in Figure 1.1.

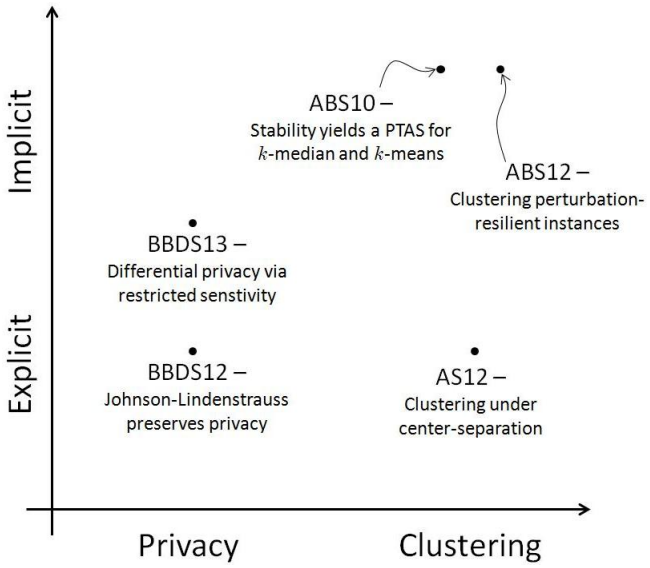


Figure 1.1: Which works in this thesis deal with privacy and which with clustering; which are based on explicit assumptions and which are based on implicit assumptions.

1.4 Acknowledgments

The results of Chapters 5 and 6, published in FOCS 2012 and ITCS 2013 respectively, are joint work with Jeremiah Blocki, Avrim Blum and Anupam Datta. We thank Moritz Hardt for bringing Weilandt's observation (detailed in Chapter 2.2.3) to our attention, and Sofya Raskhodnikova and Adam Smith for helping us to compare our work to theirs.

The results of Chapters 7 and 8, published in FOCS 2010 and in the Journal of Information Processing Letters (Vol.112, 2012) respectively, are joint work with Pranjal Awasthi and Avrim Blum.

The results in Chapter 9, published in APPROX 2012, are joint work with Pranjal Awasthi. We thank Avrim Blum for multiple helpful discussions; Amit Kumar for clarifying a certain point in the original Kumar and Kannan paper; and the anonymous referee for a clarifying discussion about the result of Achlioptas and McSherry.

Chapter 2

Notation and Technical Background

Disclaimer. *This chapter details important notations and classic results in linear algebra and probability we use throughout this entire thesis. This chapter however, does not contain any specific background regarding any of the themes of this thesis: privacy, clustering and various cases detailing non-worst-case analyses.* ***We therefore advise the reader to skip this chapter and return to it upon stumbling onto a non-familiar notation or theorem.*** *However, for clarity, and in order to avoid the use of ill-defined notation, we start with the technical background before moving into the more specific background detailed in Chapters 3 and 4.*

2.1 Graphs

A graph is a pair of sets $G = (V, E)$ where V is a set of nodes, and $E \subset \binom{V}{2}$ is a set of edges. We denote $n = |V|$ and say that the size of G is n. On occasion we discuss weighted graphs for which there exists a weight function $w : E \to \mathbb{R}_{\geq 0}$. In the integral case, we identify a non edge with an edge of weight zero, and an edge with weight 1.

The adjacency matrix of a graph is a symmetric matrix in $\mathbb{R}^{n \times n}$ with its (i, j)-entry equal $w(i, j)$ (and diagonal entries of 0). Given an ordering of the graph's nodes, we define the edge matrix of a graph G as a matrix $E_G \in \mathbb{R}^{\binom{n}{2} \times n}$ where for every pair of distinct vertices, $a < b$ there exists a row in which the only non-zero coordinates are the coordinates of a and b, and we have

$$(E_G)_{(a,b),a} = \sqrt{w(a, b)}, \qquad (E_G)_{(a,b),b} = -\sqrt{w(a, b)}$$

We define the (unnormalized) Laplacian of a graph G as $L_G = E_G^{\intercal} E_G$. Simple calcu-

lation give that for every node a we have the corresponding diagonal entry $(L_G)_{a,a} = \sum_{b \neq a} w(a,b)$, and for every distinct nodes a, b we have $(L_G)_{a,b} = -w(a,b)$. Given a nonempty strict subset of the nodes $S \subsetneq V$,[1] we call the partition $(S, \bar{S})$ of V as a S-cut, and we say its size is $\Phi_G(S) = \sum_{a \in S, b \notin S} w(a,b)$. Given $S \subset V$ we denote its indicator vector as $\mathbf{1}_S \in \{0,1\}^n$ where for every i the coordinate $(\mathbf{1}_S)_i = 1$ iff $i \in S$. Simple calculation shows that for every non-empty $S \subsetneq V$ it holds that

$$\Phi_G(S) = \|E_G \mathbf{1}_S\|^2 = \mathbf{1}_S^\intercal L_G \mathbf{1}_S^\intercal$$

2.2 Linear Algebra

2.2.1 SVD

Given a $m \times n$ matrix M its Singular Value Decomposition (SVD) is $M = U \Sigma V^\intercal$ where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ are unitary matrices, and Σ has non-zero values only on its main diagonal. Furthermore, there are exactly $rank(M)$ positive values on the main diagonal, denoted $\sigma_1(M) \geq \ldots \geq \sigma_{rank(M)}(M)$, called the *singular values*. This allows us to write M as the sum of $rank(M)$ rank-1 matrices: $M = \sum_{i=1}^{rank(M)} \sigma_i u_i v_i^\intercal$. Because Σ has non-zero values only on its main diagonal, the notation Σ^i denotes a matrix whose non-zero values lie only on the main diagonal and are $\sigma_1^i(M), \sigma_2^i(M), \ldots, \sigma_{rank(M)}^i(M)$. Using the SVD, it is clear that if M is of full-rank, then $M^{-1} = V \Sigma^{-1} U^\intercal$, and that if $n = m = rank(M)$ then $\det(M) = \prod_{i=1}^n \sigma_i(M)$. Furthermore, even when M is not full-rank, the SVD allows us to use similar notation to denote the generalizations of the inverse and of the determinant: The Moore-Penrose inverse of M is $M^\dagger = V \Sigma^{-1} U^\intercal$; and the pseudo-determinant of M is $\tilde{\det}(M) = \prod_{i=1}^{rank(M)} \sigma_i(M)$.

2.2.2 Positive Semidefinite Matrices

A $n \times n$ symmetric matrix is called *positive semidefinite* (PSD) if it holds that $x^\intercal M x \geq 0$ for every $x \in \mathbb{R}^n$. Given two PSDs M and N we denote the fact that $(N - M)$ is PSD by $M \preceq N$. For further details regarding the many properties of PSD matrices, see [86].

Fact 2.1. *Let A and B be two PSD matrices s.t. $ker(A) = ker(B)$. If for every x we have that $x^\intercal A x \leq x^\intercal B x$, then for every x it holds that $x^\intercal A^\dagger x \geq x^\intercal B^\dagger x$. Symbolically, $A \preceq B \Rightarrow B^\dagger \preceq A^\dagger$.*

[1] Throughout this thesis, we use $A \subset B$ to denote "B contains A", and $A \subsetneq B$ to denote "B strictly contains A".

Proof. The proof starts with the following claim. Let M be a positive-semidefinite matrix. If $x^\intercal Mx \geq x^\intercal x$ for every $x \in (Ker(M))^\perp$ then it also holds that $x^\intercal M^\dagger x \leq x^\intercal x$ for every $x \in (Ker(M))^\perp$.

Denote the SVD of $M = V\Sigma^2 V^\intercal = \sum_{i=1}^r \sigma_i^2 v_i v_i^\intercal$, where v_i is the i-th column of V. Fix $x \in (Ker(M))^\perp$ and observe that x is span by the same r vectors $\{v_1, v_2, \ldots, v_r\}$, so we can write $x = \sum_{i=1}^r \alpha_i v_i$. Denote $y = V\Sigma^{-1}V^\intercal x = \sum_{i=1}^r \sigma_i^{-1} v_i v_i^\intercal x$. We have that $y = \sum_{i=1}^r \alpha_i \sigma_i^{-1} v_i$ so $y \in (Ker(M))^\perp$. Therefore $y^\intercal My \geq y^\intercal y$, but $y^\intercal y = x^\intercal M^\dagger x$ and $y^\intercal My = x^\intercal x$.

Have established the claim, we can proceed with the proof. We denote the SVD of A as $A = V\Sigma^2 V^\intercal$ and also $B = W\Pi^2 W^\intercal$. Because we can split any vector x into the direct sum $x = x_0 + x_\perp$ where $x_0 \in Ker(A) = Ker(B)$ and $x_\perp \in (Ker(A))^\perp$, and since we have that the required inequality holds trivially for x_0, then we need to show it holds for $x_\perp$. Given any $z \in (Ker(A))^\perp$, set $y = V\Sigma^{-1}V^\intercal z$. We know that $y^\intercal Ay \leq y^\intercal By$, and therefore

$$z^\intercal z = y^\intercal Ay \leq y^\intercal By = z^\intercal \left(V\Sigma^{-1}V^\intercal W\Pi^2 W^\intercal V\Sigma^{-1}V^\intercal\right) z \stackrel{\text{def}}{=} z^\intercal Cz$$

The above proves that C is a positive semidefinite matrix whose kernel is exactly $Ker(A) = Ker(B)$, and so it follows from the previous claim that $z^\intercal z \geq z^\intercal C^\dagger z$. Let $I|_{Ker(C)\perp}$ be the matrix which nullifies every element in $Ker(C)$, yet operates like the identity on $(Ker(C))^\perp$. One can easily check that $C^\dagger = V\Sigma V^\intercal W\Pi^{-2}W^\intercal V\Sigma V^\intercal$ by verifying that indeed $C^\dagger C = CC^\dagger = I|_{Ker(C)\perp}$. So now, given x we denote $z = V\Sigma^{-1}V^\intercal x$ and apply the above to deduce $x^\intercal B^\dagger x = z^\intercal C^\dagger z \leq z^\intercal z = x^\intercal A^\dagger x$. □

2.2.3 Weyl and Lindskii's Inequality

We describe below two useful inequalities regarding the eigenvalues and singular values of a perturbation of a matrix. That is, we related the eigenvalues of a given matrix A with the eigenvalues of a matrix $B = A + E$.

Lemma 2.2 (Weyl's Inequality)**.** *Let A and B be positive semidefinite matrices s.t. the matrix $E = B - A$ satisfies $x^\intercal Ex \geq 0$ for every x. Then for every $1 \leq i \leq n$ it holds that the ith singular-values satisfy:*

$$sv_i(A) \leq sv_i(B)$$

Weyl's inequality is a corollary of the max-min characterization of the singular values of a matrix.

Theorem 2.3 (Courant-Fischer Min-Max Principle)**.** *For every matrix A and every $1 \leq i \leq n$, the i-th singular value of A satisfies:*

$$sv_i(A) = \max_{S:\dim(S)=i} \min_{x \in S:\ \|x\|=1} \langle Ax, x \rangle$$

Proof of Lemma 2.2. Weyl's inequality is a direct application of the theorem. Let S_A be the i-dimensional subspace s.t. $sv_i(A) = \min_{x \in S_A:\ \|x\|=1} \langle Ax, x \rangle$. For every $x \in S_A$ we have that

$$\langle Ax, x \rangle = \langle Ax, x \rangle + 0 \leq \langle Ax, x \rangle + \langle Ex, x \rangle = \langle Bx, x \rangle$$

so $sv_i(A) \leq \min_{x \in S_A:\ \|x\|=1} \langle Bx, x \rangle$. Thus $sv_i(B) = \max_S \min_{x \in S:\ \|x\|=1} \langle Bx, x \rangle \geq sv_i(A)$. □

Lemma 2.4 (Lindskii's inequality.)**.** *Let A and B be a $n \times n$ symmetric matrix. Denote $E = B - A$. Then for every k and every k indices $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ we have that*

$$\sum_{j=1}^{k} ev_{i_j}(B) \leq \sum_{j=1}^{k} ev_{i_j}(A) + \sum_{i=1}^{k} ev_i(E)$$

Much like Weyl's inequality (Lemma 2.2) follows from the Courant-Fischer Min-Max principle, Lindskii's inequality follows from a generalization of the Courant-Fischer principle.

Theorem 2.5 (Wielandt's Min-Max Principle.)**.** *Let A be a $n \times n$ symmetric matrix. Then for every k and every k indices $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ we have that*

$$\sum_{j=1}^{k} ev_{i_j}(A) = \max_{\substack{S_1 \subset S_2 \subset \ldots \subset S_k \\ \dim(S_j)=i_j}} \min_{\substack{x_j \in S_j: \\ x_j \textit{ orthonormal}}} \sum_{j=1}^{k} \langle Ax_j, x_j \rangle$$

Proof of Lemma 2.4. To prove Lindskii's inequality, fix $i_1 < i_2 < \ldots < i_k$ and let $T_1, T_2, \ldots, T_k$ the subspaces for which

$$\sum_{j=1}^{k} ev_{i_j}(B) = \min_{\substack{x_j \in T_j: \\ x_j \text{ orthonormal}}} \sum_{j=1}^{k} \langle Bx_j, x_j \rangle$$

For every $v_1, v_2, \ldots, v_k$ orthonormal we have that $\sum_{j=1}^{k} \langle Bv_j, v_j \rangle = \sum_{j=1}^{k} \langle Av_j, v_j \rangle + \sum_{j=1}^{k} \langle Ev_j, v_j \rangle \leq \sum_{j=1}^{k} \langle Av_j, v_j \rangle + \sum_{i=1}^{k} ev_i(E)$, so

$$\sum_{j=1}^{k} ev_{i_j}(B) \leq \min_{\substack{x_j \in T_j: \\ x_j \text{ orthonormal}}} \sum_{j=1}^{k} \langle Ax_j, x_j \rangle + \sum_{i=1}^{k} ev_i(E)$$

and clearly

$$\sum_{j=1}^{k} ev_{i_j}(A) = \max_{\substack{S_1 \subset S_2 \subset \ldots \subset S_k \\ \dim(S_j)=i_j}} \min_{\substack{x_j \in S_j: \\ x_j \text{ orthonormal}}} \sum_{j=1}^{k} \langle Ax_j, x_j \rangle \geq \min_{\substack{x_j \in T_j: \\ x_j \text{ orthonormal}}} \sum_{j=1}^{k} \langle Ax_j, x_j \rangle$$

□

In fact, Lindskii's inequality holds also for singular values, and not just eigenvalues. This follows from the Lemma 2.4, and from the following observation of Weilandt. Given a $m \times n$ matrix M, the matrix $N = \begin{pmatrix} 0 & M \\ M^\intercal & 0 \end{pmatrix}$ is symmetric and has eigenvalues which are (in descending order):

$$\left\{ sv_1(A), sv_2(A), \ldots, sv_m(A), 0, 0, \ldots, 0, -sv_m(A), -sv_{m-1}(A), \ldots, -sv_1(A) \right\}$$

2.3 Probability

Throughout this thesis, we make multiple uses of randomness, analysis of random variables, and multiple concentration bounds. We detail here their definition and standard inequalities regarding the sum of multiple independent random variables.

2.3.1 Types of Random Variables

2.3.1.1 Bernoulli Random Variables

Definition 2.6. *A random variable X is said to be a Bernoulli random variable if it takes the value 1 w.p. p and the value 0 w.p. $1-p$.*

The mean of Bernoulli random variable is p and its variance is $p(1-p)$. Naturally, the mean of the sum of n independent Bernoulli random variables with mean p is np, and the variance of the sum is $np(1-p)$. This means that we expect that sum to belong to the range $np \pm \sqrt{np(1-p)}$.

2.3.1.2 Laplace Random Variables

Definition 2.7. *A random variable X $Lap(\mu, \sigma)$ is said to be a Laplace random variable if its* PDF *is*

$$\mathsf{PDF}_X(x) = \frac{1}{2\sigma} e^{-\frac{|x-\mu|}{\sigma}}$$

The mean of X is μ and its variance is $2\sigma^2$. We therefore expect that $X \in \mu \pm \sqrt{2}\sigma$. In fact, standard tail bounds give that for any $1 > \delta > 0$ w.p. $\geq 1 - \delta$ we have $|x - \mu| \leq \sigma \log(1/\delta)$. Indeed,

$$\mathbf{Pr}[|x - \mu| > \sigma \log(1/\delta)] = 2 \int_{y > \sigma \log(1/\delta)}^{\infty} \frac{1}{2\sigma} e^{-y/\sigma} dy = \frac{1}{\sigma} \left(-\sigma e^{-y/\sigma} \right) |_{y=\sigma \log(1/\delta)}^{\infty} = \delta$$

In this thesis, we denote $X \sim Lap(\sigma)$ in case $X \sim Lap(0, \sigma)$.

2.3.1.3 Gaussian Random Variables

Definition 2.8. *A univariate random variable $X \sim \mathcal{N}(\mu, \sigma^2)$ is said to be a Gaussian random variable if its* PDF *is*

$$\mathsf{PDF}_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

The mean of X is μ and its variance is σ^2. X is said to be a normal Gaussian if $X \sim \mathcal{N}(0, 1)$. Classic concentration bounds on Gaussians give that $\mathbf{Pr}[(x - \mu)^2 > \log(2/\delta)\sigma^2] \leq \delta$.

Gaussian random variables abide the *linear combination* rule: for any two i.i.d normal random variables s.t. $X \sim \mathcal{N}(\mu_X, \sigma_X^2)$ and $Y \sim \mathcal{N}(\mu_Y, \sigma_Y^2)$, we have that their linear combination $Z = aX + bY$ is distributed according to $Z \sim \mathcal{N}(a\mu_X + b\mu_Y, a^2\sigma_X^2 + b^2\sigma_Y^2)$. This in turn allows us to identify a random variable $R \sim \mathcal{N}(0, \sigma^2)$ with the random variable $\sigma R'$, where $R' \sim \mathcal{N}(0, 1)$.

The multivariate normal distribution is the multi-dimension extension of the univariate normal distribution. $X \sim \mathcal{N}(\mu, \Sigma)$ denotes a m-dimensional multivariate r.v. whose mean is $\mu \in \mathbb{R}^m$, and variance is the PSD matrix $\Sigma = \mathbf{E}\left[(X - \mu)(X - \mu)^\intercal\right]$. If Σ has full rank (Σ is positive definite) then $\mathsf{PDF}_X(x) = \frac{1}{\sqrt{(2\pi)^m \det(\Sigma)}} \exp(-\frac{1}{2} x^\intercal \Sigma^{-1} x)$, a well defined function. If Σ has non-trivial kernel space then PDF_X is technically undefined (since X is defined only on a subspace of volume 0, yet $\int_{\mathbb{R}^m} \mathsf{PDF}_X(x) dx = 1$). However, if we

restrict ourselves only to the subspace $\mathcal{V} = (Ker(\Sigma))^{\perp}$, then $\mathsf{PDF}_X^{\mathcal{V}}$ is defined over $\mathcal{V}$ and $\mathsf{PDF}_X^{\mathcal{V}}(x) = \frac{1}{\sqrt{(2\pi)^{rank(\Sigma)}\tilde{\det}(\Sigma)}} \exp(-\frac{1}{2}x^{\intercal}\Sigma^{\dagger}x)$. From now on, we omit the superscript from the PDF and refer to the above function as the PDF of X. Observe that using the SVD, we can denote $\Sigma = U\ \mathrm{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_r^2, 0, \ldots, 0)\ U^{\intercal}$, and so $\mathcal{V}$ is the subspace spanned by the first r rows of U. The multivariate extension of the linear combination rule is as follows. If A is a $n \times m$ matrix, then the multivariate r.v. $Y = AX$ is distributed as though $Y \sim \mathcal{N}(A\mu, A\Sigma A^{\intercal})$. For further details regarding multivariate Gaussians see [115].

2.3.2 Concentration Bounds

We conclude this chapter by stating 4 famous inequalities. Their proofs can be found in [12].

Theorem 2.9 (Markov's inequality)**.** *Let X be a non-negative random variable. Then for every $t > 0$ we have that*

$$\mathbf{Pr}[X > t] \leq \frac{\mathbf{E}[X]}{t}$$

Theorem 2.10 (Chebyshev's inequality)**.** *Let X be a random variable with mean μ and variance σ^2. Then for every $t > 0$ we have that*

$$\mathbf{Pr}[|X - \mu| > t] \leq \frac{\sigma^2}{t^2}$$

Theorem 2.11 (Chernoff-Hoeffding bounds)**.** *Let $X_1, X_2, \ldots, X_n$ be iid random variables taking values in the range $[a, b]$ and with mean μ. Then for every $t \geq 0$ we have*

$$\mathbf{Pr}\left[\left|\tfrac{1}{n}\sum_i X_i - \mu\right| > t\right] \leq 2\exp(-2n\tfrac{t^2}{(b-a)^2})$$

and for every $\delta > 0$ we have

$$\mathbf{Pr}\left[\left|\tfrac{1}{n}\sum_i X_i - \mu\right| > \delta\mu\right] \leq 2\exp(-\tfrac{1}{3}n\delta^2\mu)$$

Theorem 2.12 (Azuma's inequality)**.** *Let $X_0, X_1, X_2, \ldots, X_n$ be a sequence of random variables satisfying that for every $i \geq 1$ we have $E[X_i|X_1, \ldots, X_{i-1}] = X_{i-1}$ and that there exists a $c > 0$ s.t. $\mathbf{Pr}[|X_i - X_{i-1}| \leq c] = 1$. Then,*

$$\mathbf{Pr}\left[\tfrac{1}{n}|X_n - X_0| \geq t\right] \leq 2\exp\left(-n\tfrac{t^2}{c^2}\right)$$

Chapter 3

Background: Differential Privacy

Consider a scenario in which a trusted curator gathers personal information from n individuals, and wishes to release statistics about these individuals to the public without compromising any individual's privacy. *Differential privacy* [63] provides a robust guarantee of privacy for such data releases. It guarantees that for any two neighboring databases (databases that differ on the details of any single individual), the curator's distributions over potential outputs are statistically close.

Definition 3.1 ([62])**.** *An algorithm ALG which maps databases into some range $\mathsf{ALG} : \mathcal{D} \to \mathcal{R}$ is said to preserve* (ϵ, δ)-differential privacy *if for all pairs D and D' of databases that differ on the details of a single individual, and for all subsets $\mathcal{S} \subset \mathcal{R}$ it holds that*

$$\mathbf{Pr}[\mathsf{ALG}(D) \in \mathcal{S}] \leq e^{\epsilon}\mathbf{Pr}[\mathsf{ALG}(D) \in \mathcal{S}] + \delta$$

We say ALG preserves ϵ-differential privacy if it preserves $(\epsilon, 0)$-differential privacy.

By itself, preserving differential privacy isn't hard, since the curator's answers to users' queries can be so noisy that they obliterate any useful data stored in the database. Therefore, the key research question in this field is to provide tight *utility and privacy tradeoffs*.

3.1 The Basic Mechanism

The most basic technique that preserves differential privacy and gives good utility guarantees is to add relatively small Laplace or Gaussian noise to a query's true answer. We use $\mathcal{D}$ to denote the set of all possible datasets, where we view a dataset as a multiset of

elements taken from some domain $\mathcal{X}$. We say two datasets $D, D' \in \mathcal{D}$ are *neighbors* if they differ on the details of a single individual. We denote the fact that D' is a neighbor of D using $D' \sim D$. We define the *distance* $d(D, D')$ between two databases $D, D' \in \mathcal{D}$ as the minimal non-negative integer k s.t. there exists a path $D_0, D_1, \ldots, D_k$ where $D_0 = D$, $D_k = D'$ and for every $1 \leq i \leq k$ we have that $D_{i-1} \sim D_i$. Alternatively, we can say that $d(D, D') = k$ if $|D \setminus D'| = k$. Given a subset $\mathcal{D}' \subset \mathcal{D}$ we denote the distance of a database D to $\mathcal{D}'$ as $d(D, \mathcal{D}') = \min_{D' \in \mathcal{D}'} d(D, D')$.

Definition 3.2 ([63]). *The* global sensitivity *of a query function $f : \mathcal{D} \to \mathbb{R}$ is $GS_f = \max_{D \sim D'} |f(D) - f(D')|$.*

For any given statistic or query, its global sensitivity measures the maximum difference in the answer to that query over all pairs of neighboring data sets. More importantly, as the next theorem illustrates, global sensitivity provides an upper bound on the amount of noise that has to be added to the actual statistic in order to preserve differential privacy.

Theorem 3.3 ([63, 62]). *Given a query function $f : \mathcal{D} \to \mathbb{R}$, the mechanism that outputs $f(D) + X$ is*

- *ϵ-differentially private, when $X \sim Lap(GS(f)/\epsilon)$*
- *(ϵ, δ)-differentially privacy, when $X \sim \mathcal{N}(0, GS(f) \cdot 2\log(2/\delta)/\epsilon^2)$.*

The basic mechanism provides good utility and privacy tradeoffs for answering a single query of small global sensitivity. However, this leaves two questions for future research: (i) can we give any reasonable utility guarantees for queries of large global sensitivity, and (ii) how well can we answer multiple queries. We start by address the latter issue, and only after surveying multiple mechanisms with better dependence on the number of queries we will return to the first issue.

It is straight forward to see that applying a (ϵ, δ)-differentially private mechanism in order to answer t queries preserves $(t\epsilon, t\delta)$-differential privacy. However, the following theorem gives better bounds (roughly stated as answering t queries while preserving $\big(\epsilon \cdot O(\sqrt{t}), \delta \cdot O(t)\big)$-differential privacy).

Theorem 3.4 ([65]). *Fix $\epsilon, \delta, \delta' > 0$. Let $M_1, \ldots, M_t$ be t mechanisms that preserve (ϵ, δ)-differential privacy. Then the concatenation of all mechanisms preserves (ϵ^*, δ^*)-differential privacy, for*

$$\epsilon^* = \epsilon \cdot \sqrt{2t \ln(1/\delta')} + t\epsilon(e^\epsilon - 1), \quad \delta^* = t\delta + \delta'$$

The proof of Theorem 3.4 uses the KL-divergence. First, Dwork et al show that given two random variables X, X' s.t. for every $x \in Supp(X)$ it holds $\mathbf{Pr}[X = x] \leq e^{\epsilon}\mathbf{Pr}[X' = x]$, it holds that

$$\mathbf{E}_x \left[\ln \frac{\mathbf{Pr}[X = x]}{\mathbf{Pr}[X' = x]} \right] \leq \epsilon(e^{\epsilon} - 1) \leq 2\epsilon^2$$

Secondly, Dwork et al use Azuma's inequality to show that for $X_1, X_2, \ldots, X_t$ and $X'_1, X'_2, \ldots, X'_t$ s.t. for every i the same holds for X_i and X'_i, we have w.h.p that

$$\left| \sum_{i=1}^{t} \ln \frac{\mathbf{Pr}[X_i = x]}{\mathbf{Pr}[X'_i = x]} - \mathbf{E} \left[\sum_{i=1}^{t} \ln \frac{\mathbf{Pr}[X_i = x]}{\mathbf{Pr}[X'_i = x]} \right] \right| \leq \sqrt{2t \ln(1/\delta')}$$

And finally they show that for every Y, Y' satisfying $\mathbf{Pr}[Y = x] \leq e^{\epsilon}\mathbf{Pr}[Y' = x] + \delta$ there exists X, X' such that $\max\{|\mathbf{Pr}[Y = x] - \mathbf{Pr}[X = x]|, |\mathbf{Pr}[Y' = x] - \mathbf{Pr}[X' = x]|\} \leq \delta$ and that $\mathbf{Pr}[X = x] \leq e^{\epsilon}\mathbf{Pr}[X' = x]$

3.2 Alternative Mechanisms

As Theorem 3.4 suggests, when answering t queries (all with small global sensitivity) by naïvely applying the basic mechanism to answer each, in order to preserve (ϵ, δ)-differential privacy, we need to add random noise of roughly $O(\sqrt{t}/\epsilon)$ to each query. It turns out that there exist other scenarios where one can improve the error's dependency on t to a logarithmic dependency. Below, we detail mechanisms that achieve such logarithmic dependency. As we show, this improvement often comes at the expense of computational efficiency.

3.2.1 The Exponential Mechanism

McSherry and Talwar [112] introduced the exponential mechanism, that preserves ϵ-differential privacy. In fact, it is somewhat of a general scheme, which can be tweaked based on its application.

Theorem 3.5 ([112])**.** *Let $\mathcal{R}$ be a discrete range of desired outputs, and let s be any* scoring-function $s : \mathcal{D} \times \mathcal{R} \to \mathbb{R}$*. Given a database D, let the weight of an element $r \in \mathcal{R}$ be defined as*

$$w(r) = \exp\left(-\tfrac{\epsilon}{2\ GS(s)} s(D, r)\right)$$

Then the mechanism that outputs r w.p. $\propto w(r)$ preserves ϵ-differential privacy.

Proof. Given any neighboring D and D', for every r we have that $e^{-\epsilon/2} \leq w_D(r)/w_{D'}(r) \leq e^{\epsilon/2}$. Therefore, we have that for every r

$$\frac{\mathbf{Pr}[r|\ D]}{\mathbf{Pr}[r|\ D']} = \frac{w_D(r)}{w_{D'}(r)} / \frac{\sum_{r\in\mathcal{R}} w_D(r)}{\sum_{r\in\mathcal{R}} w_{D'}(r)} \leq e^{\epsilon/2+\epsilon/2}$$

□

McSherry and Talwar applied the exponential mechanism to problems in mechanism design (after pointing out to the fact that applying an ϵ-differential privacy mechanism to a mechanism-design problem guarantees ϵ-truthfulness). Their work was the first to explore the connection between privacy and mechanism design, a rapidly growing field nowadays (see survey by Pai and Roth [122]). But the exponential mechanism allows for much more – using the exponential mechanism, Blum et al [43] have shown that there exists an algorithm (non necessarily efficient) to privately release a sanitized dataset that answers a very large set of queries $\mathcal{Q}$ simultaneously.

Theorem 3.6 ([43])**.** *Let D be a dataset whose elements come from a domain $\mathcal{X}$. Let $\mathcal{Q}$ be a predefined set of functions $q : \mathcal{X} \to \{0,1\}$. For every $q \in \mathcal{Q}$, let $f_q : D \to [0,1]$ be the corresponding fractional counting query $f_q(D) = \frac{1}{|D|}|\{x \in D;\ q(x) = 1\}|$. Fix $1/2 > \alpha, \beta > 0$, and define $m = \frac{20\log(1/\alpha)}{\alpha^2}\dim_{VC}(\mathcal{Q})$.*

Then for any $\epsilon > 0$, applying the exponential mechanism over the range $\mathcal{R} = \mathcal{X}^m$ of databases of size m, using the scoring function $s(D,\hat{D}) = \max_{q\in\mathcal{Q}}\left|f_q(D) - f_q(\hat{D})\right|$ outputs a database of size m s.t. with probability $\geq 1-\beta$ it holds that for every $q \in \mathcal{Q}$ we have that $\left|f_q(D) - f_q(\hat{D})\right| < \alpha$, provided that

$$|D| \geq \frac{4}{\alpha\epsilon}\left(\frac{\log(1/\alpha)}{\alpha^2}\dim_{VC}(\mathcal{Q})\ln(|\mathcal{X}|) + \ln(1/\beta)\right)$$

Proof. Preserving ϵ-differential privacy is a consequence of the exponential mechanism. To prove utility, observe first that $GS(s) = 1$. Observe also that standard techniques regarding the VC-dimension guarantee the existence of a dataset D^* for which the score is $\leq \alpha/2$. Hence, the ratio of the weight of this one good dataset to the total weight of all bad datasets (for which the score $> \alpha$) can be bounded using the size of D by

$$\frac{\mathbf{Pr}[D^*]}{\mathbf{Pr}[\text{Bad dataset}]} \geq \frac{\exp(-\epsilon\alpha|D|/4)}{|\mathcal{X}|^m\exp(-\epsilon\alpha|D|/2)} = \exp\left(\tfrac{\epsilon\alpha}{4}|D| - m\log(|\mathcal{X}|)\right) \geq \frac{1}{\beta} \geq \frac{1-\beta}{\beta}$$

□

The result of [43] was the first theoretical work to show the existence of non-interactive privacy preserving mechanisms (though the data-mining community have used the technique of Randomize-Response for privacy, as we discuss below). Such mechanism have the benefit that they do not need to interact with a querier and can simply release information that will answer any of the querier's potential questions. However, the obvious problem with the exponential mechanism is its intractability – in most reasonable applications $\mathcal{X}^m$ is not poly-time tractable. In fact, there have been several works [64, 136, 135] proving that for the case of $\mathcal{X} = \{0,1\}^d$, a sanitized dataset cannot be output in poly-time under certain cryptographic assumptions.

3.2.2 Randomized Response

The simple technique of decentralized input perturbation, or Randomized-Response [141], gives a straight-forward way of preserving ϵ-differential privacy.

Theorem 3.7. *Let D be a database whose elements are taken from a finite-size domain $\mathcal{X}$ of size T. Given $0 < \epsilon < 1$, the algorithm that for every entry $x \in D$ (recall, D is a multiset) picks $y(x)$ independently where*

$$y(x) = \begin{cases} x, & \text{w.p. } \frac{1+\epsilon}{T+\epsilon} \\ x', & \text{w.p. } \frac{1}{T+\epsilon}, \text{ for any } x' \neq x \end{cases}$$

and publishes $\hat{D} = \{y(x) : \ x \in D\}$, is a ϵ-differential privacy preserving mechanism.[1] *Furthermore, fix $S \subset \{1, 2, \ldots, n\}$ and define the multiset $D|_S = \{D_i : \ i \in S\}$. Then w.p. $\geq 1 - \beta$ we can use $\hat{D}$ to estimate the fraction of elements in $D|_S$ that are of type x up to an additive error of $O\left(\frac{T}{\epsilon}\sqrt{\frac{\log(T/\beta)}{|S|}}\right)$.*

Proof. The fact that the above-mentioned algorithm preserves ϵ-differential privacy is straight-forward. Given two neighboring datasets D and D' that differ on the i-th entry, it is easy to see that for any $y \in \mathcal{X}$ we have

$$\frac{\mathbf{Pr}[\hat{D}_i = y|\ D]}{\mathbf{Pr}[\hat{D}_i = y|\ D']} \leq 1 + \epsilon \leq e^{\epsilon}$$

Now, fix S. For every $x \in \mathcal{X}$, we denote ρ_x as the fraction of entries in $D|_S$ that are of type x. A straight-forward calculation gives that in $\hat{D}$ we expect to see a fraction of $\mu_x = \frac{1+\epsilon\rho_x}{T+\epsilon}$

[1] A slightly less elegant version picks $y(x) = x$ w.p. $\frac{e^{\epsilon}}{T-1+e^{\epsilon}}$ and $y(x) = x'$ for any other $x' \neq x$ w.p. $\frac{1}{T-1+e^{\epsilon}}$.

entries of type x. Due to standard Hoeffding and union bounds it holds that w.p. $\geq 1-\beta$ the number of elements in $\hat{D}|_S$ of type x, denoted n_x, satisfies $|\frac{1}{|S|}n_x - \mu_x| \leq O(\sqrt{\frac{\log(2T/\beta)}{2|S|}})$ for all $x \in \mathcal{X}$. Therefore, using the estimation

$$\hat{\rho}_x = \frac{1}{\epsilon}\left((T+\epsilon)\frac{n_x}{|S|} - 1\right)$$

we have that $|\rho_x - \hat{\rho}_x| \leq \frac{T+\epsilon}{\epsilon}\sqrt{\frac{\log(2T/\beta)}{2|S|}}$. □

Observe that Randomized-Response allows us to estimate queries that are based on the histogram induced by D on multiple predefined subsets of entries $S_1, S_2, \ldots, S_t$ (which we could think of as t predefined queries), with the error bound on each estimation scaling like $O(\sqrt{\frac{\log(t/\beta)}{\min_i |S_i|}})$. I.e., the error bound grows by only a factor of $\log(t)$ compared to the bound we have for answering a single query. However, this technique provides vacuous error bounds when $T = |\mathcal{X}|$ is large. Furthermore, given a fixed S, the error bounds provided by the Randomize-Response mechanism are inferior to the bounds provided by the mechanism that perturbs the histogram of $D|_S$ by adding $Lap(1/\epsilon n)$ noise to the count of each type independently. However, the main benefit of the Randomized-Response algorithm is that it is a *decentralized* algorithm – each individual randomizes her own data and provide the data-curator with the perturbed data (as opposed to previous mechanisms, in which the data-curator gets to observe the actual data but releases it with some noise). Therefore, the Randomized-Response mechanism is applicable in the case that the agents don't trust even the data-curator.

The Randomized-Response mechanism was first used for data-mining, actually applied for voting for one of two options w.p. $\{2/3, 1/3\}$. It was first analyzed theoretically by Kasiviswanathan et al [101], and was also used for privately estimating the size of cuts in a graph by Gupta et al [77].

3.2.3 The Multiplicative Weights Mechanism

Lastly, we conclude the survey of existing mechanisms with the Multiplicative Weights mechanism of Hardt and Roth [80]. This mechanism builds on the previous work of Roth and Roughgarden [128], who were the first to consider the case of queries as linear operators. In the Multiplicative Weights mechanism, the database is viewed as a histogram over $\mathcal{X}$ (normalized to 1, for convenience), so $\mathcal{D} = \{v \in \mathbb{R}_{\geq 0}^{|\mathcal{X}|} : \sum_i v_i = 1\}$. Queries are linear

queries in $[0,1]^{|\mathcal{X}|}$, and so the answer to a query q is $\langle q, D\rangle$. Clearly, for any q, the global sensitivity of q is upper bounded by $1/n$.

Algorithm 1: Multiplicative Weights Mechanism

Input: A n-size database D, privacy parameters: $\epsilon, \delta > 0$, failure probability $\beta > 0$, number of overall queries to answer: t.

Set parameters: $U = O\left(\epsilon n \frac{\sqrt{\log(|\mathcal{X}|)}}{\log(1/\delta)\log(t/\beta)}\right)$, $\sigma = O\left(\sqrt{\frac{\sqrt{\log(|\mathcal{X}|)}}{\epsilon n} \cdot \frac{\log(1/\delta)}{\log(t/\beta)}}\right)$,
$\alpha = O\left(\sigma \log(t/\beta)\right) = O\left(\sqrt{\frac{\sqrt{\log(|\mathcal{X}|)}}{\epsilon n}} \log(1/\delta)\log(t/\beta)\right)$.

Initialize vector $v = \frac{1}{n}\mathbf{1}$, and $i \leftarrow 0$.

foreach *user query* q **do**

  Sample $Z \sim Lap(\sigma)$.

  if $|\langle q, D\rangle + Z - \langle q, v\rangle| < \alpha/2$ **then**

    Answer q with $\langle q, v\rangle$. `// Easy case`

  else

    $i \leftarrow i + 1$. Abort if $i > U \cdot \log(1/\delta)$.

    Answer q with $\langle q, D\rangle + Z$. `// Hard case`

    if $\langle q, D\rangle + Z < \langle q, v\rangle$ **then**

      Let v' be a normalized histogram satisfying that $\forall i$ we have $v'_i \propto v_i \exp\left(-\frac{\alpha}{2n} q_i\right)$.

    else

      Let v' be a normalized histogram satisfying that $\forall i$ we have $v'_i \propto v_i \exp\left(-\frac{\alpha}{2n}(1 - q_i)\right)$.

    Update $v \leftarrow v'$.

Theorem 3.8 ([80, 77])**.** *Algorithm 1 preserves (ϵ, δ)-differential privacy, and w.p. $\geq 1-\beta$ answers all user queries up to an additive error of α.*

The analysis of the Multiplicative Weight mechanism is quite intricate (and it was later improved and generalized by Gupta et al [77]). It's main outline is as follows.

1. W.p. $1 - \beta$ we have that in no query $|Z| > \alpha/4$.
2. Fix two neighboring datasets, D and D'. Given a query q, we partition $[-\alpha/4, \alpha/4]$ into 2 regions: "safe" in which we update neither D nor D'; and its complimentary "non-safe". We further partition the "non-safe" region into "must-update" – in which we update both D and D', and its complimentary "unknown".

3. Conditioned on Z being "safe", *there is no privacy loss*. This is the crux of the analysis.

4. Conditioned on Z being "non-safe", then there's a constant probability Z is a "must-update". Conversely, w.p. $> 1 - \delta/2$ we have that the number of times in which Z is "non-safe" is at most $\#\mathrm{Updates} \cdot O(\log(1/\delta))$.

5. Since v is update using the standard Multiplicative Weights algorithm [108], then after U updates we have that $|\langle q, v\rangle - \langle q, D\rangle| < \alpha/2$ for every q. So at that point, no more updates are made. Therefore, we answer all t queries with an error of at most α.

6. Finally, we appeal to the standard composition arguments (Theorem 3.4) and Azuma's inequality to show that in the $O(U \log(1/\delta))$ times in which Z is "non-safe" and we applied the Laplace mechanism with std σ incurred an overall privacy loss of at most ϵ w.p. $\geq 1 - \delta/2$.

It is obvious that the counting queries setting considered by Blum et al [43] is a private case of the Multiplicative Weights mechanism: Given a query $q \in Q$ we can identify it with the vector $v_q \in \{0,1\}^{|\mathcal{X}|}$ where $v_q(x) = q(x)$ for every $x \in \mathcal{X}$, and now the counting query is precisely $\langle v_q, D\rangle$. And so, Blum et al [43] give a ϵ-differentially private mechanism that can answer each query up to an additive error of $1/(\epsilon n)^{1/3}$ (neglecting other parameters). In contrast, the Multiplicative Weights mechanism gives a (ϵ, δ)-differentially privacy mechanism (a weaker guarantee) with an additive error of $1/\sqrt{\epsilon n}$ (a tighter utility bound). The alternative analysis of the Multiplicative Weights mechanism given by Hardt et al [81] shows that as an ϵ-differentially private algorithm, it gives error bounds of $O(1/(\epsilon n)^{1/3})$. It is an open question to see whether a ϵ-differentially private algorithm exists whose utility guarantee is proportional to $1/\sqrt{n}$.

Observe also that the Multiplicative Weights mechanism is often intractable – since answering each query takes $\mathrm{poly}(|\mathcal{X}|)$ time. But even in applications where $\mathcal{X}$ has poly-size, if we wish to answer a large set of queries $\mathcal{Q}$ then the mechanism requires us to traverse all queries in $\mathcal{Q}$ and so it runs in time proportional to $|\mathcal{Q}|$. And yet – the analysis of the Multiplicative Weights mechanism guarantees the existence of a small (poly-size) number of queries for which you update the vector v and afterward it is never again updated. Therefore, one potential venue for improving its running time is to find this small set of queries, apply the mechanism solely on them, and release the resulting v. This is precisely what Hardt et al [81] do, by picking queries according to the exponential mechanism (since the set of queries you select for the update might leak privacy as well). Needless to say, applying the exponential mechanism over the queries in $\mathcal{Q}$ takes $\mathrm{poly}(|\mathcal{Q}|)$ time. It is an

open question of finding a scenario (say, cut-queries) where we can find this small-size set of update-queries efficiently.

3.2.4 Our Contribution: The Johnson Lindenstrauss Mechanism

In Chapter 5 we detail a different mechanism that allows one to answer multiple queries, with error bound growing only logarithmically with the number of queries. This mechanism is actually a well-known one: the Johnson-Lindenstrauss transform. We show that given a database represented as a $n \times d$ matrix A, the mechanism that picks a random matrix R of size $r \times n$ in which every entry is chosen independently from a normal Gaussian and publishes the multiplication RA, preserves (ϵ, δ)-differential privacy, *if all singular values of A are sufficiently large*. Furthermore, the Johnson-Lindenstrauss transform guarantees that for any query vector x we have that w.h.p $\|Ax\|^2 \approx \frac{1}{r}\|RAx\|^2$. Therefore, this mechanism allows us to answer variance queries – where the user poses a direction (unit vector) x and queries about the variance of the data along direction x. In the special case where A is the edge-matrix of a graph, this mechanism allows the user to estimate the values of cuts – the user specifies a set of vertices S and gets an estimation of the value of the cut $|E(S, \bar{S})|$.

3.3 Smooth Sensitivity: Answering Queries of Large Global Sensitivity

Recall that the basic mechanism (Theorem 3.3) gives good utility guarantees only for queries of small global sensitivity. In contrast, when the global sensitivity is large (for example, in the extreme case where there exists $D_1 \sim D_2$ such that $f(D_1) = \min_{D \in \mathcal{D}} f(D)$ and $f(D_2) = \max_{D \in \mathcal{D}} f(D)$), this mechanism may have vacuous utility guarantees. Still, there could be instances $D \in \mathcal{D}$ for which the *local* sensitivity is small.

Definition 3.9. *The* local sensitivity *of a query f at a dataset D is* $LS_f(D) = \max_{D' \sim D} |f(D) - f(D')|$.

The local sensitivity $LS_f(D)$ may be significantly lower than the global sensitivity GS_f. However, adding noise proportional to $LS_f(D)$ does not preserve differential privacy because the noise level itself may leak information. A clever way to circumvent this problem is to smooth out the noise level [119].

Definition 3.10 ([119]). *A β-smooth upper bound on the local sensitivity of a query f is a function $S_{f,\beta}$ which satisfies (i) $\forall D \in \mathcal{D},\ S_{f,\beta}(D) \geq LS_f(D)$, and (ii) $\forall D, D' \in \mathcal{D}$ it holds that $S_{f,\beta}(D) \leq \exp(\beta \cdot\ d(D, D'))S_{f,\beta}(D')$.*

Indeed, it is possible to preserve privacy while adding noise proportional to a β-smooth upper bound on the sensitivity of a query.

Theorem 3.11 ([119]). *Given a query function $f : \mathcal{D} \to \mathbb{R}$, the mechanism that for a given D outputs $f(D) + Z$ with $Z \sim Lap(\frac{2S_{f,\beta}(D)}{\epsilon})$ and with $\beta = \frac{\epsilon}{2\ln(2/\delta)}$ preserves (ϵ, δ)-differential privacy.*

Proof. Fix any two neighboring datasets D and D'. Let Z_1 be random noise sampled from $Lap(\frac{2S_{f,\beta}(D)}{\epsilon})$ and Z_2 be random noise sampled from $Lap(\frac{2S_{f,\beta}(D')}{\epsilon})$. Calculating the ratio between the two PDFs, we have

$$\begin{aligned}\frac{\mathsf{PDF}_{Z_1}(x)}{\mathsf{PDF}_{Z_2}(x)} &= \frac{S_{f,\beta}(D')}{S_{f,\beta}(D)} \exp\left(-\frac{\epsilon|x|}{2}\left(\frac{1}{S_{f,\beta}(D)} - \frac{1}{S_{f,\beta}(D')}\right)\right) \\ &= \frac{S_{f,\beta}(D')}{S_{f,\beta}(D)} \exp\left(\frac{\epsilon|x|}{2S_{f,\beta}(D')}\left(1 - \frac{S_{f,\beta}(D')}{S_{f,\beta}(D)}\right)\right)\end{aligned}$$

Now, if it is the case that $1 \leq \frac{S_{f,\beta}(D')}{S_{f,\beta}(D)} \leq e^{\beta}$ then it is simple to see that

$$\frac{\mathsf{PDF}_{Z_1}(x)}{\mathsf{PDF}_{Z_2}(x)} \leq e^{\beta}e^{0} \leq e^{\epsilon/2}$$

So we can assume that it is that case that $e^{-\beta} \leq \frac{S_{f,\beta}(D')}{S_{f,\beta}(D)} \leq 1$ in which case we have

$$\frac{\mathsf{PDF}_{Z_1}(x)}{\mathsf{PDF}_{Z_2}(x)} \leq 1 \cdot \exp\left(\frac{\epsilon(1-e^{-\beta})}{2S_{f,\beta}(D')}|x|\right) \leq \exp\left(\frac{\epsilon\beta}{2S_{f,\beta}(D')}|x|\right)$$

Because of the definition of β we have that w.p. $1 - \delta/2$ it holds that $|Z_2| \leq S_{f,\beta}(D')/\beta$, so w.p. $\geq 1 - \delta/2$ we have that the ratio of the two PDFs is bounded by

$$\frac{\mathsf{PDF}_{Z_1}(x)}{\mathsf{PDF}_{Z_2}(x)} \leq e^{\epsilon/2}$$

So now fix any $S \subset \mathbb{R}$ and we have

$$\mathbf{Pr}[f(D) + Z_1 \in S] = \mathbf{Pr}[Z_1 \in S - f(D)] \ \leq e^{\frac{\epsilon(f(D)-f(D'))}{2S_{f,\beta}(D)}} \mathbf{Pr}[Z_1 \in S - f(D')]$$

$$\leq e^{\epsilon/2}\mathbf{Pr}[Z_1 \in S - f(D')]$$
$$\leq e^{\epsilon/2}\left(e^{\epsilon/2}\mathbf{Pr}[Z_2 \in S - f(D')] + \tfrac{\delta}{2}\right)$$
$$\leq e^{\epsilon}\mathbf{Pr}[f(D') + Z_2 \in S] + \delta$$

□

Clearly, to compute the output of this mechanism efficiently one must present a tractable algorithm to compute the β-smooth upper bound $S_{f,\beta}(G)$, a task which is by itself often non-trivial.

3.3.1 Our Contribution: Restricted Sensitivity

In Chapter 6 we introduce the notion of *restricted sensitivity*, which is complementary to the notion of smooth-sensitivity. Smooth sensitivity depends on the input database – given D, the answer to a user's query f depends on the values f takes on D and the neighborhood of D. In contrast, restricted sensitivity depends on a user-defined set of "good" instances $\mathcal{H} \subset \mathcal{D}$, so that given a query f its answer depends on the values f takes on $\mathcal{H}$. We design a (ϵ, δ)-differentially private mechanism that given f answers it with fairly small noise if it is indeed the case that $D \in \mathcal{H}$, however, this mechanism has no guarantees if it is the case that $D \notin \mathcal{H}$.

We illustrate the mechanism in a particular setting which is of growing importance nowadays – answering queries regarding social networks. Many natural questions one might ask given a social network have large global sensitivity or even large local sensitivity. (E.g. the answer to the question "how many people are friends with a person from CMU" may change significantly if Justin Beiber relocates to Pittsburgh.) We illustrate the use of applying the restricted sensitivity mechanism to such queries focusing on the set $\mathcal{H}_k$ of social networks in which each person has at most k friends. Indeed, whereas the global sensitivity of profile queries is $\Omega(n)$, their restricted sensitivity over $\mathcal{H}_k$ is only $O(k)$. (Continuing with our example – focusing only on the subnetwork of computer scientists, which are naturally inclined to have a limited number of friends, using restricted sensitivity we can find out in a differentially private manner whether people from CMU are friendlier than the average person.)

Chapter 4

Background: Center-Based Clustering

Problems of clustering data arise in a wide range of different areas – clustering proteins by function, clustering documents by topic, and clustering images by who or what is in them, just to name a few. Generally speaking, the goal of clustering is to partition n given data objects into k groups that share some commonality. From a machine learning perspective, the goal of clustering is to label each datapoint with one of k distinct labels. Yet, as opposed to the classical machine-learning labeling tasks which are often done using access to a few labeled examples, clustering is often done in a non-supervised setting – i.e., without viewing the labels of any of the datapoints. Instead, we are given access to a metric over the given dataset S.

Definition 4.1. *A non-negative function $d : S \times S \to \mathbb{R}_{\geq 0}$ is called a* metric *if it satisfies the following 3 properties.*

- *(Reflexivity) For any $x, y \in S$ it holds that $d(x, y) = 0$ iff $x = y$.*
- *(Symmetry) For any $x, y \in S$ it holds that $d(x, y) = d(y, x)$.*
- *(Triangle inequality) For any $x, y, z \in S$ it holds that $d(x, z) \leq d(x, y) + d(y, z)$.*

4.1 Center-Based Clustering Objectives

Operationally, clustering is often performed by optimizing some natural objective over the given metric. The focus of this thesis is on center-based objectives, such as the popular k-median, k-means and k-centers, in which we measure a k-partition by choosing a special

point for each cluster, called the *center*, and define the *cost* of a clustering as a function of the distances between the data points and their respective centers.

Definition 4.2. *The problem of finding a k-partition $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of the given dataset and finding k* centers *$c_1, c_2, \ldots, c_k$ such that*

- *we minimize $\sum_i \sum_{x \in C_i} d(x, c_i)$ is called k-*median*.*
- *we minimize $\sum_i \sum_{x \in C_i} d^2(x, c_i)$ is called k-*means*.*
- *we minimize $\max_i \max_{x \in C_i} d(x, c_i)$ is called k-*center*.*

From here on out we denote the optimal clustering as $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_k^*\}$, its respective centers as $c_1^*, c_2^*, \ldots, c_k^*$, and its cost as OPT.

It is simple to see that given a list of k centers, the k-partition that minimizes the abovementioned costs is to assign each point to its nearest center. This partition, in which C_i is the set of points that prefer c_i to any other c_j is known as the *Voronoi partition* of the metric space, and C_i is called the *Voronoi region* of c_i. Conversely, given a k-partition $\{C_1, C_2, \ldots, C_k\}$, it is simple to find the best center point c_i for the points in C_i in a finite metric space (by brute forcing trying all points in C_i). In the special case of k-means in Euclidean space, it is a known fact that the best center point is the *centroid* of C_i, denoted $\mu(C_i) = \frac{1}{|C_i|} \sum_{x \in C_i} x$. In fact, given a set X of points in $\mathbb{R}^d$, we treat μ as an operator where $\mu(X) = \frac{1}{|X|} \sum_{x \in X} x$.

In this thesis, we view k as part of the input and not a constant, though even the 2-means problem in Euclidean space was recently shown to be NP-hard [57]. Indeed, center-based clustering in finite metrics, like many other k-partition problems (e.g. Max k-coverage, Knapsack for k items, maximizing social welfare in k-items auction), can be easily solved in n^k time. (And Kumar et al [105] give a PTAS for the k-means problem in Euclidean space.) We also comment that there is a variety of applications where k is quite large. This includes problems such as clustering images by who is in them, clustering protein sequences by families of organisms, and problems such as deduplication where multiple databases are combined and entries corresponding to the same true entity are to be clustered together [52, 117].

All objectives are known to be NP-hard. For k-median in a finite metric, there is a known $(1 + 1/e)$-hardness of approximation result [90] and substantial work on approximation algorithms [76, 48, 18, 90, 60, 106], with the best guarantee of a $(1 + \sqrt{3} + \epsilon)$-approximation.[1] For k-means in a Euclidean space, there is also a vast literature of approx-

[1] There is also a PTAS known for low-dimensional Euclidean spaces (dimension at most $\log \log n$) [14, 79].

imation algorithms [120, 24, 60, 67, 79, 98] with the best guarantee a $(9+\epsilon)$-approximation if polynomial dependence on k and the dimension d is desired. For the k-center problem there's a known greedy algorithm that yields a 2-approximation [74, 85], as well as a $2-\epsilon$ hardness of approximation result [87]. In the Euclidean plane with L_2 or L_∞ metric, there's a PTAS whose running time is $O(n\log(k)+(k/\epsilon)^{O(k^{1-1/d})})$ [5].

4.1.1 Other Clustering Techniques

In this introduction, we do not aim to survey all non-center-based clustering techniques. We mention just a few: min-sum clustering – in which the goal is to find a k-clustering that minimizes $\sum_i \sum_{x,y\in C_i} d(x,y)$; max k-cut – in which the goal is to find a k-clustering that maximizes $\sum_{i<j} \sum x \in C_i, y \in C_j d(x,y)$; and normalized cuts – in which one aims to minimize $\sum_i \frac{1}{|C_i|} \sum_{x\in C_i, y\notin C_i} d(x,y)$ or one of the other many variations of this objective (e.g, conductance).[2] However, in this thesis we will make use of the Single-Linkage algorithm, and therefore it is worth-while to mention the agglomerative clustering algorithms, that use the bottom-up approach.

Definition 4.3. *The clustering algorithm that starts with n singleton clusters and repeatedly merges the two clusters X, Y that minimize $f(X,Y)$ where*

- $f(X,Y) = \min_{x\in X, y\in Y} d(x,y)$ *is called the* Single-Linkage *algorithm.*
- $f(X,Y) = \frac{1}{|X|\,|Y|} \sum_{x\in X, y\in Y} d(x,y)$ *is called the* Average-Linkage *algorithm.*
- $f(X,Y) = \max_{x\in X, y\in Y} d(x,y)$ *is called the* Complete-Linkage *algorithm.*

We comment that the classic variants of these algorithm halt when there are k clusters left. In this thesis, however, we make use of a slightly less standard variant, in which the algorithm halts whenever a single cluster is reached. This in fact produces a hierarchical clustering or a clustering tree, and we say the algorithm is laminar with a given clustering $\mathcal{C}$ if there a pruning of the clustering tree that produces $\mathcal{C}$.

The literature regarding agglomerative clustering is rich and vast and we do not survey it here. The interested reader is referred to [83].

[2]For normalized cuts, we think of the distance as a measure of similarity rather than difference. There are numerous way of coverting distances into similary measure, the simplest of which is to put an edge between any two x, y for which $d(x,y)$ is smaller than some threshold.

4.2 Implicit Stability Assumptions for Clustering

The approximation results surveyed thus far are worst-case results. Each of the works mentioned poses some c and constructs an algorithm which is guaranteed to produce a c-approximation for *any* clustering instance. However, while they guarantee to output some $\mathcal{C}$ whose cost is at most $c\mathsf{OPT}$, they do not guarantee that the output $\mathcal{C}$ satisfies any other proximity notion to $\mathcal{C}^*$ – they do not guarantee that $\mathcal{C}$ and $\mathcal{C}^*$ agree on the label of most datapoints, they do not guarantee that the centers of $\mathcal{C}$ are close to the centers of $\mathcal{C}^*$, and they don't even guarantee that $\mathcal{C}$ and $\mathcal{C}^*$ have the same number of cluster. In this section we discuss several works that indeed give such stronger guarantees. The caveat is that the works we survey in this section do *not* fall into the framework of worst-case analysis. These works give stronger guarantees than mere c-approximation of the cost by focusing solely on a specific subset of all possible clustering instances – instances that adhere to certain "nice" properties. As always, the exact definition of the properties changes from one work to another.

4.2.1 Clustering objectives as a proxy for datapoints labeling

The work of Balcan, Blum and Gupta [25] takes a machine-learning viewpoint of clustering. Balcan et al consider a notion of stability to approximations motivated by settings in which there exists some (unknown) target clustering $\mathcal{C}^{target}$ we would like to produce. Balcan et al define a clustering instance to be $(1+\alpha, \delta)$ approximation-stable with respect to some objective Φ (such as k-median or k-means), if any k-partition whose cost under Φ is at most $(1+\alpha)\mathsf{OPT}$ agrees with the target clustering on all but at most δn data points.

Definition 4.4. *Given a clustering instance w.r.t to objective Φ, we call the instance $(1+\alpha, \delta)$*-approximation stable *if for any $(1+\alpha)$ approximation $\mathcal{C}$ to objective Φ, we have $\min_{\sigma \in S_k} \sum_i |C_i^{target} - C_{\sigma(i)}| \leq \delta n$ (here, σ is simply a matching of the indices in the target clustering to those in $\mathcal{C}$).*

For instances satisfying the $(1+\alpha, \delta)$ approximation-stability assumption, Balcan et al have shown the following results.

Theorem 4.5 ([25])**.**
- *Assuming that the instance satisfies $(1+\alpha, \delta)$-approximation stability w.r.t to either the k-median objective or the k-means objective, then there exists a clustering algorithm whose output disagrees with $\mathcal{C}^{target}$ on no more than a fraction of $O(\delta(1+1/\alpha))$ of the datapoints.*

- *Assuming that (i) the instance satisfies $(1+\alpha, \delta)$-approximation stability w.r.t to the k-median objective and (ii) all clusters have size at least $n \cdot O(\delta(1+1/\alpha))$, then there exists a clustering algorithm whose output disagrees with $\mathcal{C}^{target}$ on no more than a fraction of δ of the datapoints.*

- *Assuming that (i) the instance satisfies $(1+\alpha, \delta)$-approximation stability w.r.t to the k-means objective and (ii) all clusters have size at least $n \cdot O(\delta(1+1/\alpha))$, then there exists an algorithm that outputs a $(k+1)$-partition $\{C_1, C_2, \ldots, C_k, U\}$, s.t. $\frac{|U|}{n} = O(\delta(1+1/\alpha))$ and $\min_{\sigma \in S_k} |C_i \setminus C^{target}_{\sigma(i)}| < \delta n$.*

- *The problem of finding a $(1+\alpha)$-approximation in general can be reduce to the problem of finding a $(1+\alpha)$-approximation on instances satisfying $(1+\alpha, \delta)$-approximation stability, for any $\delta > 1/\mathrm{poly}(n)$.*

We do not give here full details as to the algorithms that Balcan et al [25] devise. However, they are all extensions of the following basic idea, best illustrated using the k-median objective. Given the optimal clustering $\mathcal{C}^*$, denote for every datapoint x its contribution to OPT by $\phi(x) = \min_i d(x, c_i)$ and its distance to the 2nd closest center as $\psi(x)$. Take the $O(\delta)n$ points whose difference $\psi(x) - \phi(x)$ is the largest, and let $\mathcal{C}'$ be the clustering in which of these points is assigned to its 2nd closest center. Since $\mathcal{C}^*$ and $\mathcal{C}'$ do not agree on δn points, then the cost of $\mathcal{C}'$ is greater than $(1+\alpha)\mathsf{OPT}$. We deduce that at most a fraction of $O(\delta)$ points have $\psi(x) - \phi(x) < \frac{\alpha}{\delta}\frac{\mathsf{OPT}}{n}$. On the other hand, using Markov's inequality we have that at most a $O(\delta/\alpha)$ fraction of the points have $\phi(x) \geq \frac{\alpha}{5\alpha}\frac{\mathsf{OPT}}{n}$. So, for a fraction of at least $1-O(\delta(1+1/\alpha))$ of the points neither property holds. For any two points x, y in this set, we have that if they belong to the same cluster then $d(x,y) < 2\frac{\alpha\mathsf{OPT}}{\delta n}$, whereas if they belong to two different clusters $d(x,y) > 3\frac{\alpha\mathsf{OPT}}{\delta n}$. The algorithms of Balcan et al [25] build on this strict-separation property, taking into account the $O(\delta(1+1/\alpha))n$ outliers and fact with don't know OPT in advance.

Following the work of [25] similar notions were considered in other works. Balcan et al [25] themselves also discuss approximation-stability w.r.t to the min-sum objective, and their results were extended by Balcan and Braverman [28]. Balcan, Röglin, and Teng [30] studied instances that satisfy $(1+\alpha, \delta)$-approximation stability only after a fraction of data is removed. The work of Schalekamp et al [130] analyzes the algorithm of [25] in practice and show it gives a good approximation to the k-median objective when all clusters are large.

4.2.2 k-clustering whose cost is much smaller than the cost of $(k-1)$-clustering

Ostrovsky et al [121] proposed an interesting condition under which one can achieve better k-means approximations in time polynomial in n and k. They consider k-means instances where the optimal k-clustering has cost noticeably smaller than the cost of any $(k-1)$-clustering, motivated by the idea that "if a near-optimal k-clustering can be achieved by a partition into fewer than k clusters, then that smaller value of k should be used to cluster the data" [121]. Formally,

Definition 4.6. *Given a clustering instance w.r.t some objective Φ, let OPT_{k-1} denote the cost of the optimal $(k-1)$-clustering, and OPT denote the cost of the optimal k-clustering. We say the instance is* γ-separable *if $\frac{\mathsf{OPT}}{\mathsf{OPT}_{k-1}} \leq \gamma^2$.*

Under the assumption that $\gamma < 1/10$, Ostrovsky et al show that one can obtain a $(1+O(\gamma^2))$-approximation for k-means in time polynomial in n and k, by using Lloyd steps: using centers $\{c_1, c_2, \ldots, c_k\}$ set C_i as the set of points whose closest center point is c_i and then set $c_i \leftarrow \mu(C_i)$, and repeat until convergence. In fact, Ostrovsky et al pay careful attention to the seeding of the Lloyd iterations – the initial set of k centers. More formally, Ostrovsky et al proved the following.

Theorem 4.7 ([121])**.** *Given a k-means instance which is γ-separable for some $\gamma < 1/10$, there exists a $O(nkd + k^3 d)$-time algorithm that gives a $(1+72\gamma^2)$-approximation of the k-means cost w.p. $1 - O(\sqrt{\gamma})$. Given $\epsilon > 0$, there exists a $2^{O(k/\epsilon)} nd$-time algorithm that gives a $(1+\epsilon)$-approximation of the k-means cost with constant probability.*

A rough outline of the $(1+O(\gamma^2))$-approximation algorithm of Ostrovsky et al is as follows.

Randomly initial centers. Pick first two datapoints w.p. proportional to their distance squared. Pick additional $O(k)$ points iteratively: given $\{p_1, p_2, \ldots, p_l\}$ choose the next point $p_{l+1} = x$ w.p. $\min_{i=1}^{l} \|x - p_i\|^2$.

Picking k centers. Given the $O(k)$ points chosen, find their Voronoi regions and their respective centroids. Starting from this set of $O(k)$ centroids, apply some deletion procedure until k points are left.

One "Lloyd step". Given $\{p_1, \ldots, p_k\}$, let $B_i = \{x : \|x - p_i\| \leq \frac{1}{3} \min_{j \neq i} \|p_i - p_j\|\}$. Return $\{\mu(B_1), \mu(B_2), \ldots, \mu(B_k)\}$.

In fact, much of the analysis of Ostrovsky et al is devoted to the initial center-procedure. In particular, they show that by picking just k centers using this sampling technique we have at least a $(1 - O(\gamma^{2/3}))^k$ probability of finding one point from each cluster. Alternatively, sampling $O(k)$ points with this sampling procedure, we have that our sample contains at least one point from each cluster with constant probability.

Following the work of Ostrovsky et al, this sampling procedure was refined slightly to what is now known as D^2 sampling. (Where instead of picking the first 2 points proportional to their distance, we pick the first u.a.r and the 2nd is picked the same way the rest are chosen.) Arthur and Vassilvitskii [15] showed that D^2 sampling yields a $O(\log(k))$-approximation to the k-means optimum. Ailon et al [8] have shown that picking $O(k \log(k))$ centers using D^2 sampling yields a $O(1)$-approximation to the k-means objective, and Aggarwal et al [6] showed the same for picking $O(k)$ centers. Jaiswal and Garg [91] showed that using D^2 sampling for sampling k centers gives a $O(1)$-approximation of the k-means objective w.p. $\geq 1/k$ if the input is γ-separable for some constant γ, and in general gives a $O(1)$-approximation w.p. $\Omega(2^{-2k})$.

4.2.3 Clustering perturbation resilient instances

Bilu and Linial [40], focusing on the max-cut problem [71], proposed considering instances where the optimal clustering is optimal not only under the given metric, *but also under any bounded multiplicative perturbation of the given metric*. This is motivated by the fact that in practice, distances between data points are typically just the result of some heuristic measure (e.g., edit-distance between strings or Euclidean distance in some feature space) rather than true "semantic distance" between objects. Thus, unless the optimal solution on the given distances is correct by pure luck, it likely is correct on small perturbations of the given distances as well. We formally define metric perturbation and perturbation resilience.

Definition 4.8. *Given a metric (S, d), and $\alpha > 1$, we say a function $d' : S \times S \to \mathbb{R}_{\geq 0}$ is an* α-perturbation *of d, if for any $x, y \in S$ it holds that $d(x, y) \leq d'(x, y) \leq \alpha d(x, y)$. Note that d' may be any non-negative function and might not satisfy the triangle inequality.*

Definition 4.9. *Given a clustering instance composed of n points residing in a metric (S, d), we say the instance is* α-perturbation resilient *for a clustering objective Φ if for any d' which is an α-perturbation of d, the (only) optimal clustering of (S, d') under Φ is identical, as a partition of points into subsets, to the optimal clustering of (S, d) under Φ.*

Bilu and Linial [40] analyze max-cut instances satisfying perturbation-resilience, and

show that for instances that are stable to perturbations of multiplicative factor roughly $O(n^{1/2})$, one can retrieve the optimal Max-Cut in polynomial time.

Theorem 4.10 ([40]). *There exists a poly-time algorithm that correctly clusters $O(\sqrt{n})$-perturbation resilient max-cut instances.*

While the original version of the algorithm is to round the SDP relaxation of Goemans-Williamson [72] for the max-cut problem, an approach that relies on heavy machinery from linear algebra, there is a simpler combinatorial and deterministic algorithm due to Bilu et al [39]. This algorithm iteratively identifies two vertices that belong to the same cluster and contracts them (while summing the weights of contracted edges). It is straight-forward to see that the contracted instance is also perturbation-resilience.

Bilu and Linial conjectured that stability up to only *constant* magnitude perturbations should be enough to solve the problem in polynomial time. A recent work of Makarychev et al [109] improves on the result of Bilu and Linial and improves the perturbation-resilience factor to $O(\sqrt{\log(n)}\log\log(n))$. Multiple works (including the one in this thesis) discuss perturbation stability for other problems, such as k-median and min-sum [29, 127], TSP [114], and finding Nash-equilibrium [19].

Additional notions of stability in clustering. We conclude by mentioning other works that tie together the notion of clustering and stability. Ben-David et al. [36, 37] consider a notion of stability of a clustering algorithm, which is called stable if it outputs similar clusters for different sets of n input points drawn from the same distribution. For k-means, the work of Meila [113] discusses the opposite direction – classifying instances where an approximated solution for k-means is close to the target clustering. And relating to the effectiveness of Lloyd-type iterations, it was shown that this iterative algorithm might take exponential time to converge even for planar $(d = 2)$ instances [137], yet other works have shown that adding random Gaussian noise to each point independently causes the algorithm to converge in time $O(n^k)$ ([17]) or in time $\mathrm{poly}(n, k)$ ([16]).

4.2.4 Our Contribution: Weak-Deletion Stability and Perturbation-Resilience for k-Median and k-Means

In Chapter 7 we improve on the works of Ostrovsky et al [121] and Balcan et al [25], and in fact unify them. Ostrovsky et al call a dataset separable if the ratio of the optimal clustering with $k-1$ clusters to the optimal clustering with k clusters is at least $1/\gamma^2 \geq 100$ and give a $(1 + O(\gamma^2))$-approximation for the k-means objective for such an instance.

We call a dataset stable if the above ratio is greater than γ for some constant $\gamma > 1$, and give a PTAS for such instance. In other words, we decouple the strength of the assumption from the quality of the approximation – given time $n^{O(1/\epsilon)}$ we show how to get a $(1+\epsilon)$-approximation for either the k-median of the k-means objectives for any separable dataset. Balcan et al give an algorithm that correctly clusters a $1-\delta$ fraction of the points give k-median instances that (i) satisfy the [25] notion of stability and (ii) all clusters in the optimal clustering are very large clusters. Using the same notion of stability and defining cluster as "large" if their size is only $\Omega(\delta n)$, we give an algorithm that approximates the k-median or the k-means cost and therefore clusters all but a fraction of δ of the points correctly. The result is obtained by defining a new notion of stability, *weak-deletion stability*, that unifies both previously considered notions of stability.

In Chapter 8 we extend the Bilu-Linial [40] notion of perturbation resilience to any center-based clustering objective, like k-median or k-means. We show that there exists an efficient algorithm that correctly clusters any 3-perturbation resilient instance. This algorithm is no other than the Single-Linkage algorithm (or rather, a simple variant on the Single-Linkage algorithm).

4.3 Clustering under Explicit Distributional Assumptions

All of the results mentioned in Section 4.2 discuss clustering instances which have certain nice properties due to some compelling story. In contrast, a different line work in clustering pin-points the nature of the clustering instance so that it comes from a specific model. More importantly, in this line of work the datapoints are not arbitrary points of some instance, but rather they are the outcome of multiple iid drawns from some distribution. For these types of problems, the algorithm is often aware of the nature of the distribution (i.e. Gaussians, log-concave, heavy-tail etc.) yet the specifics of the distribution (namely, mean and variance) are unknown.

Specifically, we discuss a *mixture model*. In this type of instance we get to view n points in a d-dimensional Euclidean space, and each point was drawn iid from some distribution. We assume that overall there exist k different distributions which leads to the existence of k clusters in our instance – where all points in cluster i are drawn from the same distribution. In fact, we can think of the points themselves as points that were sampled independently from the ith distribution w.p. $w_i = |C_i|/n$. We will use the standard notation $\mu_i = \mu(C_i) \in \mathbb{R}^d$ and $\Sigma_i \in \mathbb{R}^{d \times d}$ to denote the mean and variance of the ith distribution, and the maximal directional-variance of the ith distribution (the leading singular value of Σ_i) is denoted as σ_i. We also denote $w_{\min} = \min_i\{w_i\}$ and $\sigma_{\max} = \max_i\{\sigma_i\}$.

Observe – the standard approach is to design algorithms that give good estimation of the problem's parameters, namely $\{\langle w_i, \mu_i, \Sigma_i \rangle\}_{i=1}^k$. In this thesis however, we focus on works that retrieve the actual clustering of the data, after which it is easy to estimate these parameters using the empirical weights, means and variances.

Of the immensely rich literature regarding learning mixture models, we focus in this overview on one particular venue – learning under center-separation assumptions. We discuss works that cluster instances that arise from mixture models under the additional assumption of *center separation*: That for every pair of cluster centers μ_i, μ_j with $i \neq j$ we have that $\|\mu_i - \mu_j\|$ is lower bounded by some function $f(n, k, d, w_i, w_j, \sigma_i, \sigma_j)$.

4.3.1 Gaussian Mixture Model

Probably the most well-studied case of this mixture model is the Gaussian mixture model, in which all k distributions are multivariate Gaussians. Dasgupta [56] was the first to give an algorithm with a guaranteed utility bound for the case of Gaussian mixture model with center separation of $\forall i \neq j, \quad \|\mu_i - \mu_j\| \geq \sqrt{d}(\sigma_i + \sigma_j)$, focusing on the case where Gaussians are spherical (where $\forall i, \quad \Sigma_i = \sigma_i I_{d\times d}$) and $w_{\min} = \Omega(1/k)$. This bound was later improved to Dasgupta and Schulman [59] for the case of spherical Gaussian with center separation of $\Omega(d^{1/4}(\sigma_i + \sigma_j)\log(n))$.

While the algorithms of Dasgupta [56] and Dasgupta and Schulman [59] are non-trivial (the first involves random projections and the latter involves a variant of the EM algorithm), there are simple observations that provide intuition as to why such clustering tasks should be poly-time solvable. Focusing on spherical Gaussians, it is a known fact that is $x \sim \mathcal{N}(\mu_i, \sigma_i I_{d\times d})$ then $\mathbf{E}[\|x - \mu\|] = \sigma\sqrt{d}$ and the variance of the distance is roughly $\sigma d^{1/4}$. Moreover, up to constant factors, these bounds also apply for two independent samples from the same spherical Gaussian. In fact, the following lemma from [59] gives tight concentration bounds on the distances between any two datapoints.

Lemma 4.11 ([59])**.** *Let $x \sim \mathcal{N}(\mu_i, \sigma_i I_{d\times d})$ and $y \sim \mathcal{N}(\mu_j, \sigma_j I_{d\times d})$ for any i, j (not necessarily different). Then for any $\alpha > 0$ we have that w.p. $\geq 1 - e^{\Omega(-d^{2\alpha})}$ it holds that*

$$\|x - y\|^2 \in \|\mu_i - \mu_j\|^2 + (\sigma_i^2 + \sigma_j^2)(d \pm d^{1/2+\alpha}) \pm 2\|\mu_i - \mu_j\| d^{\alpha}\sqrt{\sigma_i^2 + \sigma_j^2}$$

So roughly speaking, when $\|\mu_i - \mu_j\|^2 > d^{\frac{1}{2}+\alpha}(\sigma_i^2 + \sigma_j^2)$ we have that w.h.p. each point is closer to the points of its own cluster than any of the points of any of the other clusters.

The next step in the line of works learning Gaussian mixture models under center separation was the work of Vempala and Wang [138]. They observed the for spherical Gaussians, projecting the data onto the subspace spanned by the top k singular vectors keeps all Gaussian centers in place. In other words, the subspace spanned by the top k singular vectors is precisely the subspace spanned by the k centers $\{\mu_1, \mu_2, \ldots, \mu_k\}$. As a result, for spherical Gaussians the center-separation bound they introduced no longer depends on d, and rather it is $\|\mu_i - \mu_j\| = \tilde{\Omega}(k^{1/4}(\sigma_i + \sigma_j))$. Achlioptas and McSherry [4] gave an algorithm for general Gaussians under the center-separation bound of $\tilde{\Omega}((\sqrt{k} + \sqrt{1/w_i} + \sqrt{1/w_j})(\sigma_i + \sigma_j))$. Achlioptas and McSherry's main observation is that projecting the data on its top k singular values shifts the cluster centers by no more than $\sigma_{\max}/\sqrt{w_i}$, while leaving the distance between a datapoint and its corresponding cluster-center at $\sqrt{k}\sigma_i$. Therefore, suppose cluster 1 has the largest directional variance $\sigma_1 = \sigma_{\max}$, then the center separation bound along with triangle inequality give that w.h.p. we have for any two projected points x, y from the same cluster and two projected points x', y' from cluster C_1 we have $\|x - y\| \leq \|x' - y'\|$, and for any two projected points $x'' \in C_1$ and $y'' \notin C_1$ it holds that $\|x' - y'\| < \|x'' - y''\|$. As a result, by running Single-Linkage on the projected instance and stopping with two cluster, we have a partition of the dataset that is laminar with the original clustering (we can then recurse of each side of the partition).

Other Mixture Models. Following Dasgupta [56], mixture models of other distributions were studied as well. Arora and Kannan [13] study arbitrary Gaussians and log-concave distributions and also Achlioptas and McSherry [4] results also apply to log-concave distributions. Kannan et al [96] and Chaudhuri and Rao [49] apply SVD projections to other mixture models. Chaudhuri and Rao [50] also study mixture model for product distributions, and Dasgupta et al [54] study general mixture model (under center separation that is polynomially dependent on n). Finally Brubaker and Vempala [47] tweak with the separation condition, just for the case of $k = 2$ Gaussians, requiring that $\|\mu_1 - \mu_2\|$ is greater than the projected variance of the Gaussian along the direction of $\mu_1 - \mu_2$. Kalai, Moitra and Valiant [95] give an algorithm for learning 2 Gaussians with no separation assumptions, and Moitra and Valiant [116] extend this result to any constant k. Belkin and Sinha [35] give an algorithm for various other mixture models (with no separation assumptions).

4.3.2 The Planted Partition Model

A very different distributional model was considered in the work of McSherry [110]. In his *Planted Partition Model* the input is a graph over n nodes of k types (corresponding

to the k clusters). The edges of this graphs were sampled independently from a Bernoulli random variable, where for every $u \neq v$, the edge uv is placed in the graph w.p. $p_{c(u),c(v)}$ where $c(u)$ and $c(v)$ are the clusters of u and v respectively. As always, our goal is to find the k-clustering that yielded this graph (and to estimate $p_{i,j}$, which is easy given the correct k-clustering).

It is clear that clustering would be easy if we had access to the matrix $P \in \mathbb{R}^{n \times n}$ where $P_{u,v} = p_{c(u),c(v)}$. But it is less clear whether the Planted Partition model fits into the framework of clustering via center proximity. In fact – what are the cluster means in this case? Observe, if we take the data and view a node u as a vector $x_u \in \{0,1\}^n$, indicating the set of nodes that are u's neighbors, then by averaging the vectors $\{x_u : u \in C_i\}$ we have that their mean, μ_i, should get very close to the row vector corresponding to the nodes in C_i of P. Furthermore, since each row x_u is a random rounding of μ_i to integral values, then x_u should be fairly close to its expectation, and in fact closer to μ_i than any other μ_j. This gives rise to the hope that with a suitable center separation bound, we should be able to cluster the nodes correctly. McSherry also observed that P is a rank k matrix, and so the separation bound in [110] has polynomial dependence on k rather than on n.

Formally, denoting $\sigma_{\max} = \max_{i,j} \sqrt{P_{ij}}$, McSherry proved that under center separation of

$$\|\mu_i - \mu_j\| = \Omega\left(\sigma_{\max}\sqrt{k}(\tfrac{1}{w_{min}} + \log(\tfrac{n}{\delta}))\right)$$

it is possible to correctly cluster all nodes w.p. $1 - \delta$.

4.3.3 Our Contribution: Improving on Kumar-Kannan Separability

Aiming to unify many of the previous works regarding mixture models, Kumar and Kannan [104] defined a *deterministic* condition, which is independent of any specific distribution, and show how to correctly cluster datasets satisfying this condition. Having established an algorithm that correctly clusters datasets satisfying this condition, they show that indeed many of the previously studied mixture models do satisfy this separation condition (w.h.p). However, their approach has two drawbacks: first, its guarantees are wasteful with respect to k, the number of clusters, and don't match the best known bounds' dependencies on k; secondly, their condition is somewhat complicated in comparison to the idea of center-separation. In Chapter 9 we give further details regarding the work of Kumar and Kannan, and discuss our improvements. In particular, we show how the basic tools of the triangle inequality and Markov inequality allow us to give a simple analysis of their algorithm and its various applications.

Part II

Differential Privacy

"Oh, look . . . they're reading '1984' in Ms. Smith's English class."

Chapter 5

The Johnson-Lindenstrauss Transform Itself Preserves Differential Privacy

5.1 Introduction

The celebrated Johnson Lindenstrauss transform [93] is widely used across many areas of Computer Science. A very non-exhaustive list of related applications include metric and graph embeddings [45, 107], computational speedups [129, 139], machine learning [26, 131], information retrieval [123], nearest-neighbor search [103, 89, 7], and compressed sensing [31]. Here we unveil a new application of the Johnson Lindenstrauss transform – it also preserves differential privacy. It allows us to release statistics about our given database, while guaranteeing that for any two neighboring databases (databases that differ on the details of any single individual), the distributions over potential outputs are statistically close.

The simplest technique of preserving differential privacy by adding small random noise (Theorem 3.3) lies at the core of an overwhelming majority of algorithms that preserve differential privacy. In fact, many differentially private algorithms follow a common outline. They take an existing algorithm and revise it by adding such random noise each time the algorithm operates on the sensitive data. Proving that the revised algorithm preserves differential privacy is almost immediate, because differential privacy is composable. On the other hand, providing good bounds on the revised algorithm's utility follows from bounding the overall noise added to the algorithm, which is often difficult. Here we take the complementary approach. We show that an existing algorithm preserves differential privacy provided we slightly alter the input in a reversible way. Our analysis of the algorithm's

utility is immediate, whereas privacy guarantees require a non-trivial proof.

We prove that by multiplying a given database with a vector of iid normal Gaussians, we can output the result while preserving differential privacy (assuming the database has certain properties, see "our technique"). This technique is no other than the Johnson-Lindenstrauss transform, and it's guaranteed to preserve w.h.p the L_2 norm of the given database up to a small multiplicative factor. Therefore, whenever answers to users' queries can be formalized as the length of the product between the given database and a query-vector, utility bounds are straight-forward.

For example, consider the case where our input is composed of n points in $\mathbb{R}^d$ given as a $n \times d$ matrix. We define two matrices as neighbors if they differ on a single row and the norm of the difference is at most 1.[1] Under this notion of neighbors, a simple privacy preserving mechanism allows us to output the mean of the rows in A, but what about the covariance matrix $A^\intercal A$? We prove that the JL transform gives a (ϵ, δ)-differentially private algorithm that outputs a sanitized covariance matrix. Furthermore, for *directional variance queries*, where users give a unit-length vector x and wish to know the variance of A along x (see definition in Section 5.2), we give utility bounds that are *independent of* d *and* n. In contrast, all other differentially private algorithms that answer directional variance queries have utility guarantees that depend on d or n. Observe that our utility guarantees are somewhat weaker than usual. Recall that the JL lemma guarantees that w.h.p lengths are preserved up to a small multiplicative error, so for each query our algorithm's estimation has w.h.p small multiplicative error and additional additive error.

A special case of directional variance queries is *cut-queries* of a graph. Suppose our database is a graph G and define two graphs as neighbors if they differ on a single edge. Indeed, this edge-adjacency notion between graphs is weaker than the notion of a vertex-adjacency, where two neighboring graphs may differ on any number of edges incident to same vertex. However, this notion of adjacency corresponds to the same notion of adjacency between matrices considered above – in Chapter 2.1 we defined the *edge matrix* E_G of a graph G, and indeed for any two G and G' that differ on a single edge, E_G and $E_{G'}$ differ on a single row (with the norm of the difference equals $O(1)$). For cut-queries, users pose a nonempty strict subset of vertices S, and wish to know how many edges in G cross the $(S, \bar{S})$-cut. Such a query can be formalized as the L_2-norm squared of the product $E_G \mathbf{1}_S$, where $\mathbf{1}_S$ is the indicator vector of S (see Chapter 2.1). Here, we prove that the JL transform allows us to publish a perturbed Laplacian of G while preserving (ϵ, δ)-differential privacy. Comparing our JL-based algorithm to existing algorithms, we show that we add (w.h.p) $O(|S|)$ random noise to the true answer (alternatively: w.h.p we

[1] This notion of neighboring inputs, also considered in [111, 82], is somewhat different than the typical notion of privacy, allowing any individual to change her attributes arbitrarily.

add only constant noise to the query $\frac{\mathbf{1}_S^\intercal E_G^\intercal E_G \mathbf{1}_S}{\mathbf{1}_S^\intercal \mathbf{1}_S}$). In contrast, all other algorithms add noise proportional to the number of vertices (or edges) in the graph.

Our technique. It is best to demonstrate our technique on a toy example. Assume D is a database represented as a $\{0,1\}^n$-vector, and suppose we sample a vector Y of n iid normal Gaussians and publish $X = Y^\intercal D$. Our output is therefore distributed like a Gaussian random variable of 0 mean and variance $\sigma^2 = \|D\|^2$. Assume a single entry in D changes from 0 to 1 and denote the new database as D'. Then $X' = Y^\intercal D'$ is distributed like a Gaussian of 0-mean and variance $\lambda^2 = \|D\|^2 + 1$. Comparing $\mathsf{PDF}_X(x) = (2\pi\sigma^2)^{-1/2}\exp(-x^2/(2\sigma^2))$ to $\mathsf{PDF}_{X'}(x) = (2\pi\lambda^2)^{-1/2}\exp(-x^2/(2\lambda^2))$ we have that $\forall x,\ \sqrt{\lambda^2/\sigma^2}\mathsf{PDF}_{X'}(x) \geq \mathsf{PDF}_X(x) \geq \exp(-\frac{x^2}{2\sigma^2}\cdot\frac{1}{\lambda^2})\mathsf{PDF}_{X'}(x)$. Using concentration bounds on Gaussians we deduce that if $\lambda^2 > \sigma^2 = \Omega(\log(1/\delta)/\epsilon)$, then w.p $\geq 1-\delta$ both PDFs are within multiplicative factor of $e^{\pm\epsilon}$. We now repeat this process r times (setting ϵ, δ accordingly) s.t. the JL lemma assures that (after scaling) w.h.p we output a vector of norm $(1\pm\eta)\|D\|^2$ for a given η. We get utility guarantees for publishing the number of ones in D while preserving (ϵ,δ)-differential privacy.

Keeping with our toy example, one step remains – to convert the above analysis so that it will hold for any database, and not only databases with $w \stackrel{\text{def}}{=} \log(1/\delta)/\epsilon$ many ones. One way is to append the data with w one entries, but observe: this ends up in outputting $X + N$ where N is random Gaussian noise! In other word, appending the data with ones makes the above technique worse (noisier) than the classical technique of adding random Gaussian noise. Instead, what we do is to "translate the database". We apply a simple *deterministic* affine transformation s.t. D turns into a $\{\sqrt{\frac{w}{n}}, 1\}^n$-vector. Applying the JL algorithm to the translated database, we output a vector whose norm squared is $\approx (1\pm\eta)(\|D\|^2 + w)$. Clearly, users can subtract w from the result, and we end up with ηw additive random noise (in addition to the multiplicative noise).[2]

It is tempting to think the above analysis suffices to show that privacy is also preserved in the multidimensional case. Consider the cases of two adjacent graphs, G and G' with an edge (a,b) present in G' and absent in G. The corresponding edge matrices E_G and $E_{G'}$ differ only on a single row (see Chapter 2.1) – which is the all 0 row in E_G and a row with only two non-zero coordinates in $E_{G'}$. So when we compare the result of multiplying $E_G^\intercal$ with a vector of iid normal Gaussians to the result of multiplying $E_{G'}^\intercal$ with such vector, only two coordinates in the outcome behave differently. Presumably, applying

[2]Observe that in this toy example, our $O(\log(1/\delta)/\epsilon)$ noise bound is still worse than the noise bound of $O(\sqrt{\log(1/\delta)}/\epsilon)$ one gets from adding Gaussian noise. However, in the applications detailed in Sections 5.3 and 5.4, the idea of changing the input will be the key ingredient in getting noise bounds that are independent of n and d.

the abovementioned univariate analysis to each of the two coordinates that change suffices to prove we preserve differential privacy. Yet this intuition is false. Multiplying E_G with a random vector does not result in n independent Gaussians, but rather in one multivariate Gaussian. This is best illustrated with an example. Suppose G is a graph and S is a subset of nodes s.t. no edge crosses the $(S, \bar{S})$-cut. Therefore $\|E_G \mathbf{1}_S\|^2 = |E(S, \bar{S})| = 0$ so $E_G \mathbf{1}_S$ is the zero-vector, and no matter what random projection R we pick, $R^\intercal E_G \mathbf{1}_S = 0$. In contrast, by adding a single edge that crosses the $(S, \bar{S})$-cut, we get a graph G' s.t. $\mathbf{Pr}[R^\intercal E_{G'} 1_S \neq 0] = 1$.

Organization. Next we detail related work. Section 5.2 details important notations and formal definitions. In Sections 5.3 and 5.4 we convert the above univariate intuition to the multivariate Gaussian case. Section 5.3 describes our results for graphs and cut-queries, and in Section 5.3.2 we compare our method to other algorithms. Section 5.4 details the result for directional queries (the general case), then a comparison with other algorithms. Even though there are clear similarities between the analyses in Sections 5.3 and 5.4, we provide both because the graph case is simpler and analogous to the univariate Gaussian case. Suppose G and G' are two graphs without and with a certain edge resp., then G induces the multivariate Gaussian with the "smaller" variance, and G' induces the multivariate Gaussian with the "larger" variance. In contrast, in the general case there's no notion of "smaller" and "larger" variances. Also, the noise bound in the general case is larger than the one for the graph case, and the theorems our analysis relies on are more esoteric. Section 5.5 concludes with a discussion and open problems.

5.1.1 Related Work

The task of preserving differential privacy when the given database is a graph or a social network was studied by Hay et al [84] who presented a privacy preserving algorithm for publishing the degree distribution in a graph. They also introduced and compared between multiple notions of neighboring graphs, one of which is for the change of a single edge. Nissim et al [119] (see full version) studied the case of estimating the number of triangles in a graph, and Karwa et al [100] extended this result to other graph structures. Gupta et al [77] studied the case of answering (S, T)-cut queries, for two disjoint subsets of nodes S and T. All latter works use the same notion of neighboring graphs as we do. In differential privacy it is common to think of a database as a matrix, but seldom one gives utility guarantees for queries regarding global properties of the input matrix. Blum et al [44] approximate the input matrix with the PCA construction by adding $O(d^2)$ noise to the input. The work of McSherry and Mironov [111] (inspired by the Netflix prize compe-

tition) defines neighboring databases as a change in a single entry, and introduces $O(k^2)$ noise while outputting a rank-k approximation of the input. Kapralov and Talwar [99] also give an algorithm for releasing the PCA of a given matrix while preserving ϵ-differential privacy (the case $\delta = 0$).

The body of work on the JL transform is by now so extensive that only a book may survey it properly [139]. Its proof have been revised numerous times and it is known to work under various projections – using Gaussians [58], using random unit-length vectors [69], using random $\{-1, 0, 1\}$ entries [3], or sparse matrices [55]. Our analysis derives its privacy guarantees from applying the variant of the JL in which all entries are picked independently from a normal Gaussian.

In the context of differential privacy, the JL lemma has been used to reduce dimensionality of an input prior to adding noise or other forms of privacy preservation. Blum et al [43] gave an algorithm that outputs a sanitized dataset for learning large-margin classifiers by appealing to JL related results of [26]. Hardt and Roth [82] gave a privacy preserving version of an algorithm of [78] that uses randomize projections onto the image space of a given matrix. The way the JL lemma was applied in these works is very different than the way we use it.

5.2 Basic Definitions, Preliminaries and Notations

Privacy and utility. In this work, we deal with two types of inputs: $[0, 1]$-weighted graphs over n nodes and $n \times d$ real matrices. (We treat $w_{a,b} = 0$ as no edge between a and b). Trivially extending the definition in [119, 100], two weighted n-nodes graphs G and G' are called *neighbors* if they differ on the weight of a single edge (a, b). Like in [82], two $n \times d$-matrices are called *neighbors* if all the coordinates on which A and A' differ lie on a single row i, s.t. $\|A_{(i)} - A'_{(i)}\|^2 \leq 1$, where $A_{(i)}$ denotes the i-th row of A.

For each type of input we are interested in answering a different type of query. For graphs, we are interesting in *cut-queries*: given a nonempty strict subset S of the vertices of the graph, we wish to know what is the total weight of edges crossing the $(S, \bar{S})$-cut. We denote this as $\Phi_G(S) = \sum_{u \in S, v \notin S} w_{u,v}$.

Definition 5.1. *We say an algorithm ALG gives a (η, τ, ν)-approximation for cut queries, if for every nonempty S it holds that*

$$\mathbf{Pr}\left[(1-\eta)\Phi_G(S) - \tau \leq \mathsf{ALG}(S) \leq (1+\eta)\Phi_G(S) + \tau\right] \geq 1 - \nu$$

For $n \times d$ matrices, we are interested in *directional variance queries*: given a unit-

length direction x, we wish to know what's the variance of A along the x direction: $\Phi_A(x) = x^\intercal A^\intercal A x$. (Our algorithm normalizes A s.t. the mean of its n rows is 0.)

Definition 5.2. *We say an algorithm* ALG *gives a* (η, τ, ν)*-approximation for directional variance queries, if for every unit-length vector* x *it holds that*

$$\mathbf{Pr}\left[(1-\eta)\Phi_A(x) - \tau \leq \mathsf{ALG}(x) \leq (1+\eta)\Phi_A(x) + \tau\right] \geq 1 - \nu$$

Finally, we conclude these Gaussian preliminaries with the famous Johnson-Lindenstrauss Lemma, our main tool in this paper.

Theorem 5.3 (The Johnson Lindenstrauss transform [93])**.** *Fix any* $0 < \eta < 1/2$*. Let* M *be a* $r \times m$ *matrix whose entries are iid samples from* $\mathcal{N}(0,1)$*. Then* $\forall x \in \mathbb{R}^m$*.*

$$\mathbf{Pr}_M\left[(1-\eta)\|x\|^2 \leq \frac{1}{r}\|Mx\|^2 \leq (1+\eta)\|x\|^2\right] \geq 1 - 2\exp(-\eta^2 r/8)$$

Additional notations. We denote by e_a the indicator vector of a. We denote by $e_{a,b} = e_a - e_b$. It follows that the $n \times n$ matrix $L_{a,b} = e_{a,b}e_{a,b}^\intercal$ is the matrix whose projection over coordinates a, b is $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$, while every other entry is 0. We also denote $E_{a,b}$ as the $\binom{n}{2} \times n$ matrix, whose rows are all zeros except for the row indexed by the (a,b) pair, which is $e_{a,b}^\intercal$. Observe: $L_{a,b} = e_{a,b}e_{a,b}^\intercal = E_{a,b}^\intercal E_{a,b}$.

5.3 Publishing a Perturbed Laplacian

5.3.1 The Johnson-Lindenstrauss Algorithm

We now show that the Johnson Lindenstrauss transform preserves differential privacy. We first detail our algorithm, then analyze it.

Algorithm 2: Outputting the Laplacian of a Graph while Preserving Differential Privacy

Input: A n-node graph G, parameters: $\epsilon, \delta, \eta, \nu > 0$
Output: A Laplacian of a graph $\tilde{L}$

1 Set $r = \frac{8\ln(2/\nu)}{\eta^2}$, and $w = \frac{\sqrt{32r\ln(2/\delta)}}{\epsilon}\ln(4r/\delta)$
2 For every $u \neq v$, set $w_{u,v} \leftarrow \frac{w}{n} + \left(1 - \frac{w}{n}\right) w_{u,v}$.
3 Pick a matrix M of size $r \times \binom{n}{2}$, whose entries are iid samples of $\mathcal{N}(0,1)$.
4 **return** $\tilde{L} = \frac{1}{r} E_G^\intercal M^\intercal M E_G$

Algorithm 3: Approximating $\Phi_G(S)$

Input: A non empty $S \subsetneq V(G)$, parameters n, w and Laplacian $\tilde{L}$ from Algorithm 2.

return $R(S) = \frac{1}{1-\frac{w}{n}}\left(\mathbf{1}_S^\intercal \tilde{L}\mathbf{1}_S - w\frac{s(n-s)}{n}\right)$

Theorem 5.4. *Algorithm 2 preserves (ϵ, δ)-differential privacy w.r.t to edge changes in G.*

Theorem 5.5. *For every $\eta, \nu > 0$, given a nonempty S of size $s < n$, Algorithm 3 gives a (η, τ, ν)-approximation for the cut $\Phi_G(S)$, for $\tau = O\big(s \cdot \frac{\sqrt{\ln(1/\delta)\ln(1/\nu)}}{\epsilon}\ln(\ln(1/\nu)/\eta^2\delta)\big)$.*

Corollary 5.6. *For every $\epsilon, \delta, \eta, \nu > 0$ and any predefined set of k cut-queries, there exists a (ϵ, δ)-differentially private mechanism that w.p. $\geq 1 - \nu$ approximates each cut query up to a multiplicative factor of $1 \pm \eta$ and an additive error of $\tau = O\big(s \cdot \frac{\sqrt{\ln(1/\delta)\ln(k/\nu)}}{\epsilon}\ln(\ln(k/\nu)/\eta^2\delta)\big)$.*

Clearly, once Algorithm 2 publishes $\tilde{L}$, any user interested in estimating $\Phi_G(S)$ for some nonempty $S \subsetneq V(G)$ can run Algorithm 3 on her own. Also, observe that w is independent of n, which we think of as large number, so we assume thoughout the proofs of both theorems that both $\frac{w}{n}, \frac{1}{w}$ are $< 1/2$. Now, the proof of Theorem 5.5 is immediate from the JL Lemma.

Proof of Theorem 5.5. Let us denote G as the input graph for Algorithm 2, and H as the graph resulting from the changes in edge-weights Algorithm 2 makes. Therefore,

$$L_H = L_{\frac{w}{n}K_n} + L_{(1-\frac{w}{n})G} = \frac{w}{n}L_{K_n} + \left(1 - \frac{w}{n}\right) L_G$$

Fix S. The JL Lemma (Theorem 5.3) assures us that w.p. $\geq 1 - \nu$ we have

$$(1-\eta)\mathbf{1}_S^\intercal L_H \mathbf{1}_S \leq \mathbf{1}_S^\intercal \tilde{L} \mathbf{1}_S \leq (1+\eta)\mathbf{1}_S^\intercal L_H \mathbf{1}_S$$

The proof now follows from basic arithmetic and the value of w.

$$\begin{aligned} R(S) &\leq \frac{1}{1-\frac{w}{n}} \left((1+\eta)\mathbf{1}_S^\intercal L_H \mathbf{1}_S - w\frac{s(n-s)}{n} \right) \\ &= \frac{1}{1-\frac{w}{n}} \left((1+\eta)\frac{w}{n}s(n-s) + (1+\eta)(1-\frac{w}{n})\mathbf{1}_S^\intercal L_G \mathbf{1}_S - w\frac{s(n-s)}{n} \right) \\ &\leq (1+\eta)\Phi_G(S) + \frac{1}{1-\frac{w}{n}}\eta w \cdot s = (1+\eta)\Phi_G(S) + \tau \end{aligned}$$

where $\tau \leq 2\eta w \cdot s$. The lower bound is obtained exactly the same way. □

Comment. The guarantee of Theorem 5.5 is not to be mistaken with a weaker guarantee of providing a good approximation to *most* cut-queries. Theorem 5.5 guarantees that any set of k predetermined cuts is well-approximated by Algorithm 3, assuming Algorithm 2 sets $\nu < 1/2k$. In contrast, giving a good approximation to most cuts can be done by a very simple (and privacy preserving) algorithm: by outputting the number of edges in the graph (with small Laplacian noise). Afterall, we expect a cut to have $\frac{m}{\binom{n}{2}}s(n-s)$ edges crossing it. In other words, the high probability of success is over internal randomness in the algorithm and not randomness in the choice of cuts.

We turn our attention to the proof of Theorem 5.4. We fix any two graphs G and G', which differ only on a single edge, (a, b). We think of (a, b) as an edge in G' which isn't present in G, and in the proof of Theorem 5.4, we identify G with the manipulation Algorithm 2 performs over G, and assume that the edge (a, b) is present in both graphs, only it has weight $\frac{w}{n}$ in G, and weight 1 in G'. Clearly, this analysis carries on for a smaller change, when the edge (a, b) is present in both graphs but with different weights. (Recall, we assume all edge weights are bounded by 1.)

Now, the proof follows from assuming that Algorithm 2 outputs the matrix $O = ME_G$, instead of $\tilde{L} = \frac{1}{r}O^\intercal O$. (Clearly, outputting O allows one to reconstruct $\tilde{L}$.) Observe that

O is composed of r identically distributed rows: each row is created by sampling a $\binom{n}{2}$-dimensional vector Y whose entries $\sim \mathcal{N}(0,1)$, then outputting $Y^\intercal E_G$. Therefore, we prove Theorem 5.4 by showing that each row maintain (ϵ_0, δ_0)-differential privacy, for the right parameters ϵ_0, δ_0. To match standard notion, we transpose row vectors to column vectors, and compare the distributions $E_G^\intercal Y$ and $E_{G'}^\intercal Y$.

Claim 5.7. *Set* $\epsilon_0 = \frac{\epsilon}{\sqrt{4r\ln(2/\delta)}}, \delta_0 = \frac{\delta}{2r}$. *Then,*

$$\forall x, \;\; \mathsf{PDF}_{E_G^\intercal Y}(x) \leq \;\; e^{\epsilon_0}\mathsf{PDF}_{E_{G'}^\intercal Y}(x) \tag{5.1}$$

Denote $S = \{x : \; \mathsf{PDF}_{E_G^\intercal Y}(x) \; \geq \;\; e^{-\epsilon_0}\mathsf{PDF}_{E_{G'}^\intercal Y}(x)\}$. *Then*

$$\mathbf{Pr}[S] \geq 1 - \delta_0 \tag{5.2}$$

Proof of Theorem 5.4 based on Claim 5.7. Apply the composition theorem of [65] for r iid samples each preserving (ϵ_0, δ_0)-differential privacy. □

To prove Claim 5.7, we denote $X = E_G^\intercal Y$ and $X' = E_{G'}^\intercal Y$. From the preliminaries it follows that X is a multivariate Gaussian distributed according to $\mathcal{N}(0, E_G^\intercal I_{\binom{n}{2}\times\binom{n}{2}} E_G) = \mathcal{N}(0, L_G)$, and similarly, $X' \sim \mathcal{N}(0, L_{G'})$. In order to analyze the two distributions, $\mathcal{N}(0, L_G)$ and $\mathcal{N}(0, L_{G'})$, we now discuss several of the properties of L_G and $L_{G'}$, then turn to the proof of Claim 5.7.

First, it is clear from definition that the all ones vector, $\mathbf{1}$, belongs to the kernel space of E_G and $E_{G'}$, and therefore to the kernel space of L_G and $L_{G'}$. Next, we establish a simple fact.

Fact 5.8. *If* G *is a graph s.t. for every* $u \neq v$ *we have that* $w_{u,v} > 0$*, then* $\mathbf{1}$ *is the only vector in the kernel space of* E_G *and* L_G.

Proof. Any non-zero $x \perp \mathbf{1}$ has at least one positive coordinate and one negative coordinate, thus the non-negative sum $\|E_G x\|^2 = x^\intercal L_G x = \sum_{u\neq v} w_{u,v}(x_u - x_v)^2$ is strictly positive. □

Therefore, the kernel space of both L_G and of $L_{G'}$ is exactly the 1-dimensional span of the $\mathbf{1}$ vector (for every possible outcome y of Y we have that $E_G^\intercal y \cdot \mathbf{1} = E_{G'}^\intercal y \cdot \mathbf{1} = 0$). Alternatively, both X and X' have support which is exactly $\mathcal{V} = \mathbf{1}^\perp$. Hence, we only need to prove the inequalities of Claim 5.7 for $x \in \mathcal{V}$. Secondly, observe that $L_{G'} = L_G + (1 - \frac{w}{n})L_{a,b}$. Therefore, it holds that for every $x \in \mathbb{R}^n$ we have $x^\intercal L_{G'} x = x^\intercal L_G x +$

$(1-\frac{w}{n})(x_a-x_b)^2 \geq x^\intercal L_G x$. In other words, $L_G \preceq L_{G'}$, a fact that yields several important corollaries.

We now introduce notation for the Singular Value Decomposition of both L_G and $L_{G'}$. We denote $E_G^\intercal = U\Sigma V^\intercal$ and $E_{G'}{}^\intercal = U'\Lambda V'^\intercal$, resulting in $L_G = U\Sigma^2 U^\intercal, L_{G'} = U'\Lambda^2 U'^\intercal, L_G^\dagger = U\Sigma^{-2}U^\intercal$ and $L_{G'}^\dagger = U'\Lambda^{-2}U'^\intercal$. We denote the singular values of L_G as $\sigma_1^2 \geq \ldots \geq \sigma_{n-1}^2 > \sigma_n^2 = 0$, and the singular values of $L_{G'}$ as $\lambda_1^2 \geq \ldots \geq \lambda_{n-1}^2 > \lambda_n^2 = 0$. Weyl's inequality 2.2 allows us to deduce the following fact.

Fact 5.9. *Since $L_G \preceq L_{G'}$ then for every i we have that $\lambda_i^2 \geq \sigma_i^2$.*

In addition, since Algorithm 2 alters the input graphs s.t. the complete graph $\frac{w}{n}L_{K_n}$ is contained in G, then it also holds that $\frac{w}{n}L_{K_n} \preceq L_G$, and so Fact 5.9 gives that for every $1 \leq i \leq n-1$ we have that $\sigma_i^2 \geq w = \frac{w}{n}\cdot n$. (It is simple to see that the eigenvalues of K_n are $\{n, n, \ldots, n, 0\}$.) Furthermore, as $L_{G'} = L_G + (1-\frac{w}{n})L_{a,b}$ and the singular values of $L_{a,b}$ are $\{2, 0, 0, \ldots, 0\}$, then we have that

$$\sum_i \lambda_i^2 = tr(L_{G'}) \leq tr(L_G) + tr\left((1-\frac{w}{n})L_{a,b}\right) \leq \sum_i \sigma_i^2 + 2$$

Another fact we can deduce from $L_G \preceq L_{G'}$, is the following. It is immediate application of Fact 2.1.

Fact 5.10. *Since the kernels of L_G and of $L_{G'}$ are identical, then for every x it holds that $x^\intercal L_{G'}^\dagger x \leq x^\intercal L_G^\dagger x$.*

Having established the above facts, we can turn to the proof of privacy.

Proof of Claim 5.7. We first prove the upper bound in (5.1). As mentioned, we focus only on $x \in \mathcal{V} = \mathbf{1}^\perp$, where

$$\mathsf{PDF}_{E_G^\intercal Y}(x) = \left((2\pi)^{n-1}\tilde{\det}(L_G)\right)^{-1/2}\exp(-\frac{1}{2}x^\intercal L_G^\dagger x)$$
$$\mathsf{PDF}_{E_{G'}^\intercal Y}(x) = \left((2\pi)^{n-1}\tilde{\det}(L_{G'})\right)^{-1/2}\exp(-\frac{1}{2}x^\intercal L_{G'}^\dagger x)$$

As noted above, we have that for every x it holds that $x^\intercal L_{G'}^\dagger x \leq x^\intercal L_G^\dagger x$, so $\exp(-\frac{1}{2}x^\intercal L_G^\dagger x) \leq \exp(-\frac{1}{2}x^\intercal L_{G'}^\dagger x)$. It follows that for every x we have that $\frac{\mathsf{PDF}_{E_G^\intercal Y}(x)}{\mathsf{PDF}_{E_{G'}^\intercal Y}(x)} \leq \left(\frac{\tilde{\det}(L_{G'})}{\tilde{\det}(L_G)}\right)^{1/2} =$

$\left(\prod_{i=1}^{n-1}\frac{\lambda_i^2}{\sigma_i^2}\right)^{1/2}$. Denoting $\Delta_i = \lambda_i^2 - \sigma_i^2 \geq 0$, and recalling that $\sum_i \Delta_i \leq 2$ and that $\forall i, \sigma_i^2 \geq w$ it holds that

$$\frac{\mathsf{PDF}_{E_G^\intercal Y}(x)}{\mathsf{PDF}_{E_{G'}^\intercal Y}(x)} \leq \sqrt{\prod_{i=1}^{n-1}\left(1+\frac{\Delta_i}{\sigma_i^2}\right)} \leq \exp\left(\frac{1}{2w}\sum_i \Delta_i\right) \leq e^{\frac{1}{w}} \leq e^{\frac{\epsilon}{\sqrt{4r\ln(2/\delta)}}} = e^{\epsilon_0}$$

We now turn to the lower bound of (5.2). We start with analyzing the term $x^\intercal L_G^\dagger x$ that appears in $\mathsf{PDF}_{E^\intercal Y}(x)$. Again, we emphasize that $x \in \mathcal{V}$, justifying the very first equality below.

$$\begin{aligned} x^\intercal L_G^\dagger x &= x^\intercal L_G^\dagger L_{G'} L_{G'}^\dagger x = x^\intercal L_G^\dagger \left(L_G + (1-\frac{w}{n})L_{ab}\right) L_{G'}^\dagger x \\ &= x^\intercal L_{G'}^\dagger x \ + \ (1-\frac{w}{n}) x^\intercal L_G^\dagger L_{a,b} L_{G'}^\dagger x \\ &= x^\intercal L_{G'}^\dagger x \ + \ (1-\frac{w}{n}) x^\intercal L_G^\dagger e_{a,b} \ \cdot \ e_{a,b}^\intercal L_{G'}^\dagger x \end{aligned}$$

Therefore, if we show that

$$\mathbf{Pr}_{x\sim E_G^\intercal Y}\left[x^\intercal L_G^\dagger e_{a,b} \ \cdot \ e_{a,b}^\intercal L_{G'}^\dagger x > \frac{2}{1-\frac{w}{n}}\epsilon_0\right] < \delta_0 \tag{5.3}$$

then it holds that w.p. $> 1-\delta_0$ we have

$$\frac{\mathsf{PDF}_{E_G^\intercal Y}(x)}{\mathsf{PDF}_{E_{G'}^\intercal Y}(x)} \geq 1 \cdot \exp\left(-\frac{1}{2}x^\intercal(L_G^\dagger - L_{G'}^\dagger)x\right) \geq \exp\left(-\frac{1-\frac{w}{n}}{2}x^\intercal L_G^\dagger e_{a,b} \ \cdot \ e_{a,b}^\intercal L_{G'}^\dagger x\right) \geq e^{-\epsilon_0}$$

which proves the lower bound of (5.2). We turn to proving (5.3).

Denote $term_1 = e_{a,b}^\intercal L_G^\dagger x$ and $term_2 = e_{a,b}^\intercal L_{G'}^\dagger x$. Since $x = E_G^\intercal y$ where $y \sim Y$ then $term_i$ is distributed like $vec_i^\intercal Y$ where $vec_1 = E_G L_G^\dagger e_{a,b}$ and $vec_2 = E_G L_{G'}^\dagger e_{a,b}$. The naïve bound, $\|vec_1\| \leq \|E_G\| \, \|L_G^\dagger\| \|e_{a,b}\|$ gives a bound on the size of vec_1 which is dependent on the ratio $\frac{\sigma_1}{\sigma_{n-1}^2}$. We can improve the bound, on both $\|vec_1\|$ and $\|vec_2\|$, using the SVD of E_G and $E_{G'}$.

$$\begin{aligned} \|vec_1\| &= \|E_G L_G^\dagger e_{a,b}\| = \|V\Sigma U^\intercal U \Sigma^{-2} U^\intercal e_{a,b}\| = \|V\Sigma^{-1}U^\intercal e_{a,b}\| \\ &\leq \|V\| \, \|\Sigma^{-1}\| \, \|U\| \, \|e_{a,b}\| = 1 \cdot \sigma_{n-1}^{-1} \cdot 1 \cdot \sqrt{2} = \frac{\sqrt{2}}{\sqrt{w}} \\ \|vec_2\| &= \|E_G L_{G'}^\dagger e_{a,b}\| = \|(E_{G'} - (1-\frac{w}{n})E_{a,b}) L_{G'}^\dagger e_{a,b}\| < \|E_{G'} L_{G'}^\dagger e_{a,b}\| + \|E_{a,b} L_{G'}^\dagger e_{a,b}\| \end{aligned}$$

$$\overset{(*)}{\leq} \lambda_{n-1}^{-1} \cdot \sqrt{2} + \|E_{a,b} L_{G'}^{\dagger} e_{a,b}\| \overset{(**)}{=} \frac{\sqrt{2}}{\sqrt{w}} + e_{a,b}^{\intercal} L_{G'}^{\dagger} e_{a,b}$$
$$\leq \frac{\sqrt{2}}{\sqrt{w}} + \frac{2}{w} = \frac{\sqrt{2}}{\sqrt{w}} \left(1 + \frac{\sqrt{2}}{\sqrt{w}}\right)$$

where the bound in $(*)$ is derived just like in vec_1 (using $E_{G'} L_{G'}^{\dagger} e_{a,b} = V' \Lambda U'^{\intercal} U' \Lambda^{-2} U'^{\intercal} e_{a,b}$) , and the equality in $(**)$ follows from the fact that all coordinates in the vector $E_{a,b} L_{G'}^{\dagger} e_{a,b}$ are zero, except for the coordinate indexed by the (a, b) pair.

We now use the fact that $term_1$ and $term_2$ are both linear combinations of i.i.d $\mathcal{N}(0, 1)$ random variables. Therefore for $i = 1, 2$ we have that $term_i \sim \mathcal{N}(0, \|vec_i\|^2)$ so $\mathbf{Pr}[|term_i| > \sqrt{\log(2/\delta_0)} \|vec_i\|] \leq e^{-\frac{\|vec_i\|^2 \log(2/\delta_0)}{\|vec_i\|^2}} < \frac{\delta_0}{2}$. It follows that w.p $> 1 - \delta_0$ both $|term_1| < \sqrt{\log(2/\delta_0)} \sqrt{\frac{2}{w}}$ and $|term_2| \leq \sqrt{\log(2/\delta_0)} \sqrt{\frac{4}{w}}$, so $term_1 \cdot term_2 \leq \sqrt{8} \log(2/\delta_0)/w$. Plugging in the value of w, we have that $\mathbf{Pr}[term_1 \cdot term_2 \leq 2\epsilon_0] \geq 1 - \delta_0$ which concludes the proof of (5.3) and of Claim 5.7. □

5.3.2 Discussion and Comparison with Other Algorithms

Recently, Gupta et al [77] have also considered the problem of answering cut-queries while preserving differential privacy, examining both an iterative database construction approach (e.g., based on the multiplicative-weights method) and a randomized-response approach. Here, we compare this and other methods to our algorithm. We compare them along several axes: the dependence on n and s (number of vertices in G and in S resp.), the dependence on ϵ, and the dependence on k – the number of queries answered by the mechanism. Other parameters are omitted. The bottom line is that for a long non-adaptive query sequence, our approach dominates in the case that $s = o(n)$. The results are summarized in Table 5.1.

Note, comparing the dependence on k for interactive and non-interactive mechanisms is not straight-forward. In general, non-interactive mechanisms are more desirable than interactive mechanisms, because interactive mechanisms require a central authority that serves as the only way users can interact with the database. However, interactive mechanisms can answer k *adaptively chosen* queries. In order for non-interactive mechanisms to do so, they have to answer correctly on $\min\{\exp(O(k)), 2^n\}$ queries. This is why outputting a sanitized database is often considered a harder task than interactively answering user queries. We therefore compare answering k adaptively chosen queries for interactive mechanisms, and k *predetermined* queries for non-interactive mechanism.

5.3.2.1 Naïvely Adding Laplace Noise

The most basic of all differentially private mechanisms is the classical Laplace mechanism which is interactive. For a given $\tilde{\epsilon}$, A user poses a cut-query S and the mechanism replies with $\Phi_G(S) + Lap(0, \tilde{\epsilon}^{-1})$ (since the global sensitivity of cut-queries is 1). The composition theorem of [65] assures us that for k queries we preserve $(O(\sqrt{k}\tilde{\epsilon}), \delta)$-privacy. As a result, if we wish to preserve (ϵ, δ)-differential privacy for a given set of k cut queries, we must set $\tilde{\epsilon} \geq \epsilon/\sqrt{k}$ and answer each query with additive error of roughly $\sqrt{k}/\epsilon$. So the mechanism completely obfuscates the true answer if $k \geq n^4$.

5.3.2.2 The Randomized Response Mechanism

The "Randomized Response" algorithm perturbs the edges of a graph in a way that allows us to publish the result and still preserve privacy. Given G, the Randomized Response algorithm constructs a weighted graph H where for every $u, v \in V(G)$, the weight of the edge (u, v) in H, denoted $w'_{u,v}$, is chosen independently to be either 1 or -1. Each edge picks its weight independently, s.t. $\mathbf{Pr}[w'_{u,v} = 1] = \frac{1+\epsilon w_{u,v}}{2}$ and $\mathbf{Pr}[w'_{u,v} = -1] = \frac{1-\epsilon w_{u,v}}{2}$. Clearly, this algorithm maintains ϵ-differential edge privacy: two neighboring graphs differ on a single edge, (a, b), and obviously

$$\mathbf{Pr}[w'_{a,b} = 1 \mid w_{a,b} = 1] \leq (1 + \epsilon)\mathbf{Pr}[w'_{a,b} = 1 \mid w_{a,b} = 0]$$

In addition, it is also evident that for every nonempty $S \subsetneq V(G)$, we have that $\mathbf{E}[\sum_{u\in S, v\in\bar{S}} w'_{u,v}] = \epsilon \sum_{u\in S, v\notin S} w_{u,v} = \epsilon\Phi_G(S)$, yet the variance of this r.v. is $\Omega(s(n-s))$. Therefore, a classical Hoeffding-type bound gives that for any nonempty $S \subsetneq V(G)$ we have that for every $0 < \nu < 1/2$,

$$\mathbf{Pr}\left[\left|\frac{1}{\epsilon}\sum_{u\in S, v\in\bar{S}} w'_{u,v} - \Phi_G(S)\right| > \frac{\sqrt{2\log(1/\nu)s(n-s)}}{\epsilon}\right] \leq 2\nu$$

Observe that while $\sqrt{s(n-s)}$ is a comparable with s when $s = \Omega(n)$, there are cuts (namely, cuts with $s = O(1)$) where $\sqrt{\frac{n-s}{s}} = \Omega(\sqrt{n})$. More generally, the additive noise of Randomized Response is a factor $\sqrt{n/s}$ worse than our algorithm. We comment that the Randomized Response algorithm can also be performed in a distributed fashion, and in contrast to our algorithm, it has no multiplicative error. In addition, the above analysis holds for any linear combination of edge, not just the $s(n-s)$ potential edges that cross the $(S, \bar{S})$ cut. So given $E' \subset E(G)$ it is possible to approximate $\sum_{e\in E'} w_e$ up

to $\pm\frac{\sqrt{|E'|\log(1/\nu)}}{\epsilon}$ w.p. $\geq 1-2\nu$. In particular, for queries regarding an (S,T)-cut (where S,T are two disjoint subsets of vertices) we can estimate the error up to $\pm\frac{\sqrt{|S||T|\log(1/\nu)}}{\epsilon}$. We also comment that the version of Randomized Response presented here differs slightly from the version of [77]. In particular, it is possible to address their concern regarding outputting a sanitized graph with non-negative weights by an affine transformation taking $\{-1,1\}\to\{0,1\}$.

5.3.2.3 Exponential Mechanism / BLR

The exponential mechanism [112, 43] is a non-interactive privacy preserving mechanism, which is typically intractable. To implement it for cut-queries one needs to (a) specify a range of potential outputs and (b) give a scoring function over potential outputs s.t. a good output's score is much higher than all bad outputs' scores.

One such set of potential outputs is derived from edge-sparsifiers. Given a graph G we say that H is an edge-sparsifier for G if for any nonempty $S\subsetneq V(G)$ it holds that $\Phi_H(S)\in(1\pm\eta)\Phi_G(S)$. There's a rich literature on sparsifiers (see [38, 133, 132]), and the current best known construction [33] gives a (weighted) sparsifier with $O(n/\eta^2)$ edges with all edge-weights $\leq \mathrm{poly}(n)$. By describing every edge's two endpoints and weight, we have that such edge-sprasifiers can be described using $O(n\log(n))$ bits (omitting dependence on η). Thus, the set of all sparsifiers is bounded above by $\exp(O(n\log(n)))$. Given an input graph G and a weighted graph H, we can score H using $q(G,H)=\max_S\left\{\min_{\alpha:\ |\alpha-1|\leq\eta}|\Phi_H(S)/\alpha-\Phi_G(S)|\right\}$. Observe that if we change G to a neighboring graph G', then the score changes by at most 1.

Putting it all together, we have that given input G the exponential mechanism gives a score of $e^{-\epsilon q(G,H)/2}$ to each possible output. The edge-sparsifier of G gets score of 1, whereas every graph with $q(G,H)>\tau$ gets a score of $e^{-\epsilon\tau/2}$. So if we wish to claim we output a graph whose error is $>\tau$ w.p. at most ν, then we need to set $\exp(n\log(n)-\epsilon\tau/2)\leq\nu$. It follows that τ is proportional to $n\log(n)/\epsilon$. Note however that the additive error of this mechanism is independent of the number of queries it answers correctly.

We comment that even though we managed to find a range of size $2^{O(n\log(n))}$, it is possible to show that the range of the mechanism has to be $2^{\Omega(n)}$. (Fix $\alpha<1/2$ and think of a set of inputs $\mathcal{G}$ where each $G\in\mathcal{G}$ has $n/2$ vertices with degree n^α and $n/2$ vertices with degree $n^{2\alpha}$. Preserving all cuts of size 1 up to $(1\pm\eta)$ requires our output to have vertices of degree $>(1-\eta)n^{2\alpha}$ and vertices of degree $<(1+\eta)n^\alpha$. Therefore, by representing vertices of high- and low-degree using a binary vector, there exists an injective mapping of balanced $\{0,1\}^n$-vectors onto the set of potential outputs.) Thus,

unless one can devise a scoring function of lower sensitivity, the exponential mechanism must have additive error proportional to n/ϵ.

5.3.2.4 The Multiplicative Weights Mechanism

The very elegant Multiplicative Weights mechanism of Hardt and Rothblum [80] can be adapted as well for answering cut queries. In the Multiplicative Weights mechanism, a database is represented by a histogram over all N "types" of individuals that exist in a certain universe. In our case, each pair of vertices is a type, and each entry in the database is an edge detailing its weight. Thus, $N = \binom{n}{2}$ and the database length $= |E|$,[3] and each query S corresponds to taking a dot-product between this histogram the $\binom{n}{2}$-length binary vector indicating the edges that cross the cut. Plugging these parameters into the main theorem of [80], we get an adaptive mechanism that answers k queries with additive noise of $\tilde{O}(\sqrt{|E|}\log(k)/\epsilon)$.

We should mention that the Multiplicative Weights mechanism, in contrast to ours, always answers correctly with no multiplicative error and can deal with k adaptively chosen queries. Furthermore, it allows one to answer any linear query on the edges, not just cut-queries and in particular answer (S,T)-cut queries. However, its additive error is bigger than ours, and should we choose to set $k = 2^n$ (meaning, answering all cut-queries) then its additive error becomes $\tilde{O}(n\sqrt{|E|}/\epsilon)$ (in contrast to our $O(s\sqrt{n}/\epsilon)$).

Gupta et al [77] have improved on the bounds on the Multiplicative Weights mechanism by generalizing it as a "Iterative Database Construction" mechanism, and providing a tighter analysis of it. In particular, they have reduced the dependency on ϵ to $1/\sqrt{\epsilon}$. Overall, their additive error is $\tilde{O}(\sqrt{|E|\log(k)}/\sqrt{\epsilon})$, which for the case of all cut-queries is $\tilde{O}(\sqrt{n|E|/\epsilon})$.

5.3.2.5 Our Algorithm

Clearly, our algorithm is non-interactive. As such, if we wish to answer correctly w.h.p. a set of k *predetermined* queries, we set $\nu' = \nu/k$, and deduce that the amount of noise added to each query is $O(s\sqrt{\log(k)}/\epsilon)$. So, if we wish to answer all 2^n cut queries correctly, our noise is set to $\tilde{O}(s\sqrt{n}/\epsilon)$. An interesting observation is that in such a case we aim to answer all 2^n queries, we generate a iid normal matrix of size $r \times n$ where

[3]Observe that it is not possible to assume $|E| = O(n)$ using sparsifiers, because sparsifiers output a *weighted* graph with edge-weights $O(n)$. Since the Multiplicative Weights mechanism views the database as a histogram the overall resolution of the problem remains roughly n^2 in the worst case.

Method	Additive Error for any k	Additive Error for all Cuts	Multi-plicative Error?	Inter-active?	Tract-able?	Comments
Laplace Noise	$O(\sqrt{k}/\epsilon)$	$O(2^{n/2}/\epsilon)$	✗	✓	✓	
Randomized Response	$O(\sqrt{sn\log(k)}/\epsilon)$	$O(n\sqrt{s}/\epsilon)$	✗	✗	✓	Can be distributed; answers (S,T)-cut queries
Exponential Mechanism	$O(n\log(n)/\epsilon)$	$O(n\log(n)/\epsilon)$	✓	✗	✗	Error ind. of k
MW IDC	$\tilde{O}(\sqrt{\|E\|\log(k)}/\epsilon)$ $\tilde{O}(\sqrt{\|E\|\log(k)}/\epsilon)$	$\tilde{O}(n\sqrt{\|E\|}/\epsilon)$ $\tilde{O}(\sqrt{n\|E\|}/\epsilon)$	✗	✓	✓	Answers (S,T)-cut queries
JL	$O(s\sqrt{\log(k)}/\epsilon)$	$\tilde{O}(s\sqrt{n})/\epsilon)$	✓	✗	✓	Can be distributed

Table 5.1: Comparison between mechanisms for answering cut-queries. ϵ – privacy parameter; n and $|E|$ – number of vertices and edges resp.; s – number of vertices in a query; k – number of queries.

$r > n$. Therefore, we now apply the JL transform to *increase* the dimensionality of the problem rather than decreasing it. This clearly sets privacy preserving apart from all other applications of the JL transform.

In addition, we comment that our algorithm can be implemented in a *distributed* fashion, where node i repeats the following procedure r times (where r is the number of rows in the matrix picked by Algorithm 2): First, i picks $n-i-1$ iid samples from $\mathcal{N}(0,1)$ and sends the j-th sample, x_j, to node $i+j$. Once node i receives $i-1$ values from nodes $1,2,\ldots,i-1$, it outputs the weighted sum $\sum_{j\neq i}(-1)^{\{j<i\}}x_j\left(\sqrt{\frac{w}{n}}+w_{i,j}(1-\sqrt{\frac{w}{n}})\right)$ (where $(-1)^{\{j<i\}}$ denotes -1 if $j<i$, or 1 otherwise).

5.4 Publishing a Covariance Matrix

5.4.1 The Algorithm

In this section, we are concerned with the question of allowing users to estimate the covariance of a given sample data along an arbitrary direction x. We think of our input as a $n \times d$ matrix A, and we maintain privacy w.r.t to changing the coordinates of a single row s.t. a vector v of size 1 is added to $A_{(i)}$. We now detail our algorithm for publishing the

covariance matrix of A. Observe that in addition to the variance, we can output $\mu = \frac{1}{n}A^\intercal \mathbf{1}$, the mean of all samples in A, in a differentially private manner by adding random Gaussian noise. (We merely output $\tilde{\mu} = \mu + \mathcal{N}(0, \frac{4\log(1/\delta)}{n^2\epsilon^2} I_{d\times d})$.) We denote by $I_{n\times d}$ the $n \times d$ matrix whose main diagonal has 1 in each coordinate and all other coordinates are 0.

Algorithm 4: Outputting a Covariance Matrix while Preserving Differential Privacy

Input: A $n \times d$ matrix A. Parameters $\epsilon, \delta, \eta, \nu > 0$.

1 Set $r = \frac{8\ln(2/\nu)}{\eta^2}$ and $w = \frac{16\sqrt{r\ln(2/\delta)}}{\epsilon}\ln(16r/\delta)$.
2 Subtract the mean from A by computing $A \leftarrow A - \frac{1}{n}\mathbf{1}\mathbf{1}^T A$.
3 Compute the SVD of $A = U\Sigma V^\intercal$.
4 Set $A \leftarrow U(\sqrt{\Sigma^2 + w^2 I_{n\times d}})V^\intercal$.
5 Pick a matrix M of size $r \times n$ whose entries are iid samples of $\mathcal{N}(0,1)$.
6 **return** $\tilde{C} = \frac{1}{r}A^\intercal M^\intercal M A$.

Algorithm 5: Approximating $\Phi_A(x)$

Input: A unit-length vector x, parameter w and a Covariance matrix $\tilde{C}$ from Algorithm 4.

return $R(x) = x^\intercal \tilde{C} x - w^2$.

Theorem 5.11. *Algorithm 4 preserves (ϵ, δ)-differential privacy.*

Theorem 5.12. *Algorithm 5 is a (η, τ, ν)-approximation for directional variance queries, where $\tau = O\left(\frac{\ln(1/\delta)\ln(1/\nu)}{\epsilon^2\eta}\ln^2\left(\frac{\ln(1/\nu)}{\delta\eta^2}\right)\right)$.*

Proof of Theorem 5.12. Again, the proof is immediate from the JL Lemma, and straightforward arithmetics give that for every x w.p. $\geq 1-\nu$ we have that

$$(1-\eta)\Phi_A(x) - \eta w^2 \leq R(x) \leq (1+\eta)\Phi_A(x) + \eta w^2$$

so $\tau = \eta w^2$. □

Comment. We wish to clarify that Theorem 5.12 does *not* mean that we publish a matrix $\tilde{C}$ which is a low-rank approximation to $A^\intercal A$. It is also not a matrix on which one can compute an approximated PCA of A, *even if we set* $\nu = 1/\mathrm{poly}(d)$. The matrix $\tilde{C}$ should be thought of as a "test-matrix" – if you believe A has high directional variance along some direction x then you can test your hypothesis on $\tilde{C}$ and (w.h.p) get the good approximated answer. However, we do not guarantee that the singular values of $A^\intercal A$ and of $\tilde{C}$ are close or that the eigenvectors of $A^\intercal A$ and $\tilde{C}$ are comparable. (See discussion in Section 5.5.)

Proof of Theorem 5.11. Fix two neighboring A and A'. We often refer to the gap matrix $A'-A$ as E. Observe, E is a rank-1 matrix, which we denote as the outer-product $E = e_i v^\intercal$ (e_i is the indicator vector of row i and v is a vector of norm 1). As such, the singular values of E are exactly $\{1, 0, \ldots, 0\}$.[4]

The proof of the theorem is composed of two stages. The first stage is the simpler one. We ignore step 4 of Algorithm 4 (shifting the singular values), and work under the premise that both A and A' have singular values no less than w. In the second stage we denote B and B' as the results of applying step 4 to A and A' resp., and show what adaptations are needed to make the proof follow through.

Stage 1. We assume step 4 was not applied, and all singular values of A and A' are at least w.

As in the proof of Theorem 5.4, the proof follows from the assumption that Algorithm 4 outputs $O^\intercal = A^\intercal M$ (which clearly allows us to reconstruct $\tilde{C} = \frac{1}{r} O^\intercal O$). Again $O^\intercal$ is composed of r columns each is an iid sample from $A^\intercal Y$ where $Y \sim \mathcal{N}(0, I_{n\times n})$. We now give the analogous claim to Claim 5.7.

Claim 5.13. *Fix* $\epsilon_0 = \frac{\epsilon}{\sqrt{4r\ln(2/\delta)}}$ *and* $\delta_0 = \frac{\delta}{2r}$. *Denote* $S = \{x : e^{-\epsilon_0}\mathsf{PDF}_{A'^\intercal Y}(x) \leq PDF_{A^\intercal Y}(x) \leq e^{\epsilon_0}\mathsf{PDF}_{A'^\intercal Y}(x)\}$. *Then* $\mathbf{Pr}[S] \geq 1 - \delta_0$.

Again, the composition theorem of [65] along with the choice of r gives that overall we preserve (ϵ, δ)-differential privacy. □

Proof of Claim 5.13. The proof mimics the proof of Claim 5.7, but there are two subtle differences. First, the problem is simpler notation-wise, because A and A' both have full rank due to Algorithm 4. Secondly, the problem becomes more complicated and requires we use some heavier machinery, because the singular values of A' aren't necessarily bigger than the singular values of A. Details follow.

First, let us formally define the PDF of the two distributions. Again, we apply the fact that $A^\intercal Y$ and $A'^\intercal Y$ are linear transformations of $\mathcal{N}(0, I_{n\times n})$.

$$\mathsf{PDF}_{A^\intercal Y}(x) = \frac{1}{\sqrt{(2\pi)^d \det(A^\intercal A)}} \exp(-\frac{1}{2} x^\intercal (A^\intercal A)^{-1} x)$$

[4]For convenience, we ignore the part of the algorithm that subtracts the mean of the rows of A. Observe that if $E = A - A'$ then after subtracting the mean from each row, the difference between the two matrices is $\tilde{e_i}^\intercal v$ where $\tilde{e_i}$ is simply subtracting $1/n$ from each coordinate of e_i. Since $\|\tilde{e_i}\| < \|e_i\|$, this has no effect on the analysis.

$$\mathsf{PDF}_{A'^{\mathsf{T}}Y}(x) = \frac{1}{\sqrt{(2\pi)^d \det(A'^{\mathsf{T}}A')}} \exp(-\frac{1}{2}x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x)$$

Our proof proceeds as follows. First, we show

$$e^{-\epsilon_0/2} \leq \sqrt{\frac{\det(A'^{\mathsf{T}}A')}{\det(A^{\mathsf{T}}A)}} \leq e^{\epsilon_0/2} \tag{5.4}$$

Then we show that no matter whether we sample x from $A^{\mathsf{T}}Y$ or from $A'^{\mathsf{T}}Y$, we have that

$$\mathbf{Pr}_x \left[\frac{1}{2}\left|x^{\mathsf{T}}\left((A^{\mathsf{T}}A)^{-1} - (A'^{\mathsf{T}}A')^{-1}\right)x\right| \geq \epsilon_0/2\right] \leq \delta_0 \tag{5.5}$$

Clearly, combining both (5.4) and (5.5) proves the claim.

Let us prove (5.4). Denote the SVD of $A = U\Sigma V^{\mathsf{T}}$ and $A' = U'\Lambda V'^{\mathsf{T}}$, where the singular values of A are $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_d > 0$ and the singular values of A' are $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d > 0$. Therefore we have $A^{\mathsf{T}}A = V\Sigma^2V^{\mathsf{T}}$, $A'^{\mathsf{T}}A' = V'\Lambda^2V'^{\mathsf{T}}$ and also $(A^{\mathsf{T}}A)^{-1} = V\Sigma^{-2}V^{\mathsf{T}}$, $(A'^{\mathsf{T}}A')^{-1} = V'\Lambda^{-2}V'^{\mathsf{T}}$. Thus $\det(A^{\mathsf{T}}A) = \prod_{i=1}^d \sigma_i^2$ and $\det(A'^{\mathsf{T}}A') = \prod_{i=1}^d \lambda_i^2$.

This time, in order to bound the gap $\sum_i(\lambda_i^2 - \sigma_i^2)/\sigma_i^2$ it isn't sufficient to use the trace of the matrices. Instead, we invoke an application of Lindskii's theorem 2.4.

Fact 5.14 (Lindskii)**.** *For every k and every $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ we have that*

$$\sum_{j=1}^k \lambda_{i_j} \leq \sum_{j=1}^k \sigma_{i_j} + \sum_{i=1}^k sv_i(E)$$

where $\{sv_i(E)\}_{i=1}^n$ are the singular values of E sorted in a descending order.

As a corollary, because E has only 1 non-zero singular value, we denote $Big = \{i : \lambda_i > \sigma_i\}$ and deduce that $\sum_{i\in Big} \lambda_i - \sigma_i \leq 1$. Similarly, since the singular values of E and of $(-E)$ are the same, we have that $\sum_{i\notin Big} \sigma_i - \lambda_i \leq 1$. Using this, proving (5.4) is straight-forward:

$$\sqrt{\prod_i \tfrac{\lambda_i^2}{\sigma_i^2}} \leq \prod_{i\in Big}\left(1 + \frac{\lambda_i - \sigma_i}{\sigma_i}\right) \leq \exp\left(\frac{1}{w}\sum_{i\in Big} \lambda_i - \sigma_i\right) \leq e^{w^{-1}} \leq e^{\epsilon_0/2}$$

and similarly, $\sqrt{\prod_i \frac{\sigma_i^2}{\lambda_i^2}} \leq e^{\epsilon_0/2}$.

We turn to proving (5.5). We start with the following derivation.

$$\begin{aligned} x^\intercal(A^\intercal A)^{-1}x - x^\intercal(A'^\intercal A')^{-1}x &= x^\intercal(A^\intercal A)^{-1}(A'^\intercal A')(A'^\intercal A')^{-1}x - x^\intercal(A'^\intercal A')^{-1}x = \\ &= x^\intercal(A^\intercal A)^{-1}((A+E)^\intercal(A+E))(A'^\intercal A')^{-1}x - x^\intercal(A'^\intercal A')^{-1}x \\ &= x^\intercal(A^\intercal A)^{-1}(A^\intercal E + E^\intercal A')(A'^\intercal A')^{-1}x \end{aligned}$$

and using the SVD and denoting $E = e_i v^\intercal$, we get

$$\begin{aligned} x^\intercal(A^\intercal A)^{-1}x - x^\intercal(A'^\intercal A')^{-1}x &= x^\intercal \left(V\Sigma^{-1}U^\intercal\right) e_i \cdot v^\intercal \left(V'\Lambda^{-2}V'^\intercal\right) x \\ &+ x^\intercal \left(V\Sigma^{-2}V^\intercal\right) v \cdot e_i^\intercal \left(U'\Lambda^{-1}V'^\intercal\right) x \end{aligned}$$

So now, assume x is sampled from $A^\intercal Y$. (The case of $A'^\intercal Y$ is symmetric. In fact, the names A and A' are interchangeable.) That is, assume we've sampled y from $Y \sim \mathcal{N}(0, I_{n\times n})$ and we have $x = A^\intercal y = V\Sigma U^\intercal y$ and equivalently $x = (A'^\intercal - E^\intercal)y = V'\Lambda U'^\intercal y - v e_i^\intercal y$. The above calculation shows that

$$\left|x^\intercal(A^\intercal A)^{-1}x - x^\intercal(A'^\intercal A')^{-1}x\right| \le term_1 \cdot term_2 + term_3 \cdot term_4$$

where for $i = 1, 2, 3, 4$ we have $term_i = |vec_i \cdot y|$ and

$$\begin{aligned} vec_1 &= U\Sigma V^\intercal V\Sigma^{-1}Ue_i = e_i, && \text{so } \|vec_1\| = 1 \\ vec_2 &= U'\Lambda^{-1}V'^\intercal v - e_i v^\intercal V'\Lambda^{-2}V'^\intercal v, && \text{so } \|vec_2\| \le \frac{1}{\lambda_d} + \frac{1}{\lambda_d^2} \\ vec_3 &= U\Sigma^{-1}V^\intercal v, && \text{so } \|vec_3\| \le \frac{1}{\sigma_d} \\ vec_4 &= e_i - e_i v^\intercal V'\Lambda^{-1}U'^\intercal e_i, && \text{so } \|vec_4\| \le 1 + \frac{1}{\lambda_d} \end{aligned}$$

Recall that all singular values, both of A and A', are greater than w and that $vec_i \cdot y \sim \mathcal{N}(0, \|vec_i\|^2)$, so w.p. $\ge 1 - \delta_0$ we have that for every i it holds that $term_i \le \sqrt{\ln(4/\delta_0)}\|vec_i\|$ so

$$\left|x^\intercal(A^\intercal A)^{-1}x - x^\intercal(A'^\intercal A')^{-1}x\right| \le 2(\frac{1}{w} + \frac{1}{w^2})\ln(4/\delta_0) \le \frac{4\ln(4/\delta_0)}{w} \le \epsilon_0$$

this concludes the proof in our first stage.

Stage 2. We assume step 4 was applied, and denote $B = U(\sqrt{\Sigma^2 + w^2 I})V^\intercal$ and $B' = U'(\sqrt{\Lambda^2 + w^2 I})V'^\intercal$. We denote the singular values of B and B' as $\sigma_1^B \geq \sigma_2^B \geq \ldots \geq \sigma_d^B$ and $\lambda_1^B \geq \lambda_2^B \geq \ldots \geq \lambda_d^B$ resp. Observe that by definition, for every i we have $(\sigma_i^B)^2 = \sigma_i^2 + w^2$ and $(\lambda_i^B)^2 = \lambda_i^2 + w^2$.

Again, we assume we output $O^\intercal = B^\intercal Y$, and compare $X = B^\intercal Y$ to $X' = B'^\intercal Y$. The theorem merely requires Claim 5.13 to hold, and they, in turn, depend on the following two conditions.

$$e^{-\epsilon_0/2} \leq \sqrt{\frac{\det(B'^\intercal B')}{\det(B^\intercal B)}} \leq e^{\epsilon_0/2} \tag{5.6}$$

$$\mathbf{Pr}_x \left[\frac{1}{2} \left| x^\intercal \left((B^\intercal B)^{-1} - (B'^\intercal B')^{-1} \right) x \right| \geq \epsilon_0/2 \right] \leq \delta_0 \tag{5.7}$$

The second stage deals with the problem that now, the gap $\Delta = B' - B$ is not necessarily a rank-1 matrix. However, what we show is that all stages in the proof of Claim 5.13 either rely on the singular values or can be written as the sum of a few rank-1 matrix multiplications.

The easier part is to claim that Eq. (5.6) holds. The analysis is a simple variation on the proof of Eq. (5.4). Fact 5.14 still holds for the singular values of A and A'. Observe that $\lambda_i^B > \sigma_i^B$ iff $\lambda_i > \sigma_i$. And so we have

$$\sqrt{\prod_i \frac{(\lambda_i^B)^2}{(\sigma_i^B)^2}} \leq \sqrt{\prod_{i \in Big} \frac{\lambda_i^2 + w^2}{\sigma_i^2 + w^2}} \leq \sqrt{\prod_{i \in Big} \frac{\lambda_i^2}{\sigma_i^2}}$$

and the remainder of the proof follows.

We now turn to proving Eq. (5.7). We start with an observation regarding $A'^\intercal A$ and $B'^\intercal B'$.

$$\begin{aligned} A'^\intercal A' &= (A+E)^\intercal (A+E) = A^\intercal A + A'^\intercal E + E^\intercal A \\ B^\intercal B &= V(\Sigma^2 + w^2 I)V^\intercal = V\Sigma^2 V^\intercal + w^2 I = A^\intercal A + w^2 I \\ B'^\intercal B' &= V'(\Lambda^2 + w^2 I)V'^\intercal = A'^\intercal A' + w^2 I \\ \Rightarrow\ B'^\intercal B' - B^\intercal B &= A'^\intercal E + E^\intercal A \end{aligned}$$

Now we can follow the same outline as in the proof of (5.5). Fix x, then:

$$x^\intercal (B^\intercal B)^{-1} x - x^\intercal (B'^\intercal B')^{-1} x \ = x^\intercal (B^\intercal B)^{-1} (B'^\intercal B')(B'^\intercal B')^{-1} x - x^\intercal (B'^\intercal B')^{-1} x =$$

$$
\begin{aligned}
&= x^\intercal (B^\intercal B)^{-1} \left[B^\intercal B + A'^\intercal E + E^\intercal A\right] (B'^\intercal B')^{-1} x \\
&\quad - x^\intercal (B'^\intercal B')^{-1} x \\
&= x^\intercal (B^\intercal B)^{-1} \left[A'^\intercal E + E^\intercal A\right] (B'^\intercal B')^{-1} x \\
&= x^\intercal (B^\intercal B)^{-1} (A^\intercal + E^\intercal) e_i \ \cdot \ v^\intercal (B'^\intercal B')^{-1} x \\
&\quad + x^\intercal (B^\intercal B)^{-1} v \ \cdot \ e_i^\intercal (A' - E) (B'^\intercal B')^{-1} x
\end{aligned}
$$

It is straight-forward to see that the i-th spectral values of $(B^\intercal B)^{-1} A$ is $\frac{\sigma_i}{\sigma_i^2 + w^2} \leq \frac{1}{\sqrt{\sigma_i^2 + w^2}} \leq 1/w$, and similarly for the spectral values of $(B'^\intercal B')^{-1} A'$. We now proceed as before and partition the above sum into multiplications of pairs of terms where $term_i \leq |vec_i \cdot y|$, and y is sampled from $\mathcal{N}(0, I_{n \times n})$ and $x = B^\intercal y$:

$$
\begin{aligned}
x^\intercal (B^\intercal B)^{-1} x - x^\intercal (B'^\intercal B')^{-1} x \ &= y^\intercal \left[B (B^\intercal B)^{-1} (A^\intercal + E^\intercal) e_i\right] \ \cdot \ \left[v^\intercal (B'^\intercal B')^{-1} B^\intercal\right] y \\
&\quad + y^\intercal \left[B (B^\intercal B)^{-1} v\ \right] \ \cdot \ \left[e_i^\intercal (A' - E) (B'^\intercal B')^{-1} B^\intercal\right] y
\end{aligned}
$$

Lastly, we need to bound all terms that contain the multiplication $(B'^\intercal B')^{-1} B^\intercal y$ in comparison to $(B'^\intercal B')^{-1} B'^\intercal y = B'^\dagger y$. For instance, take the $term = |vec^\intercal y|$ for $vec^\intercal = e_i^\intercal (A' - E) (B'^\intercal B')^{-1} B^\intercal$, and define it as $vec^\intercal = z^\intercal B^\intercal$. We can only bound $\|Bz\|$ using $\sigma_1^B / (\lambda_d^B)^2$, whereas we can bound $\|B'z\|$ with $1/\lambda_d^B < 1/w$. In contrast to before, we do not use the fact that $B^\intercal y = (B' - \Delta)^\intercal y$. Instead, we make the following derivations.

First, we observe that for every vector z we have that $\|B'z\| \geq \|A'z\|$ and $\|B'z\| \geq w\|z\|$. Using the fact that $B^\intercal B - \ B'^\intercal B' = -A'^\intercal E - E^\intercal A$, a simple derivation gives that $\|Bz\|^2 \leq (\|B'z\| + \|z\|)^2 \leq \left(1 + \frac{1}{w}\right)^2 \|B'z\|^2$, and vice-versa. So if y is s.t. $\frac{|z^\intercal B^\intercal y|}{\left(1 + \frac{1}{w}\right)\|B'z\|} > Threshold$ then $\frac{|z^\intercal B^\intercal y|}{\|Bz\|} > Threshold$. Observe that $z^\intercal B^\intercal y$ is distributed like $\mathcal{N}(0, \|Bz\|^2) = \|Bz\| \mathcal{N}(0, 1)$, and so we have that for every $\delta' > 0$

$$
\begin{aligned}
&\mathbf{Pr}\left[|z^\intercal B^\intercal y| \geq \sqrt{\log(1/\delta')} \left(1 + \frac{1}{w}\right) \|B'z\|\right] \\
&= \mathbf{Pr}\left[\left(\left(1 + \frac{1}{w}\right) \|B'z\|\right)^{-1} |z^\intercal B^\intercal y| \geq \sqrt{\log(1/\delta')}\right] \\
&\leq \mathbf{Pr}\left[\left(\|Bz\|\right)^{-1} |z^\intercal B^\intercal y| \geq \sqrt{\log(1/\delta')}\right] \leq \delta'
\end{aligned}
$$

□

Corollary. Using the definitions of r and w as in Algorithm 4 – the proof of Theorem 5.11 actually shows that in the case that A is a matrix with all singular values $\geq w$,

then the following simple algorithm preserves (ϵ, δ)-differential privacy: pick a random $r \times n$ matrix M whose entries are iid normal Gaussians, and output $O = MA$. Furthermore, observe that if σ_d, the least singular value of A, is bigger than, say, $10w$, then one can release $\sigma_d + Lap(1/\epsilon)$ then release $O = MA$. In such a case, users know that for any unit vector x w.p. $\geq 1 - \nu$ it holds that $\frac{1}{r}\|Ox\|^2 \leq (1 \pm \eta)\|Ax\|^2$.

Comment. Comparing Algorithms 2 and 4, we have that in $L_G = E_G^{\intercal} E_G$ we "translate" the spectral values by w, and in $A^{\intercal}A$ we "translated" the spectral values by w^2. This is an artifact of the ability to directly compare the spectal values of L_G and $L_{G'}$ in the first analysis, whereas in the second analysis we compare the spectral values of A and A' (vs. $A^{\intercal}A$ and $A'^{\intercal}A'$). This is why the noise bounds in the general case are $\tilde{O}(1/\epsilon\eta)$ times worse than for graphs.

5.4.2 Comparison with Other Algorithms

To the best of our knowledge, no previous work has studied the problem of preserving the variance of A in the same formulation as us. We deal with a scenario where users pose the directions on which they wish to find the variance of A. Other algorithms, that publish the PCA or a low-rank approximation of A without compromising privacy (see Section 6.1.1), provide users with specific directions and variances. These works are not comparable with our algorithm, as they give a different utility guarantee. For example, low-rank approximations aim at nullifying the projection of A in certain directions.

Here, we compare our method to the Laplace mechanism, the Multiplicative Weights mechanism and Randomized Response. The bottom line is clear: our method allows one to answer directional variance queries with additive noise which is independent of the given input. Other methods require we add random noise that depends on the size of the matrix, assuming we answer polynomially many queries.

Our notation is as follows. n denotes the number of rows in the matrix (number of individuals in the data), d denotes the number of columns in the matrix, and we assume each entry is at most 1. As before, ϵ denotes the privacy parameter and k denotes the number of queries. Observe that we (again) compare k *predetermined* queries for non-interactive mechanisms with k *adaptively chosen* queries for interactive ones. The remaining parameters are omitted from this comparison. Results are summarized in Table 5.2.

5.4.2.1 Naïvely Adding Laplace Noise

Again, the simplest alternative is to answer each directional-variance query with $\Phi_x(A) + Lap(0, \tilde{\epsilon}^{-1})$ for a suitable value of $\tilde{\epsilon}$. The composition theorem of [65] assures us that to answer k queries while overall preserving (ϵ, δ)-differential privacy, we must set $\tilde{\epsilon} < \epsilon/\sqrt{k}$, and so the additive error per query is $O(\sqrt{k}/\epsilon)$.

5.4.2.2 Randomized Response

We now consider a Randomized Response mechanism, similar to the Randomized Response mechanism of [77]. We wish to output a noisy version of $A^\intercal A$, by adding some iid random noise to each entry of $A^\intercal A$. Since we call two matrices neighbors if they differ only on a single row, denote v as the difference vector on that row. It is simple to see that by adding v to some row in A, each entry in $A^\intercal A$ can change by at most $\|v\|_1$. Recall that we require $\|v\|_2 = 1$ and so $\|v\|_1 \leq \sqrt{d}$. Therefore, we have that in order to preserve (ϵ, δ)-differential privacy, it is enough to add a random Gaussian noise of $\mathcal{N}(0, \frac{d\log(d)}{\epsilon^2})$ to each of the d^2 entries of $A^\intercal A$.

Next we give the utility guarantee of the Randomized Response scheme. Fix any unit length vector x. We think of the matrix we output as $A^\intercal A + N$, where N is a matrix of iid samples from $\mathcal{N}(0, \frac{d\log(d)}{\epsilon^2})$. Therefore, in direction x, we add to the true answer a random noise distributed like $x^\intercal N x \sim \mathcal{N}(0, \left(\sum_{i,j} x_i^2 x_j^2\right) \frac{d\log(d)}{\epsilon^2}) = \mathcal{N}(0, \frac{d\log(d)}{\epsilon^2})$. So w.h.p the noise we add is within factor of $\tilde{O}(\sqrt{d}/\epsilon)$ for each query, and for k queries it is within factor of $\tilde{O}(\sqrt{d\log(k)}/\epsilon)$.

5.4.2.3 The Multiplicative Weights Mechanism

It is not straight-forward to adapt the Multiplicative Weights mechanism to answer directional variance queries. We represent $A^\intercal A$ as a histogram over its d^2 entries (so the size of the "universe" is $N = d^2$), but it is not simple to estimate what is the equivalent of number of individuals in this representation. We chose to take the pessimistic bound of nd^2, since this is the L_1 bound on the sum of entries in $A^\intercal A$, but we comment this is a highly pessimistic bound. It is fairly likely that the number of individuals in this representation can be set to only $O(d^2)$.

Plugging these parameters into the utility bounds of the Multiplicative Weights mechanism, we get a utility bound of $\tilde{O}(d\sqrt{n}\log(k)/\epsilon)$. Plugging them into the improved bounds of the IDC mechanism, we get $\tilde{O}(d\sqrt{n\log(k)/\epsilon})$. Observe that even if replace the

Method	Additive Error	Multiplicative Error?	Interactive?	Tractable?
Laplace Noise	$O(\sqrt{k}/\epsilon)$	✗	✓	✓
Randomized Response	$\tilde{O}(\sqrt{d\log(k)}/\epsilon)$	✗	✗	✓
MW IDC	$\tilde{O}(d\sqrt{n}\log(k)/\epsilon)$ $\tilde{O}(d\sqrt{n\log(k)}/\epsilon)$	✗	✓	✓
JL	$O(\log(k)/\epsilon^2)$	✓	✗	✓

Table 5.2: Comparison between mechanisms for answering directional variance queries.

pessimistic bound of nd^2 with just d^2, these bounds depend on d.

5.4.2.4 Our Algorithm

Our algorithm's utility is computed simply by plugging in $\nu = O(1/k)$ to Theorem 5.12, which gives a utility bound of $O(\log(k)/\epsilon^2)$.

5.5 Discussion and Open Problems

The fact that the JL transform preserves differential privacy is likely to have more theoretical and practical applications than the ones detailed in this chapter. Below we detail a few of the open questions we find most compelling.

Error depedency on r. Our algorithm projects the edge-matrix of a given graph on r random directions, then publishes these projections. The value of r determines the probability we give a good approximation to a given cut-query, and provided that we wish to give a good approximation to all cut-queries, our analysis requires us to set $r = \Omega(n)$. But is it just an artifact of the analysis? Could it be that a better analysis gives a better bound on r? It turns out that the answer is "no". In fact, the direction on which we project the data now have high correlation with the published Laplacian. We demonstrate this with an example.

Assume our graph is composed of a single perfect matching between $2n$ nodes, where node i is matched with node $n + i$. Focus on a single random projection – it is chosen by picking $\binom{2n}{2}$ iid random values $x_{i,j} \sim \mathcal{N}(0, 1)$, and for the ease of exposition imagine that

the values of the edges in the matching are picked first, then the values of all other pairs of vertices. Now, if we pick the value $x_{i,n+i}$ for the $\langle i, n+i\rangle$ edge, then node i is assigned $x_{i,n+i}$ while node $n+i$ is assigned $-x_{i,n+i}$. So regardless of the sign of $x_{i,n+i}$, *exactly one* of the two nodes $\{i, n+i\}$ is assigned the positive value $|x_{i,n+i}|$ and exactly one is assigned the negative value $-|x_{i,n+i}|$. Define S as the set of n nodes that are assigned the positive values and $\bar{S}$ as the set of n nodes that are assigned the negative values. The sum of weight crossing the $(S, \bar{S})$-cut is distributed like $(X + \frac{w}{n}Y)^2$ where $X = \sum_i |x_{i,n+i}|$ and $Y = \sum_i \sum_{j \neq n+i} x_{i,j}$. Indeed, Y is the sum of $n(n-1)$ random normal iid Gaussians, but X is the sum of n *absolute values* of Gaussians. So w.h.p. both X and Y are proportional to n. Therefore, in the direction of this particular random projection we estimate the $(S, \bar{S})$-cut as $\Omega((n \pm w)^2) = \Omega(n^2)$ rather than $O(n)$. (If X was distributed like the sum of n iid normal Gaussians, then the estimation would be proportional to $(\sqrt{n})^2 = n$.)

Assuming that the remaining $r-1$ projections estimate the cut as $O(n)$, then by averaging over all r random projections our estimation of the $(S, \bar{S})$-cut is $\omega(n)$, as long as $r = o(n)$.

Error amplification or error detection. Having established that we do err on some cuts, we pose the question of error amplification. Can we introduce some error-correction scheme to the problem without increasing r significantly? Error amplification without increasing r will allow us to keep the additive error fairly small. One can view $\tilde{L}$ as a coding of answers to all 2^n cut-queries which is guaranteed to have at least $1-\nu$ fraction of the code correct, in the sense that we get a (η, τ)-approximation to the true cut-query answer. As such, it is tempting to try some self-correcting scheme – like adding a random vector x to the vector $\mathbf{1}_S$, then finding the estimation to $x^\intercal L_G x$ and $(\mathbf{1}_s + x)^\intercal L_G x$ and inferring $\mathbf{1}_S^\intercal L_G \mathbf{1}_S$. We were unable to prove such scheme works due to the dot-product problem (see next paragraph) and to query dependencies.

A related question is of error detection: can we tell whether $\tilde{L}$ gives a good estimation to a cut query or not? One potential avenue is to utilize the trivial guess for $\Phi_G(S)$ – the expected value $\frac{m}{\binom{n}{2}} s(n-s)$ (we can release m via the Laplace mechanism). We believe this question is related to the problem of estimating the variance of $\{\Phi_G(S) \ : \ |S| = s\}$.

Edges between S and T. Our work assures utility only for cut-queries. It gives no utility guarantees for queries regarding $E(S,T)$, the set of edges connecting two disjoint vertex-subsets S and T. The reason is that it is possible to devise a graph where both $E(S, \bar{S})$ and $E(T, \bar{T})$ are large whereas $E(S,T)$ is fairly small. When $E(S, \bar{S})$ and $E(T, \bar{T})$ are big, the *multiplicative error* η given to both quantities might add too much noise to an estimation

of $E(S,T)$.

The problem relates to the dot-product estimation of the JL transform. It is a classical result that if M is a distance-preserving matrix and u and v are two vectors s.t. $\|M(u+v)\|^2 \approx \|u+v\|^2$ and $\|M(u-v)\|^2 \approx \|u-v\|^2$ then it is possible to bound the difference $|Mu \cdot Mv - u \cdot v|$. But this bound is a function of $\|u\|$ and $\|v\|$, which in our case translates to a bound that depends on $\|E_G \mathbf{1}_S\|$ and $\|E_G \mathbf{1}_T\|$, both vectors of potentially large norms.

Other Versions of JL. The analysis in this works deals with the most basic JL transform, using normal Gaussians. We conjecture that qualitatively the same results should apply for most of the other versions of the JL transform that utilize a dense transform (e.g., using unit-length projections or uniformly chosen entries in $[-1,1]$). Notice however that some versions of the JL clearly do not preserve privacy. For example, the version of Achlioptas [3] where each entry is chosen uniformly to be 1 or -1, clearly does not preserve privacy – it is easy to differentiate between the complete graph K_n and its neighbor based on observing whether a specific coordinate is even or odd. It is an interesting open problem to see whether *sparse* JL transforms (see [55]) preserve differential privacy or not.

Chapter 6

Differentially Private Data Analysis of Social Networks via Restricted Sensitivity

6.1 Introduction

The social networks we inhabit have grown significantly in recent decades with digital technology enabling the rise of networks like Facebook that now connect over 900 million people and house vast repositories of personal information. At the same time, the study of various characteristics of social networks has emerged as an active research area [66]. Yet the fact that the data in a social network might be used to infer sensitive details about an individual, like sexual orientation [92], is a growing concern among social networks' participants. Even in an 'anonymized' unlabeled graph it is possible to identify people based on graph structures [23]. Here we study the feasibility of and design efficient algorithms to release statistics about social networks (modeled as graphs with vertices labeled with attributes) while satisfying the semantic definition of differential privacy [63, 62].

A differentially private mechanism guarantees that any two neighboring data sets (i.e., data sets that differ only on the information about a single individual) induce similar distributions over the statistics released. For social networks, which are graphs with node coloring, we consider two notions of neighboring or adjacent networks: (1) *edge adjacency* stipulating that adjacent graphs differ in just one edge or in the attributes of just one vertex; and (2) *vertex adjacency* stipulating that adjacent networks differ on just one vertex—its attributes or *any number* of edges incident to it. We comment that the edge

adjacency model considered here is slightly different than the edge-adjacency model considered in Chapter 5 as we also allow a node to change its attributes.

For any given statistic or query, its global sensitivity measures the maximum difference in the answer to that query over all pairs of neighboring data sets [63]; global sensitivity provides an upper bound on the amount of noise that has to be added to the actual statistic in order to preserve differential privacy. Since the global sensitivity of certain types of queries can be quite high, the notion of smooth sensitivity was introduced to reduce the amount of noise that needs to be added while still preserving differential privacy [119].

However, a key challenge in the differentially private analysis of social networks is that for many natural queries, both global and smooth sensitivity can be very large. In the vertex adjacency model, consider the query "How many people in G_1 are a doctor or are friends with a doctor?" Even if the answer is 0 (e.g., there are no doctors in the social network) there is a neighboring social network G_2 in which the answer is n (e.g., pick an arbitrary person from G_1, relabel him as a doctor, and connect him to everyone). Even in the edge adjacency model, the sensitivity of queries may be high. Consider the query "How many people in G_1 are friends with two doctors who are also friends with each other?" In G_1 the answer may be 0 even if there are two doctors that everyone else is friends with (e.g, the doctors are not friends with each other), but the answer jumps to $n-2$ in a neighboring graph G_2 (e.g, if we simply connect the doctors to each other).

In fact, such queries have high global sensitivity in the edge-adjacency model we consider here, because this model allows for a node to change its attributes arbitrarily. For example, the global sensitivity of the query "how many people are friends with a doctor" is $\Omega(n)$ – as illustrated by the two adjacent star graphs (one node with degree $n-1$ and all other nodes with degree 1) where in one no node is a doctor and in the other the central node has a M.D. Therein lies the difference between the problem considered in Chapter 5 and this chapter. In Chapter 5 we consider only the graph structure, or alternatively a social networks where the labeling of each node is public knowledge – and so cut-queries have global sensitivity of 1. Here we consider possible changes both to the graph structure and to the labeling of the nodes, so cut-queries may have a sensitivity of $\Omega(n)$. For example, consider the query "do Chinese people have many non-Chinese friends on average?" If the labeling of each node is public we may use any of the techniques detailed in Chapter 5 to answer such query. However, if the labeling of each node is private (we want each person to be able to pretend she is Chinese), the only existing differentially privacy technique that might answer this query with $o(n)$ error is the framework of smooth-sensitivity [119].

Yet, while the examples obtaining global sensitivity of n respect the mathematical definitions of neighboring graphs and networks, we note that in a real social network no single individual is likely to be directly connected with everyone else. Suppose that

in fact a querier has some such belief $\mathcal{H}$ about the given network ($\mathcal{H}$ is a subset of all possible networks) such that its query f has low sensitivity restricted only to inputs and deviations within $\mathcal{H}$. For example, the querier may believe the following hypothesis ($\mathcal{H}_k$): the maximum degree of any node in the network is at most $k = 5000 \ll n \approx 9 \times 10^8$ (e.g, after reading a study on the anatomy of Facebook [134]). Can one in that case provide accurate answers in the event that indeed $G \in \mathcal{H}$ and yet preserve privacy no matter what (even if $\mathcal{H}$ is not satisfied)?

Here we provide a positive answer to this question. We do so by introducing the notion of *restricted sensitivity*, which represents the sensitivity of the query f over only the given subset $\mathcal{H}$, and providing procedures that map a query f to an alternative query $f_{\mathcal{H}}$ s.t. f and $f_{\mathcal{H}}$ identify over the inputs in $\mathcal{H}$, yet the global sensitivity of $f_{\mathcal{H}}$ is comparable to just the restricted sensitivity of f. Therefore, the mechanism that answers according to $f_{\mathcal{H}}$ and adds Laplace random noise preserves privacy for *all* inputs, while giving good estimations of f for inputs in $\mathcal{H}$.

While our general scheme for devising such $f_{\mathcal{H}}$ is inefficient and requires that we construct a separate $f_{\mathcal{H}}$ for each query f, we also design a complementary *projection-based* approach. A projection of $\mathcal{H}$ is a function mapping all possible inputs (e.g., all possible n-node social networks) to inputs in $\mathcal{H}$ with the property that any input in $\mathcal{H}$ is mapped to itself. Therefore, a projection μ allows us to define $f_{\mathcal{H}}$ for any f, simply by composing $f_{\mathcal{H}} = f \circ \mu$. Moreover, if this projection μ satisfies certain smoothness properties, which we define in Section 6.4, then this function $f_{\mathcal{H}}$ will have its global sensitivity—or at least its smooth sensitivity over inputs in $\mathcal{H}$—comparable to only the restricted sensitivity of f. In particular, for the case $\mathcal{H} = \mathcal{H}_k$ (the assumption that the network has degree at most $k \ll n$), we show we can *efficiently* construct projections μ satisfying these conditions, therefore allowing us to efficiently take advantage of low restricted sensitivity. These results are given in Section 6.4 and summarized in Table 6.1.

The next natural question is: how much advantage does restricted sensitivity provide, compared to global or smooth sensitivity, for natural query classes and natural sets $\mathcal{H}$? In Section 6.5 we consider two natural classes of queries: local profile queries and subgraph counting queries. A local profile query asks how many nodes v in a graph satisfy a property which depends only on the immediate neighborhood of v (e.g, queries relating to clustering coefficients and bridges [66], or queries such as "how many people know two spies who don't know each other?"). A subgraph counting query asks how many copies of a particular subgraph P are contained in the network (e.g., number of triangles involving at least one spy). For the case $\mathcal{H} = \mathcal{H}_k$ for $k \ll n$ we show that the restricted sensitivity of these classes of queries can indeed be much lower than the smooth sensitivity. These results, presented in Section 6.5, are summarized in Table 6.2.

	Adjacency	Hypothesis	Query	Sensitivity	Efficient
Theorem 6.5	Any	Any	Any	$GS_{f_{\mathcal{H}}} = RS_f(\mathcal{H})$	No
Theorem 6.10	Edge	$\mathcal{H}_k$	Any	$GS_{f_{\mathcal{H}}} = 3RS_f(\mathcal{H})$	Yes
Theorem 6.16	Vertex	$\mathcal{H}_k$	Any	$S_{f_{\mathcal{H}}} = O(1) \times RS_f(\mathcal{H}_{2k})$	Yes

Table 6.1: Summary of Results. GS = global sensitivity, RS = restricted sensitivity, and S = smooth bound of local sensitivity.

	Subgraph Counting Query P		Local Profile Query	
Adjacency	Smooth	Restricted	Smooth	Restricted
Edge	$\lvert P\rvert\, k^{\lvert P\rvert-1}$	$\lvert P\rvert\, k^{\lvert P\rvert-1}$	$k+1$	$k+1$
Vertex	$O\left(n^{\lvert P\rvert-1}\right)$	$\lvert P\rvert\, k^{\lvert P\rvert-1}$	$n-1$	$2k+1$

Table 6.2: Worst Case Smooth Sensitivity over $\mathcal{H}_k$ vs. Restricted Sensitivity $RS_f(\mathcal{H}_k)$.

6.1.1 Related Work

Easley and Kleinberg provide an excellent summary of the rich literature on social networks [66]. Previous literature on differentially-private analysis of social networks has primarily focused on the edge adjacency model in unlabeled graphs where sensitivity is manageable. Triangle counting queries can be answered in the edge adjacency model by efficiently computing the smooth sensitivity [119], and this result can be extended to answer other counting queries [100]. Hay et al [84] show how to privately approximate the degree distribution in the edge adjacency model. The Johnson-Lindenstrauss transform can be used to answer all cut queries in the edge adjacency model [41]. Kasiviswanathan, Nissim, Raskhodnikova and Smith [102] have independently been exploring and developed an analysis for node level privacy using an approach similar to ours.

The approach taken in the work of Rastogi et al. [125] on answering subgraph counting queries is the most similar to ours. They consider a bayesian adversary whose prior (background knowledge) is drawn from a distribution. Leveraging an assumption about the adversary's prior they compute a high probability upper bound on the local sensitivity of the data and then answer by adding noise proportional to that bound. Loosely, they assume that the presence of an edge does not presence of other edges more likely. In the specific context of a social network this assumption is widely believed to be false (e.g., two people are more likely to become friends if they already have common friends [66]). The privacy guarantees of [125] only hold if these assumptions about the adversaries prior are true. By contrast, we always guarantee privacy even if the assumptions are incorrect.

A relevant approach that deals with preserving differential privacy while providing better utility guarantees for nice instances is detailed in the work of Nissim et al [119]

who define the notion of smooth sensitivity. In their framework, the amount of random noise that the mechanism adds to a query's true ansewr is dependent on the extent for which the input database is "nice" – having small *local sensitivity*. As we discuss later, in social networks many natural queries (e.g., local profile queries) even have high local and smooth sensitivity.

6.2 Preliminaries – Graphs and Social Networks

Our work is motivated by the challenges posed by differentially private analysis of social networks. As always, a graph is a pair of a set of vertices and a set of edges $G = \langle V, E \rangle$. We often just denote a graph as G, referring to its vertex-set or edge-set as $V(G)$ or $E(G)$ resp. A key aspect of our work is modeling a social network as a *labeled* graph.

Definition 6.1. *A* social network (G, ℓ) *is a graph with labeling function* $\ell : V(G) \to \mathbb{R}^m$. *The set of all social networks is denoted* $\mathcal{G}$.

The labeling function ℓ allows us to encode information about the nodes (e.g., age, gender, occupation). For convenience, we assume all social networks are over the same set of vertices, which is denotes as V and has size n, and so we assume $|V| = n$ is public knowledge.[1] Therefore, the graph structures of two social networks are equal if their edge-sets are identical. Similarly, we also fix the dimension m of our labeling.

Defining differential privacy over the labeled graphs $\mathcal{G}$ requires care. What does it mean for two labeled graphs $G_1, G_2 \in \mathcal{G}$ to be neighbors? There are two natural notions: edge-adjacency and vertex adjacency.

Definition 6.2 (Edge-adjacency)**.** *We say that two social networks* (G_1, ℓ_1) *and* (G_2, ℓ_2) *are neighbors if either (i)* $E(G_1) = E(G_2)$ *and there exists a vertex* u *such that* $\ell_1(u) \neq \ell_2(u)$ *whereas for every other* $v \neq u$ *we have* $\ell_1(v) = \ell_2(v)$ *or (ii)* $\forall v, \ell_1(v) = \ell_2(v)$ *and the symmetric difference* $E(G_1) \triangle E(G_2)$ *contains a single edge.*

In the context of a social network, differential-privacy w.r.t edge-adjacency can, for instance, guarantee that an adversary will not be able to distinguish whether a particular individual has friended some specific pop-singer on Facebook. However, such guarantees do not allow a person to pretend to listen only to high-end indie rock bands, should that person have friended numerous pop-singers on Facebook. This motivates the stronger vertex-adjacency neighborhood model.

[1] Adding or removing vertices could be done by adding one more dimension to the labeling, indicating whether a node is active or inactive.

Definition 6.3 (Vertex-adjacency)**.** *We say that two social networks* (G_1, ℓ_1) *and* (G_2, ℓ_2) *are neighbors if there exists a vertex* v_i *such that* $G_1 - v_i = G_2 - v_i$ *and* $\ell_1(v_j) = \ell_2(v_j)$ *for every* $v_j \neq v_i$*.*

where for a graph G and a vertex v we denote $G - v$ as the result of removing every edge in $E(G)$ that touches v.

It is evident that any two social networks that are edge-adjacent are also vertex-adjacent. Preserving differential privacy while guaranteeing good utility bounds w.r.t vertex-adjacency is a much harder task than w.r.t edge-adjacency.

Distance Given two social networks (G_1, ℓ_1) and (G_2, ℓ_2), recall that their distance is the minimal k s.t. one can form a path of length k, starting with (G_1, ℓ_1) and ending at (G_2, ℓ_2), with the property that every two consecutive social-networks on this path are adjacent. Given the above two definitions of adjacency, we would like to give an alternative characterization of this distance.

First of all, the set $U = \{v : \ \ell_1(v) \neq \ell_2(v)\}$ dictates $|U|$ steps that we must take in order to transition from (G_1, ℓ_1) to (G_2, ℓ_2). It is left to determine how many adjacent social-networks we need to transition through until we have $E(G_1) = E(G_2)$. To that end, we construct the difference-graph whose edges are the symmetric difference of $E(G_1)$ and $E(G_2)$. Clearly, to transition from (G_1, ℓ_1) to (G_2, ℓ_2), we need to alter every edge in the difference graph. In the edge-adjacency model, a pair of adjacent social networks covers precisely a single edge, and so it is clear that the distance $d\big((G_1, \ell_1), (G_2, \ell_2)\big) = |U| + |E(G_1) \triangle E(G_2)|$. In the vertex-adjacency model, a single vertex can cover all the edges that touch it, and so the distance between the graphs $G_1 - U$ and $G_2 - U$ is precisely the *vertex cover* of the difference graph. Denoting this vertex cover as $VC(G_1 - U \triangle G_2 - U)$ we have that $d\big((G_1, \ell_1), (G_2, \ell_2)\big) = |U| + |VC(G_1 - U \triangle G_2 - U)|$. It is evident that computing the distance of between any two social-networks in the vertex-adjacency model is a NP-hard problem.

To avoid cumbersome notation, throughout this entire chapter we omit the differentiation between graphs and social networks, and denote networks as graphs $G \in \mathcal{G}$.

6.3 Restricted Sensitivity

We now introduce the notion of restricted sensitivity, using a hypothesis about the dataset D to restrict the sensitivity of a query. A hypothesis $\mathcal{H}$ is a subset of the set $\mathcal{D}$ of all

possible datasets (so in the context of social networks, $\mathcal{H}$ is a set of labeled graphs). We say that $\mathcal{H}$ is true if the true dataset $D \in \mathcal{H}$. Because the hypothesis $\mathcal{H}$ may not be a convex set we must consider all pairs of datasets in $\mathcal{H}$ instead of all pairs of adjacent datasets as in the definition of global sensitivity.

Definition 6.4. *For a given notion of adjacency among datasets, the* restricted sensitivity *of f over a hypothesis $\mathcal{H} \subset \mathcal{D}$ is*

$$RS_f(\mathcal{H}) = \max_{D_1, D_2 \in \mathcal{H}} \left(\frac{|f(D_1) - f(D_2)|}{d(D_1, D_2)} \right).$$

To be clear, $d(D_1, D_2)$ denotes the length of the shortest-path in $\mathcal{D}$ between D_1 and D_2 (not restricting the path to only use $D \in \mathcal{H}$) using the given notion of adjacency (e.g., edge-adjacency or vertex-adjacency). That is, we restrict the set of databases for which we compute the sensitivity, but we do not re-define the distances.

Observe that $RS_f(\mathcal{H})$ may be smaller than $LS_f(D)$ for some $D \in \mathcal{H}$ if D has a neighbor $D' \notin \mathcal{H}$. In fact we often have $LS_f(D) \geq |f(D) - f(D')| \gg RS_f(\mathcal{H})$. In such cases $RS_f(\mathcal{H})$ will be significantly lower than any β-smooth upper bound on $LS_f(D)$. For example, consider the query f defined as "how many people are friends with a doctor" under the vertex adjacency model. For any social network with at least one doctor there exists a neighboring network where f takes the value n – the one where the doctor node connects to all other nodes. Therefore, even if the given network is such that everyone has no more than k friends, it still holds that the local sensitivity of f is high. In contrast, by hypothesizing that the given network is such that no one has more than k friends, the restricted sensitivity of f is $O(k)$. Further discussion is given in Section 6.5.

6.4 Using Restricted Sensitivity to Reduce Noise

To achieve differential privacy while adding noise proportional to $RS_f(\mathcal{H})$ we must be willing to sacrifice accuracy guarantees for datasets $D \notin \mathcal{H}$. Our goal is to create a new query $f_{\mathcal{H}}$ such that $f_{\mathcal{H}}(D) = f(D)$ for every $D \in \mathcal{H}$ ($f_{\mathcal{H}}$ is accurate when the hypothesis is correct) and $f_{\mathcal{H}}$ either has low global sensitivity or low β-smooth sensitivity over datasets $D \in \mathcal{H}$. (Again we emphasize – the global sensitivity of $f_{\mathcal{H}}$ is defined w.r.t to *any* pair of neighboring datasets and not only neighboring datasets in $\mathcal{H}$. Similarly, the smooth sensitivity of $f_{\mathcal{H}}(D)$ is w.r.t any of the neighbors of D regardless of whether they belong to $\mathcal{H}$ or not.)

In this section, we first give a non-efficient generic construction of such $f_{\mathcal{H}}$, showing that it is always possible to devise $f_{\mathcal{H}}$ whose global sensitivity equals *exactly* the restricted

sensitivity of f over $\mathcal{H}$. We then show how for the case of social networks and for the hypothesis $\mathcal{H}_k$ that the network has bounded degree, we can construct functions $f_{\mathcal{H}_k}$ having approximately this property, efficiently.

6.4.1 A General Construction

We now show how given $\mathcal{H}$ to generically (but not efficiently) construct $f_{\mathcal{H}}$ whose global sensitivity *exactly* equals the restricted sensitivity of f over $\mathcal{H}$. (The theorem we give can be viewed as a special case of constructing a Lipschitz extension of $f|_{\mathcal{H}}$ over $\mathcal{D}$ – i.e., mimicking either the Whitney or McShane extensions, see [94] for details.)

Theorem 6.5. *Given any query f and any hypothesis $\mathcal{H} \subset \mathcal{D}$ we can construct a query $f_{\mathcal{H}}$ such that*

1. *$\forall D \in \mathcal{H}$ it holds that $f_{\mathcal{H}}(D) = f(D)$, and*
2. *$GS_{f_{\mathcal{H}}} = RS_f(\mathcal{H})$*

Proof. For each $D \in \mathcal{H}$ set $f_{\mathcal{H}}(D) = f(D)$. Now fix an arbitrary ordering of the set $\{D : D \notin \mathcal{H}\}$, and denote its elements as $D_1, D_2, \ldots, D_m$, where m is the size of the set. For every $D \notin \mathcal{H}$ we define the value of $f_{\mathcal{H}}(D)$ inductively. Denote $\mathcal{T}_i = \mathcal{H} \bigcup \{D_1, ..., D_i\}$. Initially, we are given the values of every $D \in \mathcal{T}_0$. Given $i > 0$, we denote $\Delta_i = RS_{f_{\mathcal{H}}}(\mathcal{T}_i)$. We now prove one can pick the value $f_{\mathcal{H}}(D_i)$ in a way that preserves the invariant that $\Delta_{i+1} = \Delta_i$. By applying the induction m times we conclude that

$$RS_f(\mathcal{H}) = \Delta_0 = \Delta_m = RS_{f_{\mathcal{H}}}(\mathcal{D}) = GS_{f_{\mathcal{H}}}.$$

Fix $i > 0$. Observe that

$$\Delta_{i+1} = \max\left(\Delta_i, \left(\max_{D \in \mathcal{T}_i} \frac{|f_{\mathcal{H}}(D) - f_{\mathcal{H}}(D_{i+1})|}{d(D, D_{i+1})}\right)\right)$$

so to preserve the invariant it suffices to find any value of $f_{\mathcal{H}}(D_{i+1})$ that satisfies that for every $D \in \mathcal{T}_i$ we have $|f_{\mathcal{H}}(D) - f_{\mathcal{H}}(D_{i+1})| \leq \Delta_i \cdot d(D, D_{i+1})$. Suppose for contradiction that no value exists. Then there must be two intervals

$$[f_{\mathcal{H}}(D_1^*) - \Delta_i \cdot d(D_1^*, D_{i+1}),\ f_{\mathcal{H}}(D_1^*) + \Delta_i \cdot d(D_1^*, D_{i+1})]$$
$$[f_{\mathcal{H}}(D_2^*) - \Delta_i \cdot d(D_2^*, D_{i+1}),\ f_{\mathcal{H}}(D_2^*) + \Delta_i \cdot d(D_2^*, D_{i+1})]$$

which don't intersect. This would imply that

$$\frac{|f_{\mathcal{H}}(D_1^*) - f_{\mathcal{H}}(D_2^*)|}{d(D_1^*, D_2^*)} \geq \frac{|f_{\mathcal{H}}(D_1^*) - f_{\mathcal{H}}(D_2^*)|}{d(D_{i+1}, D_1^*) + d(D_{i+1}, D_2^*)} > \Delta_i$$

which contradicts the fact that Δ_i is the restricted sensitivity of $\mathcal{T}_i$. □

6.4.2 Efficient Procedures for $\mathcal{H}_k$ via Projection Schemes

Unfortunately, the construction of Theorem 6.5 is highly inefficient. Furthermore, this construction deals with one query at a time. We would like to a-priori have a way to efficiently devise $f_{\mathcal{H}}$ for any f. In this section, the way we devise $f_{\mathcal{H}}$ is by constructing a *projection* – a function $\mu : \mathcal{D} \to \mathcal{H}$ with the property that $\mu(D) = D$ for every $D \in \mathcal{H}$. Such μ allows us to canonically convert *any* f into $f_{\mathcal{H}}$ using the naïve definition $f_{\mathcal{H}} = f \circ \mu$. Below we discuss various properties of projections that allow us to derive "good" $f_{\mathcal{H}}$-s. Following each property, we exhibit the existence of such projections μ for the specific case of social networks and $\mathcal{H} = \mathcal{H}_k$, the class of graphs of degree at most k.

Definition 6.6. *The class $\mathcal{H}_k$ is defined as the set $\{G \in \mathcal{G} : \forall v,\ \deg(v) \leq k\}$.*

In many labeled graphs, it is reasonable to believe that $\mathcal{H}_k$ holds for $k \ll n$ because the degree distributions follow a power law. For example, the number of telephone numbers receiving t calls in a day is proportional to $1/t^2$, and the number of web pages with t incoming links is proportional to $1/t^2$ [118, 46, 66]. For these networks it would suffice to set $k = O(\sqrt{n})$. The number of papers that receive t citations is proportional to $1/t^3$ so we could set $k = O(\sqrt[3]{n})$ [66]. While the degrees on Facebook don't seem to follow a power law, the upper bound $k = 5,000$ seems reasonable [134]. By contrast, Facebook had approximately $n = 901,000,000$ users in June, 2012 [1].

6.4.2.1 Smooth Projection

The first property we discuss is perhaps the simplest and most coveted property such projection can have – smoothness. Smoothness dictates that there exists a global bound on the distance between any two mappings of two neighboring databases.

Definition 6.7. *A projection $\mu : \mathcal{D} \to \mathcal{H}$ is called* c-smooth *if for any two neighboring databases $D \sim D'$ we have that $d\big(\mu(D), \mu(D')\big) \leq c$.*

Lemma 6.8. *Let $\mu : \mathcal{D} \to \mathcal{H}$ be a c-smooth projection (i.e., for every $D \in \mathcal{H}$ we have $\mu(D) = D$). Then for every query f, the function $f_{\mathcal{H}} = f \circ \mu$ satisfies that $GS_{f_{\mathcal{H}}} \leq c \cdot RS_f(\mathcal{H})$.*

Proof.

$$\begin{aligned} GS_{f_{\mathcal{H}}} &= \max_{D_1 \sim D_2} |f_{\mathcal{H}}(D_1) - f_{\mathcal{H}}(D_2)| \\ &= \max_{D_1 \sim D_2} |f(\mu(D_1)) - f(\mu(D_2))| \cdot 1 \\ &\leq \max_{D_1 \sim D_2} |f(\mu(D_1)) - f(\mu(D_2))| \frac{c}{d(\mu(D_1), \mu(D_2))} \\ &\leq c \cdot \max_{D_1, D_2 \in \mathcal{H}} \frac{|f(D_1) - f(D_2)|}{d(D_1, D_2)} \\ &= c \cdot RS_f(\mathcal{H}) \end{aligned}$$

□

As we now show, for $\mathcal{H} = \mathcal{H}_k$ and for distances defined via the edge-adjacency model, we can devise an efficient smooth projection.

Claim 6.9. *In the edge-adjacency model, there exists an efficiently computable 3-smooth projection to $\mathcal{H}_k$.*

Proof. We construct our smooth-projection μ by first fixing a canonical ordering over all possible edges. Let $e_1^v, ..., e_t^v$ denote the edges incident to v in canonical order. For each edge $e = \{u, v\}$ we delete e if and only if (i) $e = e_j^v$ for $j > k$ or (ii) $e = e_j^u$ for $j > k$ (Intuitively for each v with $\deg(v) \geq k$ we keep this first k edges incident to v and flag the other edges for deletion). If $G \in \mathcal{H}_k$ then no edges are deleted, so $\mu(G) = G$. Suppose that G_1, G_2 are neighbors differing on one edge $e = \{x, y\}$ (wlog, say that e is in G_1). Observe that for every $v \neq x, y$, the same set of edges incident to v will be deleted from both G_1 and G_2. In fact, if $\mu(G_1)$ does not contain e then $\mu(G_1) = \mu(G_2)$. Otherwise, if e is not deleted we may assume then there may be at most one edge e_x (incident to x) and at most one edge e_y (incident to y) that were deleted from $\mu(G_1)$ but not from $\mu(G_2)$. Hence, $d(\mu(G_1), \mu(G_2)) \leq 3$. □

An immediate corollary of Lemma 6.8 and Claim 6.9 is the following theorem.

Theorem 6.10. *(Privacy wrt Edge Changes) Given any query for social networks f, the mechanism that uses the projection μ from Claim 6.9, and answers the query using $A(f, G) = f(\mu(G)) + Lap(3 \cdot RS_f(\mathcal{H}_k)/\epsilon)$ preserves $(\epsilon, 0)$ privacy for any graph G.*

Now, it is evident that this mechanism has the guarantee that for every $G \in \mathcal{H}_k$ it holds that $\mathbf{Pr}[\ |A(f,G) - f(G)| \leq O(RS_f(\mathcal{H}_k)/\epsilon)] \geq 2/3$. Furthermore, if the querier "lucked out" to ask a query f for which $f(G)$ and $f(\mu(G))$ are close (say, identical), then the same guarantee holds for such G as well. Note however that we *cannot reveal* to the querier whether $f(G)$ and $f(\mu(G))$ are indeed close, as such information might leak privacy.

6.4.2.2 Projections and Smooth Distances Estimators

Unfortunately, the smooth projections do not always exist, as the following toy-example demonstrates. Fix n graphs, where $d(G_i, G_j) = |i-j|$ for $1 \leq i,j \leq n$, and let $\mathcal{H} = \{G_1, G_n\}$. Because $\mu(G_1) = G_1$ and $\mu(G_n) = G_n$, then there must exist some value i such that $\mu(G_i) \neq \mu(G_{i+1})$, thus every μ cannot be c-smooth for $c < n-1$.

Note that c-smooth projections have the property that they also provide a $(c+1)$-approximation of the distance of D to $\mathcal{H}$. Meaning, for every D we have that $d(D,\mathcal{H}) \leq d(D,\mu(D)) \leq (c+1) \cdot d(D,\mathcal{H})$. In the vertex adjacency model however, it is evident that we cannot have a $O(1)$-smooth projection since it is NP-hard to approximate $d(G,\mathcal{H}_k)$.

Claim 6.11. *(Privacy wrt Vertex Adjacency) Unless $P = NP$ there is no efficiently computable mapping $\mu : \mathcal{G} \to \mathcal{H}_k$ such that*

1. $\forall G \in \mathcal{H}_k, \mu(G) = G$.
2. $\forall G \in \mathcal{G}, d(G, \mu(G)) \leq O(\ln(k)\, d(G,\mathcal{H}_k))$.

Proof. (Sketch) Our reduction is from the minimum set cover problem. It is NP-hard to approximate the minimum set cover problem to a factor better than $O(\log n)$ [126, 9]. Given a set cover instance with sets $S_1, ..., S_m$ and universe $U = \{x_1, ..., x_n\}$ we set $m_i = |\{j :\ x_i \in S_j\}|$ and $k = n+1$. We construct our labeled graph G as follows:

1. Add a node for each S_i.
2. Add a node for each x_j.
3. Add the edge $\{x_j, S_i\}$ if and only if $x_j \in S_i$.
4. For each x_i, create $k+1-m_i$ fresh nodes $y_1, ..., y_{k-m_i}$ and add each edge $\{y_j, x_i\}$.

Intuitively each node x_j has $k+1$ incident edges. By deleting all of the edges incident to the node S_i we can fix all of the nodes $x \in S_i$. Hence, $d(G,\mathcal{H}_k)$ corresponds exactly to the size of the minimum set cover. □

Though computing $d(G, \mathcal{H}_k)$ is hard, we can approximate it.

Claim 6.12. *(Privacy wrt Vertex Adjacency) There is an efficiently computable projection $\mu : \mathcal{G} \to \mathcal{H}_k$ such that for every $G \in \mathcal{G}$ it holds that $d\left(G, \mu\left(G\right)\right) \leq \left(\ln\left(2d^2 + kd\right)\right) d\left(G, \mathcal{H}_k\right)$,*

Proof. First, we define the potential of a graph G as follows

$$\phi\left(G\right) = \sum_{v \in G : \deg(v) \geq k} \left(\deg\left(v\right) - k\right) \ .$$

We give an algorithm to construct μ. The algorithm traverses all options for $d\left(G, \mathcal{H}_k\right)$. For every $d \in \{1, 2, \ldots, n-k\}$ we do the following: (1) as long as there exists nodes with degree $\geq k + d + 1$ we arbitrarily pick one such node v and remove all edges touching v; (2) as long as there are nodes with degree $\geq k + 1$ we greedily pick a vertex v maximizing $\phi(G) - \phi(G - v)$ and remove all edges incident to v (we use $G - v$ to denote the graph where we remove all edges incident to v). We define $\mu(G)$ as the result of the iteration touching the minimum number of vertices.

To analyze this algorithm, we consider the output of the algorithm for the correct guess $d = d\left(G, \mathcal{H}_k\right)$. Pick a path of neighboring graphs from G to $\mathcal{H}_k$ of length d, and label each pair of neighboring graphs by the vertex that changes between the two. It is evident that the path labels must contain any node of degree $\geq k + d + 1$, since any other path of length d may reduce the degree of such nodes only to $k + 1$.

So now, denote ϕ_0 as the potential of G after step (1), and denote the set of t nodes we pick in step (2) as $v_1, v_2, \ldots, v_t$ and let ϕ_i denote the potential after removing all edges touching v_i. Observe that $\phi_0 \leq 2d^2 + kd$, because there exists a set of $\leq d$ nodes that by removing any edge incident to them we convert the graph to a graph in $\mathcal{H}_k$, and each of these nodes decrease the potential by no more than $\deg(v) + \deg(v) - k \leq (d + k) + d = 2d + k$. By standard greedy analysis, in every time we pick a vertex, there always exists some vertex that reduces the potential by a factor of $1 - \frac{1}{d}$. Therefore, after $t = d \ln(2d^2 + kd)$ iterations we have that $\phi_t \leq (1 - \frac{1}{d})^t \phi_0 < 1$. □

The reduction in Claim 6.12 might be used to produce a function $f_{\mathcal{H}}\left(G\right) = \mu\left(f\left(G\right)\right)$ with low smooth-sensitivity over the nice graphs $\mathcal{H}_k$. Unfortunately, we don't know of any efficient algorithm to compute the smooth upper bound for such $f_{\mathcal{H}}$. Yet, we show that it is possible to devise a somewhat relaxed projection s.t. the distance between a database and its mapped image is a smooth function. To that end, we relax a little the definition of projection, allowing it to map instances to some predefined $\bar{\mathcal{H}} \supset \mathcal{H}$.

Definition 6.13. *Fix $\bar{\mathcal{H}} \supset \mathcal{H}$. Let μ be a projection of $\mathcal{H}$, so μ is a mapping $\mu : \mathcal{D} \to \bar{\mathcal{H}}$ that maps every element of $\mathcal{H}$ to itself ($\forall D \in \mathcal{H}$ we have that $\mu(D) = D$). A* c-smooth distance estimator *is a function $\hat{d}_\mu : \mathcal{D} \to \mathbb{R}$ that satisfies all of the following. (1) For every $D \in \mathcal{H}$ it is defined as $\hat{d}_\mu(D) = 0$. (2) It is lower bounded by the distance of D to its projection: $\forall D \in \mathcal{D},\ \hat{d}_\mu(D) \geq d(D, \mu(D))$. (3) Its value over neighboring databases changes by at most c: $\forall D \sim D',\ \left|\hat{d}_\mu(D) - \hat{d}_\mu(D')\right| \leq c$.*

It is simple to verify that for every $D \in \mathcal{D}$ we have that $\hat{d}_\mu(D) \leq c \cdot d(D, \mathcal{H})$ (using induction on $d(D, \mathcal{H})$). We omit the subscript when μ is specified.

The following lemma suggests that a smooth distance estimator allows us to devise a good smooth-upper bound on the local-sensitivity, thus allowing us to apply the smooth-sensitivity scheme of [119].

Lemma 6.14. *Fix $\bar{\mathcal{H}} \supset \mathcal{H}$ and let $\mu : \mathcal{D} \to \bar{\mathcal{H}}$ be a projection of $\mathcal{H}$. Let $\hat{d} : \mathcal{D} \to \mathbb{R}$ be an efficiently computable c-smooth distance estimator. Then for every query f, we can define the composition $f_{\mathcal{H}} = f \circ \mu$ and define the function*

$$S_{f_{\mathcal{H}},\beta}(D) = \max_{d \in \mathbb{Z}, d \geq \hat{d}(D)} e^{\left(-\frac{\beta}{c}\left(d - \hat{d}(D)\right)\right)} (2d + c + 1) \cdot RS_f\left(\bar{\mathcal{H}}\right)$$

Then $S_{f_{\mathcal{H}},\beta}$ is an efficiently computable β-smooth upper bound on the local sensitivity of $f_{\mathcal{H}}$. Furthermore, define g as the function $g(x) = \begin{cases} 2\frac{1}{x}e^{-1+\frac{c+1}{2}x}, & 0 \leq x \leq \frac{2}{c+1} \\ c+1, & x > \frac{2}{c+1} \end{cases}$. Then for every D it holds that

$$S_{f,\beta}(D) \leq \exp(\tfrac{\beta}{c}\hat{d}(D)) \cdot g(\beta/c) RS_f(\bar{\mathcal{H}})$$

Proof. First, we show that indeed $S_{f_{\mathcal{H}},\beta}$ is an upper bound on the local sensitivity of $f_{\mathcal{H}}$.

Fix any $D \in \mathcal{D}$ and indeed

$$\begin{aligned}
LS_{f_{\mathcal{H}}}(D) &= \max_{D' \sim D} \left| f_{\mathcal{H}}(D) - f_{\mathcal{H}}(D') \right| \\
&= \max_{D' \sim D} \left| f(\mu(D)) - f(\mu(D')) \right| \\
&\leq \max_{D' \sim D} RS_f(\bar{\mathcal{H}}) \cdot d(\mu(D), \mu(D')) \\
&\leq \max_{D' \sim D} RS_f(\bar{\mathcal{H}}) \cdot \\
&\qquad \big(d(D, \mu(D)) + d(D, D') + d(D', \mu(D')) \big) \\
&\leq RS_f(\bar{\mathcal{H}}) \cdot \big(\hat{d}(D) + 1 + \max_{D' \sim D} \hat{d}(D') \big) \\
&\leq RS_f(\bar{\mathcal{H}}) \cdot \big(2\hat{d}(D) + c + 1 \big) \\
&\leq \max_{d \geq \hat{d}(D)} e^{-\beta c \left(d - \hat{d}(D) \right)} \left(2d + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right) \\
&= S_{f_{\mathcal{H}}, \beta}(D) \ .
\end{aligned}$$

Next we prove that $S_{f_{\mathcal{H}}, \beta}$ is β-smooth. Let D_1 and D_2 be two neighboring databases, and wlog assume $\hat{d}(D_2) \leq \hat{d}(D_1)$. Then

$$\begin{aligned}
&\frac{S_{f_{\mathcal{H}}, \beta}(D_1)}{S_{f_{\mathcal{H}}, \beta}(D_2)} \\
&= \frac{\max_{d \geq \hat{d}(D_1)} e^{\left(-\frac{\beta}{c} \left(d - \hat{d}(D_1) \right) \right)} \left(2d + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)}{\max_{d \geq \hat{d}(D_2)} e^{\left(-\frac{\beta}{c} \left(d - \hat{d}(D_2) \right) \right)} \left(2d + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)}
\end{aligned}$$

Let d_0 be the value of d on which the maximum of numerator is obtained. Then

$$\begin{aligned}
&\frac{S_{f_{\mathcal{H}}, \beta}(D_1)}{S_{f_{\mathcal{H}}, \beta}(D_2)} \\
&= \frac{\exp\left(-\frac{\beta}{c} \left(d_0 - \hat{d}\left(D_1 \right) \right) \right) \left(2d_0 + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)}{\max_{d \geq \hat{d}(D_2)} \exp\left(-\frac{\beta}{c} \left(d - \hat{d}\left(D_2 \right) \right) \right) \left(2d + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)} \\
&\leq \frac{\exp\left(-\frac{\beta}{c} \left(d_0 - \hat{d}\left(D_1 \right) \right) \right) \left(2d_0 + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)}{\exp\left(-\frac{\beta}{c} \left(d_0 - \hat{d}\left(D_2 \right) \right) \right) \left(2d_0 + c + 1 \right) RS_f\left(\bar{\mathcal{H}} \right)} \\
&= \exp\left(-\tfrac{\beta}{c} (\hat{d}(D_2) - \hat{d}(D_1)) \right) \leq \exp(\beta)
\end{aligned}$$

where the last inequality uses the smoothness property, i.e. that $\hat{d}(D_2) - \hat{d}(D_1) \geq -c$.

Finally, we wish to prove the global upper bound on $S_{f_{\mathcal{H},\beta}}$, i.e., that for every $D \in \mathcal{D}$

$$S_{f_{\mathcal{H},\beta}}(D) \leq \exp(\tfrac{\beta}{c}\hat{d}(D)) \cdot g(c/\beta)\, RS_f\left(\bar{\mathcal{H}}\right) .$$

Fix D and define $h(x) = \exp\left(-\frac{\beta}{c}x\right)(2x+c+1)$, so that $S_{f_{\mathcal{H},\beta}} = \exp\left(\frac{\beta}{c}\hat{d}(D)\right) RS_f(\bar{\mathcal{H}}) \cdot \max_{d \geq d_0} h(d)$. Taking the derivative of h we have

$$h'(x) = e^{-\frac{\beta}{c}x}\left(-2x\tfrac{\beta}{c} - \beta - \tfrac{\beta}{c} + 2\right)$$

which means that $h(x)$ is maximized at $x_0 = \frac{c}{\beta} - \frac{c+1}{2}$. In the case that $x_0 < 0$ (i.e. for $\beta/c > \frac{2}{c+1}$) we can upper bound the function $h(x)$ with $h(0) = c+1$ for every $x \geq 0$. Otherwise, we have that $h(x) \leq h(x_0)$ for every $x \geq 0$, and indeed $h(x_0) = 2\frac{c}{\beta}e^{-1+\frac{\beta}{c}\cdot\frac{c+1}{2}} = g(\beta/c)$.

To conclude the proof, observe that computing $S_{f_{\mathcal{H},\beta}}(D)$ is just a simple optimization once $\hat{d}(D)$ is known, much like the derivation done above. So since $\hat{d}$ is efficiently computable, we have that $S_{f_{\mathcal{H},\beta}}$ is efficiently computable. □

Like in the edge-adjacency model, we now exhibit a projection and a smooth distance estimator for the vertex-adjacency model.

Claim 6.15. *In the vertex-adjacency model, there exists a projection $\mu : \mathcal{G} \to \mathcal{H}_{2k}$ and a 4-smooth distance estimator $\hat{d}$, both of which are efficiently computable.*

Proof. To construct μ and $\hat{d}$ we start with the linear program that determines a "fractional distance" from a graph to $\mathcal{H}_k$. This LP has $n + \binom{n}{2}$ variables: x_u which intuitively represents whether x_u ought to be removed from the graph or not, and $w_{u,v}$ which represents whether the edge between u and v remains in the projected graph or not. We also use the notation $a_{u,v}$, where $a_{uv} = 1$ if the edge $\{u,v\}$ is in G; otherwise $a_{uv} = 0$.

$$\begin{aligned} \min \textstyle\sum_{v \in V} x_v \quad & s.t. \\ & (1)\ \forall v,\ x_v \geq 0 \\ & (2)\ \forall u,v,\ w_{u,v} \geq 0 \\ & (3)\ \forall u,v,\ a_{uv} \geq w_{uv} \geq a_{uv} - x_u - x_v \\ & (4)\ \forall u, \sum\nolimits_{v \neq u} w_{u,v} \leq k \end{aligned}$$

To convert our fractional solution $(\bar{x}^*, \bar{w}^*)$ to a graph $\mu(G) \in \mathcal{H}_{2k}$ we define $\mu(G)$ to be the graph we get by removing every edge $(u,v) \in E(G)$ whose either endpoint has weight $x^*_u > 1/4$ or $x^*_v \geq 1/4$. We define our distance estimator as $\hat{d}(G) = 4\sum_u x^*_u$.

We need to show that μ and $\hat{d}$ satisfy the conditions of claim 6.15. We first prove that μ is a projection mapping every graph to a graph in $\mathcal{H}_{2k}$. Suppose that some $v \in G$ has degree $\geq 2k$, then clearly $x^*_v \leq 1/4$, for otherwise we would have removed all of the edges touching v. Observe that every edge we keep has $w^*_{u,v} \geq 1 - 1/4 - 1/4 = 1/2$. Consequently, we can have at most $2k$ edges with $w_{u,v} \geq \frac{1}{2}$ because of the constraint $\sum_u w_{u,v} \leq k$. So there are at most $2k$ edges incident to v in $\mu(G)$.

Now, let us prove that $\hat{d}$ satisfies all of the requirements of a 4-smooth distance estimator. First, if $G \in \mathcal{H}_k$ then the optimal solution of the LP is the all zero vector, so $\hat{d}(G) = 0$ for all graphs of max-degree $\leq k$. Secondly, observe that in the process of computing $\mu(G)$, every edge that is removed from G can be "charged" to a vertex v with $x^*_v \geq 1/4$. If follows that

$$d(G, \mu(G)) \leq \sum_{v: x^*_v \geq 1/4} 1 \;\leq \sum_{v: x^*_v \geq 1/4} 4x^*_v \;\leq 4\sum_v x^*_v \;= \hat{d}(G)\,.$$

Lastly, fix any neighboring $G_1, G_2 \in \mathcal{G}$, and let v be the vertex whose edges differ in G_1 and G_2. Clearly, if $\bar{x}^*$ is a solution for $LP(G_1)$, then we set $y_v = 1$ for $i = 1...d$ and $y_v = x^*_v$ otherwise. Now $\vec{y}$ is a feasible (not necessarily optimal) solution to $LP(G_2)$. It is simple to infer that

$$\begin{aligned}
\hat{d}(G_2) - \hat{d}(G_1) &= \hat{d}(G_2) - 4\sum_u x^*_u \\
&\leq 4\sum_u y_u - 4\sum_u x^*_u \leq 4\sum_u |(y_u - x^*_u)| \\
&= 4\,|y_v - x^*_v| \leq 4
\end{aligned}$$

□

As before, combining Lemma 6.14 with Claim 6.15 gives the following theorem as an immediate corollary.

Theorem 6.16. *(Privacy wrt Vertex Adjacency) Given any query for social networks f, the mechanism that uses the projection μ from Claim 6.15 and the β-smooth upper bound of Lemma 6.14, and answers the query using $A(f,G) = f(\mu(G)) + Lap(2 \cdot S_{f_{\mathcal{H}}, -\epsilon/2\ln\delta}(G)/\epsilon)$ preserves (ϵ, δ) privacy for any graph G.*

Again, it is evident from the definition that the algorithm has the guarantee that for every $G \in \mathcal{H}_k$ it holds that $\mathbf{Pr}[\,|A(f,G) - f(G)| \leq O(g(\frac{\epsilon}{8\ln(1/\delta)})RS_f(\mathcal{H}_{2k})/\epsilon)] \geq 2/3$.

6.5 Restricted Sensitivity and $\mathcal{H}_k$

Now that we have constructed the machinery of restricted sensitivity, we compare the restricted sensitivity over $\mathcal{H}_k$ with smooth sensitivity for specific types of queries, in order to demonstrate the benefits of our approach. In a nutshell, restricted sensitivity offers a significant advantage over smooth sensitivity whenever $k \ll n$. I.e., we show that there are queries f s.t. for some $G \in \mathcal{H}_k$ it holds that $RS_f(\mathcal{H}_k) \ll S_{f,\beta}(G)$. We define two types of queries: local profile queries and subgraph counting queries.

6.5.1 Local Profile Queries

First, let us introduce some notation. A profile is a function that maps a vertex v in a social network (G, ℓ) to $[0, 1]$. Given a set of vertices $\{v_1, v_2, \ldots, v_t\}$, we denote by $G[v_1, v_2, \ldots, v_t]$ the social network derived by restricting G and ℓ to these t vertices. We use $G_v = G[\{v\} \cup \{w \,|(v, w) \in E(G)\}]$ to denote the social network derived by restricting G and ℓ to v and its neighbors. A *local* profile satisfies the constraint $p(v, (G, \ell)) = p(v, G_v)$.

Definition 6.17. *A* (local) profile query

$$f_p(G, \ell) = \sum\nolimits_{v \in V(G)} p\big(v, (G, \ell)\big)$$

sums the (local) profile p across all nodes.

Indeed, local profile queries are our motivating example. They capture a class of question often asked in social networks, such as: "how many people are doctors?", "how many people are friends with t doctors?", "how many people are friends with at least 7 doctors who are all friends with each other?", etc.

Claim 6.18 bounds the restricted sensitivity of a local profile query over $\mathcal{H}_k$ (e.g., in the vertex adjacency model a node v can at worst affect the local profiles of itself, its k old neighbors and its k new neighbors).

Claim 6.18. *For any local profile query f, we have that $RS_f(\mathcal{H}_k) \leq 2k + 1$ in the vertex adjacency model, and $RS_f(\mathcal{H}_k) \leq k + 1$ in the edge adjacency model.*

Proof. Consider a local profile query f_p.
(Label change) Let $G_1, G_2 \in \mathcal{H}$ be two graphs with the same exact edge set, but with labeling functions ℓ_1, ℓ_2 that are different on a single vertex. Let v be the vertex whose label

differs on G_1 and G_1, and let N_v denote the set of its (at most k) neighbors. Then for every $u \notin \{v\} \cup N_v$ we have that $p\left(u, (G_1, \ell_1)\right) = p\left(u, (G_2, \ell_2)\right)$. Hence, $|f_p\left(G_1\right) - f_p\left(G_2\right)| \leq |\{v\} \cup N_v| \leq k + 1$.
(Vertex Adjacency) Let $G_1, G_2 \in \mathcal{H}$ be any two neighboring labeled graphs such that $G_1 - v = G_2 - v$. Let N_v^1 (resp. N_v^2) denote the neighborhood of v then for any $y \notin N_v^1 \cup N_v^2$ we have that $p\left(y, (G_1, \ell_1)\right) = p\left(y, (G_2, \ell_2)\right)$. Hence, $|f_p\left(G_1\right) - f_p\left(G_2\right)| \leq |N_v^1 \cup N_v^2 \cup \{v\}| \leq 2k + 1$.
(Edge Adjacency) Let $G_1, G_2 \in \mathcal{H}$ be any two neighboring labeled graphs. Wlog, there is an edge $e = \{u, v\}$ such that $E(G_1) = E(G_2) \cup \{e\}$. In order to have a vertex y s.t. $p\left(y, (G_1, \ell_1)\right) \neq p\left(y, (G_2, \ell_2)\right)$ we need that the edge e appears in graph we get by restricting the social network to set of y and its neighbors. It follows that the only vertices whose local profile can change are in the union $\{u, v\} \cup \left(N_u \cap N_v\right)$. Hence, $|f\left(G_1\right) - f\left(G_2\right)| \leq |\{u\} \cup \{v\}| + |N_u \setminus \{v\}| \leq 2 + k - 1 = k + 1$. □

By contrast the smooth sensitivity of a local profile query may be as large as $O(n)$ even for graphs in $\mathcal{H}_k$. Consider the local profile query "how many people are friends with a spy?" In the vertex-adjacency model, the $n-1$-star graph G_1 in which a spy v is friends with everyone is adjacent to the empty graph $G_0 \in \mathcal{H}_k$. Therefore, any smooth upper bound $S_{f,\beta}$ must have $S_{f,\beta}(G) \geq n - 1$. It is also worth observing that the assumption $G \in \mathcal{H}_k$ does not necessarily shrink the range of possible answers to a local profile query f (e.g., there are graphs $G \in \mathcal{H}_k$ in which everyone is friends with a spy).

We comment that local profile queries are a natural extension of predicates to social networks, which can be used to study many interesting properties of a social network like clustering coefficients, local bridges and 2-betweeness). The clustering coefficient $c(v)$ [142, 118] of a node v (e.g., the probability that two randomly selected friends of v are friends with each other) has been used to identify teenage girls who are more likely to consider suicide [34]. One explanation, is that it becomes an inherent source of stress if a person has many friends who are not friends with each other [66]. Observe that $c(v)$ *is* a local profile query. An edge $\{v, w\}$ is a local bridge if its endpoints have no friends in common. A local profile could score a vertex v based on the number local bridges incident to v. A marketing agency may be interested in identifying nodes that are incident to many local bridges because local bridges "provide their endpoints with access to parts of the network - and hence sources of information - that they would otherwise be far away from [66]." For example, a 1995 study showed that the best job leads often come from aquaintances rather than close friends [75]. 2-betweeness (a variant of betweeness [70]) measures the centrality of a node. We say that the 2-betweeness of a vertex v is the probability that the a randomly chosen shortest path between $x, y \in G_v$, two randomly chosen neighbors of v, goes through v.

6.5.2 Subgraph Counting Queries

Subgraph queries allows us to ask questions such as "how many triplets of people are all friends when two of them are doctors and the other is a pop-singer?" or "how many paths of length 2 are there connecting a spy and a pop-singer over 40?" The average clustering coefficient of a graph can be computed from the number of triangles and 2-stars in a graph.

Definition 6.19. *A* subgraph counting *query $f = \langle H, \bar{p} \rangle$ is given by a connected graph H over t vertices and t predicates $p_1, p_2, \ldots, p_t$. Given a social network (G, ℓ), the answer to $f(G, \ell)$ is the size of the set*

$$\left\{ v_1, v_2, \ldots, v_t : \; G[v_1, v_2, \ldots, v_t] = H \text{ and } \forall i, \; \ell(v_i) \in p_i \right\}$$

The smooth sensitivity of a subgraph counting query may be as high as $O(n^{t-1})$ in the vertex adjacency model. Let $f = \langle H, \bar{p} \rangle$ be a subgraph counting query where H is a t-star and each predicate p_i is identically true. Let G_1 be a n-star ($f(G_1) = \binom{n}{c-1}$). Then in the vertex adjacency model there is a neighboring graph G_2 with no edges ($f(G_2) = 0$). We have that $LS_f(G_2) \geq \binom{n}{t-1}$. Observe that $G_2 \in \mathcal{H}_k$. We now show that the smooth sensitivity of $f = \langle K_3, \bar{p} \rangle$ is *always* greater than n when each predicate p_i is identically true.

Claim 6.20. *Let $f = \langle K_3, \bar{p} \rangle$ be a subgraph counting query with predicates p_i that are identically true. In the vertex adjacency model for any β smooth upper bound on the local sensitivity of f and* any *graph G we have $S^*_{f^P,\beta}(G) \geq \exp(-2\beta)(n-2)$.*

Proof. Let G be given. Pick $v_1, v_2 \in V(G)$ and let G_1 be obtained from G by adding all possible edges incident to v_1 and let G_2 be obtained from G_1 by deleting all edges incident to v_2. Finally, let G_3 be obtained from G_2 by adding all possible edges incident to v_2. Now the local sensitivity of f at G_2 is at least $n-2$,

$$\begin{aligned} LS_f(G_2) &= \max_{G' : d(G_2, G') = 1} |f(G_2) - f(G')| \\ &\geq f(G_3) - f(G_2) \geq n - 2 \end{aligned}$$

Plugging this lower bound into the definition of β smooth sensitivity we obtain the required result: $S^*_{f^P,\beta}(G) \geq e^{-\beta d(G,G_2)} LS_{f^P}(G_2) \geq e^{-2\beta}(n-2)$. □

By contrast Claim 6.21 bounds the restricted sensitivity of subgraph counting queries.

Claim 6.21. *Let $f = \langle H, \bar{p} \rangle$ be subgraph counting query and let $t = |H|$ then $RS_f(\mathcal{H}_k) \leq tk^{t-1}$ in the edge adjacency model and in the vertex adjacency model.*

Proof sketch. Let $G_1, G_2 \in \mathcal{H}_k$ be neighbors and let v be a vertex such that $G_1 - v = G_2 - v$, and let N_i denote the neighbors of v in G_i. Any copy of H which occurs in G_1 but not in G_2 must contain v. Because H is connected we can bound the number of G_1 copies of H. We can start with v, and we pick one of the t vertices of H to be mapped to v. Denote this vertex as v_0. Now, we proceed inductively. We pick a vertex $v \in H$, connected to the set $\{v_0, v_1, \ldots, v_{i-1}\}$. The vertex v_i must be assigned to a vertex in G which is incident to some specific vertex of the i vertices that we already mapped. Because we have bounded degree, then there are at most k options from which to choose v_i. We obtain the bound: $f(G_1) - f(G_2) \leq t \prod_{i=1}^{t-1} k = tk^{t-1}$. □

While the assumption $G \in \mathcal{H}_k$ may shrink the range of a subgraph counting query f, the restricted sensitivity of f will typically be much smaller than this reduced range. For example, if $f(G)$ counts the number of triangles in G then $f(G) \leq nk^2$ for any $G \in \mathcal{H}_k$, while $RS_f(\mathcal{H}_k) \leq 3k^2 \ll nk^2$.

6.6 Future Questions/Directions

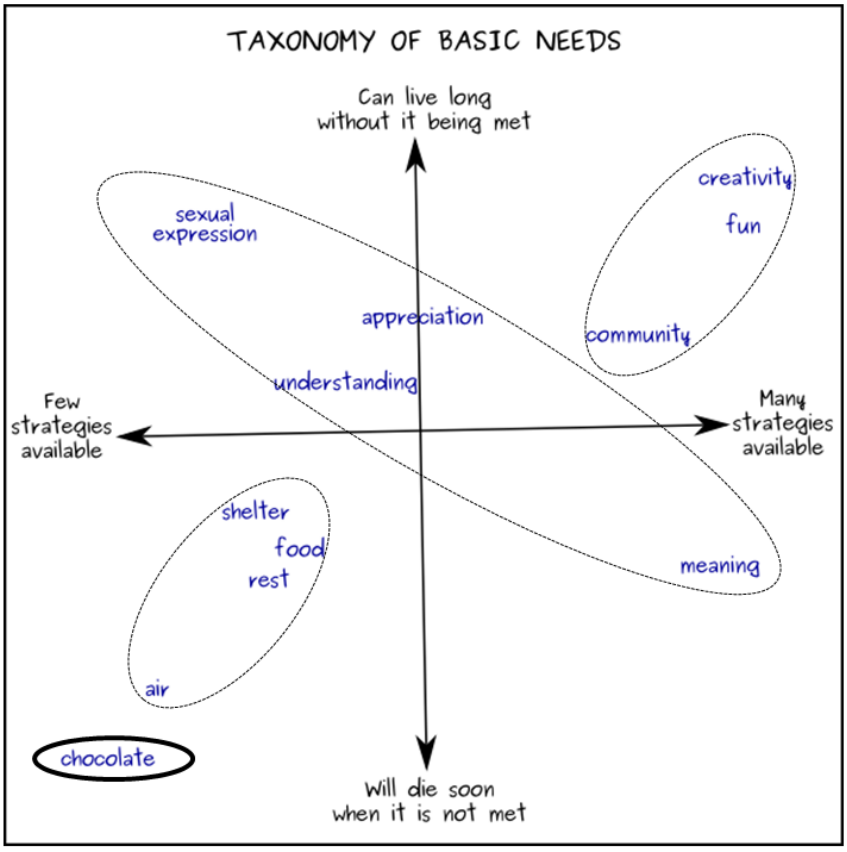
Efficient Mappings: While we can show that there doesn't exist an efficiently computable $O(1)$-smooth projection $\mu : \mathcal{G} \to \mathcal{H}_k$, we don't know whether the construction of Claim 6.15 can be improved. Meaning, there could be a mapping $\mu : \mathcal{G} \to \bar{\mathcal{H}}$ for some $\bar{\mathcal{H}} \supset \mathcal{H}_k$, whether the solution itself, the set of vertices that dominate the removed edges, is smooth. In other words, Is there an efficiently computable mapping $\mu : \mathcal{G} \to \bar{\mathcal{H}} \subset \mathcal{H}_k$ which satisfies $|d(\mu(G_1), G_1) - d(\mu(G_2), G_2)| \leq c$ for some constant c?

Multiple Queries: We primarily focus on improving the accuracy of a single query f. Could the notion of restricted sensitivity be used in conjunction with other mechanisms (e.g., exponential mechanism [43], Private Multiplicative Weights mechanism [80], etc.) to accurately answer an entire class of queries? Chapter 5 gives a mechanism that answers multiple cut-queries in the edge-adjacency model, but those are queries whose global sensitivity is 1. It would be interesting to consider answering multiple cut-queries for the vertex-adjacency model, where the query has local sensitivity of n.

Alternate Hypotheses: We focused on the specific hypothesis $\mathcal{H}_k$. What other natural hypothesis could be used to restrict sensitivity in private data analysis? Given such a hypothesis $\mathcal{H}$ can we efficiently construct a query f_H with low global sensitivity or with low smooth sensitivity over datasets $D \in \mathcal{H}$?

Part III

Clustering



Chapter 7

Stability yields a PTAS for k-Median and k-Means Clustering

7.1 Introduction

In this chapter we study the two popular clustering objectives of k-median and k-means. Both problems were surveyed in the introduction (see Chapter 4). For the problem of k-means, the work of Ostrovsky et al [121] gives a new approximation based on separability of the clustering instance (See 4.2.2). Ostrovsky et al showed that for a k-means instance for which the optimal k-clustering has cost significantly smaller than the cost of any $(k-1)$-clustering, then one can get a very good poly-time (in both n and k) approximation of the k-means objective. Specifically, if the ratio of the optimal $(k-1)$-clustering to the optimal k-clustering is at least $\max\{100, 1/\alpha\}$, then one can find a k-clustering whose cost is $(1+O(\alpha))\mathsf{OPT}$. Here, we substantially improve on this approximation guarantee. We show that under the much weaker assumption that the ratio of these costs is just at least $(1+\alpha)$ for *some* constant $\alpha > 0$, we can achieve a PTAS: namely, $(1+\epsilon)$-approximate the k-means optimum, for any constant $\epsilon > 0$. Our approximation scheme runs in time which is $\mathrm{poly}(n,k)$ and exponential only in $1/\epsilon$ and $1/\alpha$. Thus, we decouple the strength of the assumption from the quality of the conclusion, and in the process allow the assumption to be substantially weaker. For k-means clustering we in addition give a randomized algorithm with improved running time $n^{O(1)}(k \log n)^{\mathrm{poly}(1/\epsilon, 1/\alpha)}$.

We also reiterate the results of Balcan, Blum and Gupta [25] from Chapter 4.2.1. Balcan et al, motivated by the fact that objective functions are often just a proxy for the underlying goal of getting the data clustered correctly, propose clustering instances that

satisfy the condition that all $(1+\alpha)$ approximations to the given objective (e.g., k-median or k-means) are δ-close, in terms of how points are partitioned, to a target clustering (such as a correct clustering of proteins by function or a correct clustering of images by who is in them). Balcan et al prove that for any α and δ, given an instance satisfying this property for k-median or k-means objectives, one can in fact efficiently produce a clustering that is $O(\delta/\alpha)$-close to the target clustering (so, $O(\delta)$-close for any constant $\alpha > 0$), *even though obtaining a* $1+\alpha$ *approximation to the objective is* NP*-hard* for $\alpha < \frac{1}{e}$. Thus they show that one can approximate the target even though it is hard to approximate the objective. One interesting question that has remained is the approximability of the objectives when all target clusters are large compared to δn, since the hardness of approximation requires allowing small clusters.

Here, we show that for both k-median and k-means objectives, if all clusters contain more than δn points, then for any constant $\alpha > 0$ we can in fact get a PTAS. Thus, we (nearly) resolve the approximability of these objectives under this condition. Note that under this condition, this further implies finding a δ-close clustering (setting $\epsilon = \alpha$). Thus, we also extend the results of Balcan et al [25] in the case of large clusters and constant α by getting exactly δ-close for both k-median and k-means objectives. (In [25] this exact closeness was achieved for the k-median objective but needed a somewhat larger $O(\delta n(1+1/\alpha))$ minimum cluster size requirement).

Our algorithmic results are achieved by examining implications of a property we call *weak deletion-stability* that is implied by both the separation condition of Ostrovsky et al [121] as well as (when target clusters are large) the stability condition of Balcan et al [25]. In particular, an instance of k-median/k-means clustering satisfies weak deletion-stability if in the optimal solution, deleting any of the centers c_i^* and assigning all points in cluster i instead to one of the remaining $k-1$ centers c_j^*, results in an increase in the k-median/k-means cost by an (arbitrarily small) constant factor. We also show that weak deletion-stability still allows for NP-hard instances and that no FPTAS is possible as well (unless P = NP). Thus, our algorithm, whose running time is $(nk)^{\mathrm{poly}(1/\epsilon,1/\beta)}$, is optimal in the sense that the super-polynomial dependence on $1/\epsilon$ and $1/\beta$ is unavoidable.

Organization. After presenting notation and preliminaries, we introduce weak deletion-stability and relate it to the stability notions of [121] and [25] in Section 7.2. We then define another property of a clustering being β*-distributed* which, while not so intuitive, we show is implied by weak deletion-stability and will be the actual condition that our algorithms will use. We then go on to prove that being β-distributed suffices to give a PTAS for k-median in Section 7.3. We extend the algorithm to k-means clustering in Section 7.4, where we also introduce a randomized version whose run-time is bounded

by $n^3 \left((\log(n) \cdot k)\right)^{\text{poly}(1/\epsilon, 1/\beta)}$. We conclude with discussion and open problems in Section 7.5.

Notation. Throughout the chapter, we assume that we are given a set S of n points, which we partition into k disjoint subsets. When discussing k-median, we assume the n points reside in a finite metric space, and when discussing k-means, we assume they all reside in a finite dimensional Euclidean space. We also use the abbreviation $\mu_{C_i} = \frac{1}{|C_i|} \sum_{x \in C_i} x$, for the center of mass of the points in cluster C_i. We use OPT to denote the cost of the optimal clustering $\mathcal{C}^*$. We use OPT_i to denote the contribution of the cluster i to OPT, that is $\mathsf{OPT}_i = \sum_{x \in C_i^*} d(x, c_i^*)$ in the k-median case, or $\mathsf{OPT}_i = \sum_{x \in C_i^*} d^2(x, c_i^*)$ in the k-means case.

7.2 Stability Properties

As mentioned above, our results are achieved by exploiting implications of a stability condition we call weak deletion-stability, and in particular an implication we call being β-distributed. In this section we define weak deletion-stability and of being β-distributed, relate weak deletion-stability to conditions of Ostrovsky et al [121] and Balcan et al [25], and show that weak deletion-stability implies the clustering is β-distributed. In Sections 7.3 and 7.4 we use the property of being β-distributed to obtain a PTAS.[1]

Definition 7.1. *For $\alpha > 0$, a k-median/k-means instance satisfies* $(1+\alpha)$ weak deletion-stability*, if it has the following property. Let $\{c_1^*, c_2^*, \ldots, c_k^*\}$ denote the centers in the optimal k-median/k-means solution. Let OPT denote the optimal k-median/k-means cost and let $\mathsf{OPT}^{(i \to j)}$ denote the cost of the clustering obtained by removing c_i^* as a center and assigning all its points instead to c_j^*. Then for any $i \neq j$, it holds that*

$$\mathsf{OPT}^{(i \to j)} > (1+\alpha)\mathsf{OPT}$$

We use weak deletion-stability via the following implication we call being β-distributed.

Definition 7.2. *For $\beta > 0$, a k-median instance is* β-distributed *if for any center c_i^* of the optimal clustering and any data point $x \notin C_i^*$, it holds that*

$$d(x, c_i^*) \geq \beta \cdot \frac{\mathsf{OPT}}{|C_i^*|}.$$

[1]Technically, we could skip the "middleman" of weak deletion-stability and just define the property of being β-distributed as our main stability notion, but weak deletion-stability is a more intuitive condition.

A k-means instance is β-distributed if for any such c_i^* and $x \notin C_i^*$, it holds that

$$d^2(x, c_i^*) \geq \beta \cdot \frac{\mathsf{OPT}}{|C_i^*|}$$

We prove that $(1+\alpha)$ weak deletion-stability implies the clustering is $\alpha/2$-distributed for k-median ($\alpha/4$-distributed for k-means) in Theorem 7.5 below. First, however, we relate weak deletion-stability to the conditions considered in [121] and [25].

7.2.1 ORSS-Separability

Ostrovsky, Rabani, Schulman and Swamy [121] define a clustering instance to be ϵ-separated if the optimal k-means solution is cheaper than the optimal $(k-1)$-means solution by at least a factor ϵ^2. For a given objective (k-means or k-median) let us use $\mathsf{OPT}_{(k-1)}$ to denote the cost of the optimal $(k-1)$-clustering. Introducing a parameter $\alpha > 0$, say a clustering instance is $(1+\alpha)$-*ORSS separable* if

$$\frac{\mathsf{OPT}_{(k-1)}}{\mathsf{OPT}} > 1 + \alpha$$

If an instance satisfies $(1+\alpha)$-ORSS separability then all $(k-1)$ clusterings must have cost more than $(1+\alpha)\mathsf{OPT}$ and hence it is immediately evident that the instance will also satisfy $(1+\alpha)$-weak deletion-stability. Hence we have the following claim:

Claim 7.3. *Any $(1+\alpha)$-ORSS separable k-median/k-means instance is also $(1+\alpha)$-weakly deletion stable.*

7.2.2 BBG-Stability

Balcan, Blum, and Gupta [25] (see also Balcan and Braverman [28] and Balcan, Röglin, and Teng [30]) consider a notion of stability to approximations motivated by settings in which there exists some (unknown) target clustering $\mathcal{C}^{target}$ we would like to produce. Balcan et al [25] define a clustering instance to be $(1+\alpha, \delta)$ approximation-stable with respect to some objective Φ (such as k-median or k-means), if any k-partition whose cost under Φ is at most $(1+\alpha)\mathsf{OPT}$ agrees with the target clustering on all but at most δn data points. That is, for any $(1+\alpha)$ approximation $\mathcal{C}$ to objective Φ, we have $\min_{\sigma \in S_k} \sum_i |C_i^{target} - C_{\sigma(i)}| \leq \delta n$ (here, σ is simply a matching of the indices in the target clustering to those in $\mathcal{C}$). In general, δn may be larger than the smallest target cluster

size, and in that case approximation-stability need not imply weak deletion-stability (not surprisingly since [25] show that k-median and k-means remain hard to approximate). However, when all target clusters have size greater than δn (note that δ need not be a constant) then approximation-stability indeed also implies weak deletion-stability, allowing us to get a PTAS (and thereby δ-close to the target) when $\alpha > 0$ is a constant.

Claim 7.4. *A k-median/k-means clustering instance that satisfies $(1+\alpha, \delta)$ approximation-stability, and in which all clusters in the target clustering have size greater than δn, also satisfies $(1 + \alpha)$ weak deletion-stability.*

Proof. Consider an instance of k-median/k-means clustering which satisfies $(1 + \alpha, \delta)$ approximation-stability. As before, let $\{c_1^*, c_2^*, \ldots, c_k^*\}$ be the centers in the optimal solution and consider the clustering $C^{(i\to j)}$ obtained by no longer using c_i^* as a center and instead assigning each point from cluster i to c_j^*, making the ith cluster empty. The distance of this clustering from the target is defined as $\frac{1}{n}\min_{\sigma\in S_k}\sum_{i'}|C_{i'}^{target} - C_{\sigma(i')}^{(i\to j)}|$. Since $C^{(i\to j)}$ has only $(k-1)$ nonempty clusters, one of the target clusters must map to an empty cluster under any permutation σ. Since by assumption, this target cluster has more than δn points, the distance between C^{target} and $C^{(i\to j)}$ will be greater than δ and hence by the BBG stability condition, the k-median/k-means cost of $C^{(i\to j)}$ must be greater than $(1 + \alpha)\mathsf{OPT}$. □

7.2.3 Weak Deletion-Stability implies β-distributed

We show now that weak deletion-stability implies the instance is β-distributed.

Theorem 7.5. *Any $(1+\alpha)$-weakly deletion-stable k-median instance is $\frac{\alpha}{2}$-distributed. Any $(1 + \alpha)$-weakly deletion-stable k-means instance is $\frac{\alpha}{4}$-distributed.*

Proof. Fix any center in the optimal k-clustering, c_i^*, and fix any point p that does not belong to the C_i^* cluster. Denote by C_j^* the cluster that p is assigned to in the optimal k-clustering. Therefore it must hold that $d(p, c_j^*) \leq d(p, c_i^*)$. Consider the clustering obtained by deleting c_i^* from the list of centers, and assigning each point in C_i^* to C_j^*. Since the instance is $(1 + \alpha)$-weakly deletion-stable, this should increase the cost by at least $\alpha\mathsf{OPT}$.

Suppose we are dealing with a k-median instance. Each point $x \in C_i^*$ originally pays $d(x, c_i^*)$, and now, assigned to c_j^*, it pays $d(x, c_j^*) \leq d(x, c_i^*) + d(c_i^*, c_j^*)$. Thus, the new cost of the points in C_i^* is upper bounded by $\sum_{x\in C_i^*} d(x, c_j^*) \leq \mathsf{OPT}_i + |C_i^*|d(c_i^*, c_j^*)$. As the increase in cost is lower bounded by $\alpha\mathsf{OPT}$ and upper bounded by $|C_i^*|d(c_i^*, c_j^*)$,

we deduce that $d(c_i^*, c_j^*) > \alpha \frac{\mathsf{OPT}}{|C_i^*|}$. Observe that triangle inequality gives that $d(c_i^*, c_j^*) \leq d(c_i^*, p) + d(p, c_j^*) \leq 2d(c_i^*, p)$, so we have that $d(c_i^*, p) > (\alpha/2)\frac{\mathsf{OPT}}{|C_i^*|}$.

Suppose we are dealing with a Euclidean k-means instance. Again, we have created a new clustering by assigning all points in C_i^* to the center c_j^*. Thus, the cost of transitioning from the optimal k-clustering to this new $(k-1)$-clustering, which is at least $\alpha\mathsf{OPT}$, is upper bounded by $\sum_{x \in C_i^*} \|x - c_j^*\|^2 - \|x - c_i^*\|^2$. As $c_i^* = \mu_{C_i^*}$, it follows that this bound is *exactly* $\sum_{x \in C_i^*} \|c_j^* - c_i^*\|^2 = |C_i^*| d^2(c_i^*, c_j^*)$, see [88] (§2, Theorem 2). It follows that $d^2(c_i^*, c_j^*) > \alpha \frac{\mathsf{OPT}}{|C_i^*|}$. As before, $d^2(c_i^*, c_j^*) \leq \big(d(c_i^*, p) + d(p, c_j^*)\big)^2 \leq 4d^2(c_i^*, p)$, so $d^2(c_i^*, p) > \frac{\alpha}{4}\frac{\mathsf{OPT}}{|C_i^*|}$. □

7.2.4 NP-hardness under weak deletion-stability

Finally, we would like to point out that NP-hardness of the k-median problem in maintained even if we restrict ourselves only to weakly deletion-stable instances. Also the reduction sketched below uses only integer poly-size distances, and hence rules out the existence of a FPTAS for the problem, unless P = NP. In addition, the reduction can be modified to show that NP-hardness is maintained under the conditions studied in [121] and [25].

Theorem 7.6. *For any constant $\alpha > 0$, finding the optimal k-median clustering of $(1+\alpha)$-weakly deletion-stable instances is* NP*-hard.*

Proof Sketch. Fix any constant $\alpha > 0$. We give a 1-1 poly-time reduction from Set-Cover to $(1+\alpha)$-weakly deletion-stable k-median instances. Under standard notation, we assume our input consists of n subsets of a given universe of size m, for which we seek a k-cover. We reduce such an instance to a k-median instance over $m + k(n + 4\beta km)$ points. We start with the usual reduction of Set-Cover to an instance with m points representing the items of the universe and n points representing all possible sets. Fix integer $D \gg 1$ to be chosen later. If j belongs to the ith set, fix the distance $d(i, j) = D$, otherwise we fix the distance $d(i, j) = D + 1$, and between any two set-points we fix the distance to be 1. (The distance between any two item points is shortest-path distance.) However, we augment the n set-points with additional $2mD$ points, setting the distance between all of the $(n + 2mD)$ points as 1. Furthermore, we replicate k copies of the augmented set-point, all connected only via the m-item points.

Observe that each of the k copies of our augmented set-points components contains many points, and all points outside this copy are of distance $\geq D$ from it. Therefore, in

the optimal k-median solution, each center resides in one unique copy of the augmented set-points. Now, if our Set-Cover instance has a k-cover, then we can pick the respective centers and have an optimal solution with cost exactly $k(n+2mD-1)+mD$. Otherwise, no k sets cover all m items, so for any k centers, some item-point must have distance $D+1$ from its center, and so the cost of any k-partition is $\geq k(n+2mD-1)+mD+1$. Furthermore, the resulting instance is $(1+\alpha)$ weakly deletion-stable as using one center from each augmented set-point results in a k-median solution of cost $\leq m(D+1)+k(n+2mD-1)$. Hence, OPT is atmost this quantity. However, in any $k-1$ clustering, one of the set-points must pay a high cost and hence $\mathsf{OPT}_{(k-1)} \geq (m-1)D+(k-1)(n+2mD-1)+(n+2mD)D$. One can choose D large enough so that this cost is at least $(1+\alpha)\mathsf{OPT}$. □

7.3 A PTAS for any β-distributed k-Median Instance

We now present the algorithm for finding a $(1+\epsilon)$-approximation of the k-median optimum for β-distributed instances. First, we comment that using a standard doubling technique, we can assume we approximately know the value of OPT.[2] Our algorithm works if instead of OPT we use a value v s.t. $\mathsf{OPT} \leq v \leq (1+\epsilon/2)\mathsf{OPT}$, but for ease of exposition, we assume that the exact value of OPT is known.

Below, we informally describe the algorithm for a special case of β-distributed instances in which no cluster dominates the overall cost of the optimal clustering. Specifically, we say a cluster C_i^* in the optimal k-median clustering $\mathcal{C}^*$ (hereafter also referred to as the target clustering) is *cheap* if $\mathsf{OPT}_i \leq \frac{\beta\epsilon\mathsf{OPT}}{32}$, otherwise, we say C_i^* is *expensive*. Note that in any event, there can be at most a constant $(\frac{32}{\beta\epsilon})$ number of expensive clusters.

Algorithm Intuition: The intuition for our algorithm and for introducing the notion of cheap clusters is the following. Pick some cluster C_i^* in the optimal k-median clustering. Since the instance is β-distributed, any $x \notin C_i^*$ is far from c_i^*, namely, $d(x,c_i^*) > \beta\frac{\mathsf{OPT}}{|C_i^*|}$. In contrast, the average distance of $x \in C_i^*$ from c_i^* is $\frac{\mathsf{OPT}_i}{|C_i^*|}$. Thus, if we focus on a cluster whose contribution, OPT_i, is no more than, say, $\frac{\beta}{100}\mathsf{OPT}$, we have that c_i^* is 100 times closer, on *average*, to the points of C_i^* than to the points outside C_i^*. Furthermore, using the triangle inequality we have that any two "average" points of C_i^* are of distance at most $\frac{2\beta}{100}\frac{\mathsf{OPT}}{|C_i^*|}$, while the distance between any such "average" point and any point outside of C_i^* is at least $\frac{99\beta}{100}\frac{\mathsf{OPT}}{|C_i^*|}$. So, if we manage to correctly guess the size s of a cheap cluster, we can

[2]Instead of doubling from 1, we can alternatively run an off-the-shelf 5-approximation of OPT, which will return a value $v \leq 5\mathsf{OPT}$.

set a radius $r = \Theta\left(\beta \frac{\mathsf{OPT}}{s}\right)$ and collect data-points according to the size and intersection of the r-balls around them. We note that this use of balls with an inverse relation between size and radius is similar to that in the min-sum clustering algorithm of [28].

Note that in the general case we might have up to $\frac{32}{\beta\epsilon}$ expensive clusters. We handle them by brute force guessing their centers. In Subsection 7.3.1, we present the algorithm for clustering β-distributed instances of k-median under the assumption that for all the expensive clusters we have made the correct guess for their cluster centers. The algorithm populates a list $\mathcal{Q}$, where each element in this list is a subset of points. Ideally, each subset is contained in some target cluster, yet we might have a few subsets with points from two or more target clusters. The first stage of the algorithm is to add components into $\mathcal{Q}$, and the second stage is to find k good components in $\mathcal{Q}$, and use these k components to retrieve a clustering with low cost.

Since we do not have many expensive clusters, we can run the algorithm for all possible guesses for the centers of the expensive clusters and choose the solution which has the minimum cost. The analysis below shows that one such guess will lead to a solution of cost at most $(1+\epsilon)\mathsf{OPT}$. Later, in Section 7.4, when we deal with k-means in Euclidean space, we use sampling techniques, similar to those of Kumar et al [105] and Ostrovsky et al [121], to get good substitutes for the centers of the expensive clusters. Note however an important difference between the approach of [105, 121] and ours. While they sample points from all k clusters, we sample points only for the $O(1)$ expensive clusters. As a result, the runtime of the PTAS of [105, 121] has exponential dependence in k, while ours has only a polynomial dependence in k.

7.3.1 Clustering β-distributed Instances

The algorithm is presented in Figure 7.1. In this section we assume that at the beginning, the list $\mathcal{Q}$ is initialized with $\mathcal{Q}_{init}$ which contains the centers of all the expensive clusters. In general, the algorithm will be run several times with $\mathcal{Q}_{init}$ containing different guesses for the centers of the expensive clusters. Before going into the proof of correctness of the algorithm, we introduce another definition. We define the *inner ring* of C_i^* as the set $\left\{x;\ d(x,c_i^*) \leq \frac{\beta\mathsf{OPT}}{8|C_i^*|}\right\}$. Note the following fact:

Fact 7.7. *If C_i^* is a cheap cluster, then no more than an $\epsilon/4$ fraction of its points reside outside the inner ring. In particular, at least half of a cheap cluster is contained within the inner ring.*

Proof. This follows from Markov's inequality. If more than $(\epsilon/4)|C_i^*|$ points are outside

1. **Initialization Stage:** Set $\mathcal{Q} \leftarrow \mathcal{Q}_{init}$.

2. **Population Stage:** For $s = n, n-1, n-2, \ldots, 1$ do:

   (a) Set $r = \frac{\beta \mathsf{OPT}}{4s}$.

   (b) Remove any point x such that $d(x, \mathcal{Q}) < 2r$.
   (Here, $d(x, \mathcal{Q}) = \min_{T \in \mathcal{Q}; y \in T} d(x, y)$.)

   (c) For any remaining data point x, denote the set of data points whose distance from x is at most r, by $B(x, r)$. Connect any two remaining points a and b if: (i) $d(a, b) \leq r$, (ii) $|B(a, r)| > \frac{s}{2}$ and (iii) $|B(b, r)| > \frac{s}{2}$.

   (d) Let T be a connected component of size $> \frac{s}{2}$. Then:

      i. Add T to $\mathcal{Q}$. (That is, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{T\}$.)

      ii. Define the set $B(T) = \{x : \ d(x, y) \leq 2r \text{ for some } y \in T\}$. Remove the points of $B(T)$ from the instance.

3. **Centers-Retrieving Stage:** For any choice of k components $T_1, T_2, \ldots, T_k$ out of $\mathcal{Q}$ (we later show that $|\mathcal{Q}| < k + O(1/\beta)$)

   (a) Find the best center c_i for $T_i \cup B(T_i)$. That is $c_i = \arg\min_{p \in T_i \cup B(T_i)} \sum_{x \in T_i \cup B(T_i)} d(x, p)$.[a]

   (b) Partition all n points according to the nearest point among the k centers of the current k components.

   (c) If a clustering of cost at most $(1+\epsilon)\mathsf{OPT}$ is found – output these k centers and halt.

[a] This can be done before fixing the choice of k components out of $\mathcal{Q}$.

Figure 7.1: A PTAS for β-distributed instances of k-median.

of the inner ring, then $\mathsf{OPT}_i > \frac{\epsilon|C_i^*|}{4} \cdot \frac{\beta\mathsf{OPT}}{8|C_i^*|} = \beta\epsilon\mathsf{OPT}/32$. This contradicts the fact that C_i^* is cheap. □

Our high level goal is to show that for any cheap cluster C_i^* in the target clustering, we insert a component T_i that is contained within C_i^*, and furthermore, contains only points that are close to c_i^*. It will follow from the next claims that the component T_i is the one that contains points from the inner ring of C_i^*. We start with the following Lemma which we will utilize a few times.

Lemma 7.8. *Let T be any component added to $\mathcal{Q}$. Let s be the stage in which we add T to $\mathcal{Q}$. Let C_i^* be any cheap cluster s.t. $s \geq |C_i^*|$. Then (a) T does not contain any point z s.t. the distance $d(c_i^*, z)$ lies within the range $\left[\frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}, \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}\right]$, and (b) T cannot contain both a point p_1 s.t. $d(c_i^*, p_1) < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}$ and a point p_2 s.t. $d(c_i^*, p_2) > \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}$.*

Proof. We prove (a) by contradiction. Assume T contains a point z s.t. $\frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|} \leq d(c_i^*, z) \leq \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}$. Set $r = \frac{\beta\mathsf{OPT}}{4s} \leq \frac{\beta\mathsf{OPT}}{4|C_i^*|}$, just as in the stage when T was added to $\mathcal{Q}$, and let p be any point in the ball $B(z, r)$. Then by the triangle inequality we have that $d(c_i^*, p) \geq d(c_i^*, z) - d(z, p) \geq \frac{\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}$, and similarly $d(c_i^*, p) \leq d(c_i^*, z) + d(z, p) \leq \frac{\beta\mathsf{OPT}}{|C_i^*|}$. Since our instance is β-distributed it holds that p belongs to C_i^*, and from the definition of the inner ring of C_i^*, it holds that p falls *outside* the inner ring. However, z is added to T because the ball $B(z, r)$ contains more than $s/2 \geq |C_i^*|/2$ many points. So more than half of the points in C_i^* fall outside the inner ring of C_i^*, which contradicts Fact 7.7.

Assume now (b) does not hold. Recall that T is a connected component, so exists some path $p_1 \to p_2$. Each two consecutive points along this path were connected because their distance is at most $\frac{\beta\mathsf{OPT}}{4s} \leq \frac{\beta\mathsf{OPT}}{4|C_i^*|}$. As $d(c_i^*, p_1) < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}$ and $d(c_i^*, p_2) > \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}$, there must exist a point z along the path whose distance from c_i^* falls in the range $\left[\frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}, \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}\right]$, contradicting (a). □

Claim 7.9. *Let C_i^* be any cheap cluster in the target clustering. By stage $s = |C_i^*|$, the algorithm adds to $\mathcal{Q}$ a component T that contains a point from the inner ring of C_i^*.*

Proof. Suppose that up to the stage $s = |C_i^*|$ the algorithm has not inserted such a component into $\mathcal{Q}$. Now, it is possible that by stage s, the algorithm has inserted some component T' to $\mathcal{Q}$, s.t. some x in the inner ring of C_i^* is too close to some $y \in T'$ (namely, $d(x, y) \leq 2r$), thus causing x to be removed from the instance. Assume for now this is not the case. This means that the inner ring of cluster C_i^* still contains more than $|C_i^*|/2$ points. Also observe that all inner ring points are of distance at most $\frac{\beta\mathsf{OPT}}{8|C_i^*|}$ from the center,

so every pair of inner ring points has a distance of at most $\frac{\beta \mathsf{OPT}}{4|C_i^*|}$. Hence, when we reach stage $s = |C_i^*|$, any ball of radius $r = \frac{\beta \mathsf{OPT}}{4s} = \frac{\beta \mathsf{OPT}}{4|C_i^*|}$ centered at any inner-ring point, must contain all other inner-ring points. This means that at stage $s = |C_i^*|$ all inner ring points are connected among themselves, so they form a component (in fact, a clique) of size $> s/2$. Therefore, the algorithm inserts a new component, containing all inner ring points.

So, by stage $s = |C_i^*|$, one of two things can happen. Either the algorithm inserts a component that contains some inner ring point to Q, or the algorithm removes an inner ring point due to some component $T' \in \mathcal{Q}$. If the former happens, we are done. So let us prove by contradiction that we cannot have only the latter.

Let $s \geq |C_i^*|$ be the stage in which we throw away the first inner ring point of the cluster C_i^*. At stage s the algorithm removes this inner ring point x because there exists a point y in some component $T' \in Q$, s.t. $d(x,y) \leq 2r = \frac{\beta \mathsf{OPT}}{2s}$, and so $d(c_i^*, y) \leq d(c_i^*, x) + d(x,y) \leq \frac{\beta \mathsf{OPT}}{8|C_i^*|} + \frac{\beta \mathsf{OPT}}{2s} \leq \frac{5}{8}\frac{\beta \mathsf{OPT}}{|C_i^*|}$. This immediately implies that T' cannot be the center of an expensive cluster since any such point will be at a distance at least $\frac{\beta \mathsf{OPT}}{|C^*|}$ from c_i^*. Let $s' \geq s \geq |C_i^*|$ be the previous stage in which we added the component T' to Q. As Lemma 7.8 applies to T', we deduce that $d(c_i^*, y) < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}$. Recall that T' contains $> s'/2 \geq |C_i^*|/2$ many points, yet, by assumption, contains none of the $|C_i^*|/2$ points that reside in the inner ring of C_i^*. It follows from Fact 7.7 that some point $w \in T'$ must belong to a different cluster C_j^*. Since the instance is β-distributed, we have that $d(c_i^*, w) > \frac{\beta \mathsf{OPT}}{|C_i^*|}$. The existence of both y and w in T' contradicts part (b) of Lemma 7.8. □

We call a component $T \in \mathcal{Q}$ *good* if it contains an inner ring point of some cheap cluster C_i^*. A component is called *bad* if it is not good and is not one of the initial centers present in $\mathcal{Q}_{init}$. We now discuss the properties of good components.

Claim 7.10. *Let T be a good component added to $\mathcal{Q}$, containing an inner ring point from a cheap cluster C_i^*. (By Claim 7.9 we know at least one such T exists.) Then: (a) all points in T are of distance at most $\frac{\beta \mathsf{OPT}}{2|C_i^*|}$ from c_i^*, (b) $T \cup B(T)$ is fully contained in C_i^*, and (c) the entire inner ring of C_i^* is contained in $T \cup B(T)$, and (d) no other component $T' \in \mathcal{Q}$, $T' \neq T$, contains an inner ring point from C_i^*.*

Proof. As we do not know (d) in advance, it might be the case that $\mathcal{Q}$ contains many good components, all containing an inner-ring point from the same cluster, C_i^*. Out of these (potentially many) components, let T denote the first one inserted to $\mathcal{Q}$. Denote the stage in which T was inserted to $\mathcal{Q}$ as s. Due to the previous claim, we know $s \geq |C_i^*|$, and so

Lemma 7.8 applies to T. We show (a), (b), (c) and (d) hold for T, and deduce that T is the only good component to contain an inner ring point from C_i^*.

Part (a) follows immediately from Lemma 7.8. We know T contains some inner ring point x from C_i^*, so $d(c_i^*, x) \leq \frac{\beta}{8}\frac{\mathsf{OPT}}{|C_i^*|} < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}$, so we know that any $y \in T$ must satisfy that $d(c_i^*, y) < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C_i^*|}$. Since we now know (a) holds and the instance is β-distributed, we have that $T \subset C_i^*$, so we only need to show $B(T) \subset C_i^*$. Fix any $y \in B(T)$. The point y is assigned to $B(T)$ (thus removed from the instance) because there exists some point $x \in T$ s.t. $d(x, y) \leq 2r$. So again, we have that $d(c_i^*, y) \leq d(c_i^*, x) + d(x, y) \leq \frac{\beta\mathsf{OPT}}{|C_i^*|}$, which gives us that $y \in C_i^*$ (since the instance is β-distributed).

We now prove (c). Because of (b), we deduce that the number of points in T is at most $|C_i^*|$. However, in order for T to be added to $\mathcal{Q}$, it must also hold that $|T| > s/2$. It follows that $s < 2|C_i^*|$. Let x be an inner ring point of C_i^* that belongs to T. Then the distance of any other inner ring point of C_i^* and x is at most $\frac{\beta\mathsf{OPT}}{4|C_i^*|} < \frac{\beta\mathsf{OPT}}{2s} = 2r$. It follows that any inner ring point of C_i^* which isn't added to T is assigned to $B(T)$. Thus $T \cup B(T)$ contains all inner-ring points. Finally, observe that (d) follows immediately from the definition of a good component and from (c). □

We now show that in addition to having all k good components, we cannot have too many bad components.

Claim 7.11. *We have less than* $16/(3\beta)$ *bad components.*

Proof. Let T be a bad component, and let s be the stage in which T was inserted to $\mathcal{Q}$. Let y be any point in T, and let C^* be the cluster to which y belongs in the optimal clustering with center c^*. We show $d(c^*, y) > \frac{3\beta}{8}\frac{\mathsf{OPT}}{s}$. We divide into cases.

Case 1: C^* is an expensive cluster. Note that we are working under the assumption that $\mathcal{Q}_{init}$ contains the correct centers of the expensive clusters. In particular, $\mathcal{Q}_{init}$ contains c^*. Also, the fact that point y was not thrown out in stage s implies that $d(c^*, y) > 2r = \frac{\beta\mathsf{OPT}}{2s} > \frac{3\beta\mathsf{OPT}}{8s}$.

Case 2: C^* is a cheap cluster and $s \geq |C^*|$. We apply Lemma 7.8, and deduce that either $d(c^*, y) < \frac{\beta}{2}\frac{\mathsf{OPT}}{|C^*|}$ or that $d(c^*, y) > \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C^*|} \geq \frac{3\beta}{4}\frac{\mathsf{OPT}}{s}$. As the inner ring of C^* contains $> |C^*|/2$ and T contains $> s/2 \geq |C^*|/2$ many points, none of which is an inner ring point, some point $w \in T$ does not belong to C^* and hence $d(c^*, w) > \frac{\beta\mathsf{OPT}}{|C^*|} > \frac{3\beta}{4}\frac{\mathsf{OPT}}{|C^*|}$. Part (b) of Lemma 7.8 assures us that all points in T are also far from c^*.

Case 3: C^* is a cheap cluster and $s < |C^*|$. Using Claim 7.9 we have that some good component containing a point x from the inner ring of C^* was already added to $\mathcal{Q}$. So it

must hold that $d(x,y) > 2r$, for otherwise we removed y from the instance and it cannot be added to any T. We deduce that $d(c^*,y) \geq d(x,y) - d(c^*,x) \geq \frac{\beta \mathsf{OPT}}{2s} - \frac{\beta \mathsf{OPT}}{8|C^*|} > \frac{3\beta}{8}\frac{\mathsf{OPT}}{s}$.

All points in T have distance $> \frac{3\beta \mathsf{OPT}}{8s}$ from their respective centers in the optimal clustering, and recall that T is added to $\mathcal{Q}$ because T contains at least $s/2$ many points. Therefore, the contribution of all elements in T to OPT is at least $\frac{3\beta \mathsf{OPT}}{16}$. It follows that we can have no more than $16/3\beta$ such bad components. □

We can now prove the correctness of our algorithm.

Theorem 7.12. *The algorithm outputs a k-clustering whose cost is no more than $(1+\epsilon)\mathsf{OPT}$.*

Proof. Using Claim 7.10, it follows that there exists some choice of k components, $T_1, \ldots, T_k$, such that we have the center of every expensive cluster and the good component corresponding to every cheap cluster $\mathcal{C}^*$. Fix that choice. We show that for the optimal clustering, replacing the true centers $\{c_1^*, c_2^*, ..., c_k^*\}$ with the centers $\{c_1, c_2, ..., c_k\}$ that the algorithm outputs, increases the cost by at most a $(1+\epsilon)$ factor. This implies that using the $\{c_1, c_2, ..., c_k\}$ as centers must result in a clustering with cost at most $(1+\epsilon)\mathsf{OPT}$.

Fix any C_i^* in the optimal clustering. Let OPT_i be the cost of this cluster. If C_i^* is an expensive cluster then we know that its center c_i^* is present in the list of centers chosen. Hence, the cost paid by points in C_i^* will be at most OPT_i. If C_i^* is a cheap cluster then denote by T the good component corresponding to it. We break the cost of C_i^* into two parts: $\mathsf{OPT}_i = \sum_{x \in C_i^*} d(x, c_i^*) = \sum_{x \in T \cup B(T)} d(x, c_i^*) + \sum_{x \in C_i^*, \text{ yet } x \notin T \cup B(T)} d(x, c_i^*)$ and compare it to the cost C_i^* using c_i, the point picked by the algorithm to serve as center: $\sum_{x \in C_i^*} d(x, c_i) = \sum_{x \in T \cup B(T)} d(x, c_i) + \sum_{x \in C_i^*, \text{ yet } x \notin T \cup B(T)} d(x, c_i)$. Now, the first term is exactly the function that is minimized by c_i, as $c_i = \arg\min_p \sum_{x \in T \cup B(T)} d(x,p)$. We also know c_i^*, the actual center of C_i^*, resides in the inner ring, and therefore, by Claim 7.10 must belong to $T \cup B(T)$. It follows that $\sum_{x \in T \cup B(T)} d(x, c_i) \leq \sum_{x \in T \cup B(T)} d(x, c_i^*)$. We now upper bound the 2nd term, and show that $\sum_{x \in C_i^*, \text{ yet } x \notin T \cup B(T)} d(x, c_i) \leq (1+\epsilon) \sum_{x \in C_i^*, \text{ yet } x \notin T \cup B(T)} d(x, c_i^*)$

Any point $x \in C_i^*$, s.t. $x \notin T \cup B(T)$, must reside outside the inner ring of C_i^*. Therefore, $d(x, c_i^*) > \frac{\beta \mathsf{OPT}}{8|C_i^*|}$. We show that $d(c_i, c_i^*) \leq \epsilon \frac{\beta \mathsf{OPT}}{8|C_i^*|}$, and thus we have that $d(x, c_i) \leq d(x, c_i^*) + d(c_i^*, c_i) \leq (1+\epsilon) d(x, c_i^*)$, which gives the required result.

Note that thus far, we have only used the fact that the cost of any cheap cluster is proportional to $\beta \mathsf{OPT}/|C_i^*|$. Here is the first (and the only) time we use the fact that the cost is actually at most $(\epsilon/32) \cdot \beta \mathsf{OPT}/|C_i^*|$. Using the Markov inequality, we have that the set of points satisfying $\{x;\ d(x, c_i^*) \leq \epsilon \cdot \beta \mathsf{OPT}/(16|C_i^*|)\}$ contains at least half of

the points in C_i^*, and they all reside in the inner ring, thus belong to $T \cup B(T)$. Assume for the sake of contradiction that $d(c_i, c_i^*) \geq \epsilon \frac{\beta \mathsf{OPT}}{8|C_i^*|}$. Then at least half of the points in C_i^* contribute more than $\epsilon \frac{\beta \mathsf{OPT}}{16|C_i^*|}$ to the sum $\sum_{x \in T \cup B(T)} d(x, c_i)$. It follows that this sum is more than $\epsilon \frac{\beta \mathsf{OPT}}{32|C_i^*|} \geq \mathsf{OPT}_i$. However, c_i is the point that minimizes the sum $\sum_{x \in T \cup B(T)} d(x, p)$, and by using $p = c_i^*$ we have $\sum_{x \in T \cup B(T)} d(x, p) \leq \mathsf{OPT}_i$. Contradiction.

□

7.3.2 Runtime analysis

A naive implementation of the 2nd step of algorithm in Section 7.3.1 takes $O(n^3)$ time (for every s and every point x, find how many of the remaining points fall within the ball of radius r around it). Finding c_i for all components takes $O(n^2)$ time, and measuring the cost of the solution using a particular set of k data points as centers takes $O(nk)$ time. Guessing the right k components takes $k^{O(1/\beta)}$ time. Overall, the running time of the algorithm in Figure 7.1 is $O(n^3 k^{O(1/\beta)})$. The general algorithm that brute-force guesses the centers of all expensive clusters, makes $n^{O(1/\beta\epsilon)}$ iterations of the given algorithm, so its overall running time is $n^{O(1/\beta\epsilon)} k^{O(1/\beta)}$.

7.4 A PTAS for any β-distributed Euclidean k-Means Instance

7.4.1 Intuition

Analogous to the k-median algorithm, we present an essentially identical algorithm for k-means in Euclidean space. Indeed, the fact that k-means considers distances squared, makes upper (or lower) bounding distances a bit more complicated, and requires that we fiddle with the parameters of the algorithm. In addition, the centers c_i^* may not be data points. However, the overall approach remains the same. Roughly speaking, converting the k-median algorithm to the k-means case, we use the same constants, only squared. As before we handle expensive clusters by guessing good substitutes for their centers and obtain *good* components for cheap clusters.

Often, when considering the Euclidean space k-means problem, the dimension of the space plays an important factor. In contrast, here we make no assumptions about the dimension, and our results hold for any $\mathrm{poly}(n)$ dimension. In fact, for ease of exposition,

we assume all distances between any two points were computed in advance and are given to our algorithm. Clearly, this only adds $O(n^2 \cdot \mathsf{dim})$ to our runtime. In addition to the change in parameters, we utilize the following facts that hold for the center of mass in Euclidean space.

Fact 7.13. *Let U be a (finite) set of points in an Euclidean space, and let μ_U denote their center of mass ($\mu = \frac{1}{|U|}\sum_{x\in U} x$). Let A be a random subset of U, and denote by μ_A the center of mass of A. Then for any $\delta < 1/2$, we have both*

$$\mathbf{Pr}\left[\|\mu_U - \mu_A\|^2 > \frac{1}{\delta|A|}\cdot\frac{1}{|U|}\sum_{x\in U}\|x-\mu_U\|^2\right] < \delta \tag{7.1}$$

$$\mathbf{Pr}\left[\sum_{x\in U}\|x-\mu_A\|^2 > (1+\frac{1}{\delta|A|})\cdot\sum_{x\in U}\|x-\mu_U\|^2\right] < \delta \tag{7.2}$$

Fact 7.14. *Let U be a (finite) set of points in an Euclidean space, and let $A \neq \emptyset$ and B be a partition of U. Denote by μ_U and μ_A the center of mass of U and A resp. Then $\|\mu_U - \mu_A\|^2 \leq \frac{1}{|U|}\sum_{x\in U}\|x-\mu_U\|^2 \cdot \frac{|B|}{|A|}$.*

Fact 7.14, proven in [121] (Lemma 2.2), allows us to upper bound the distance between the real center of a cluster and the empirical center we get by averaging all points in $T \cup B(T)$ for a good component T. Fact 7.13 allows us to handle expensive clusters. Since we cannot brute force guess a center (as the center of the clusters aren't necessarily data points), we guess a sample of $O(\beta^{-1}+\epsilon^{-1})$ points from every expensive cluster, and use their average as a center. Both properties of Fact 7.13, proven in [88] (§3, Lemma 1 and 2), assure us that the center is an adequate substitute for the real center and is also close to it. This motivates the approach behind our first algorithm, in which we brute-force traverse all choices of $O(\epsilon^{-1}+\beta^{-1})$ points for any of the expensive clusters.

The second algorithm, whose runtime is $(k\log n)^{\mathrm{poly}(1/\epsilon,1/\beta)}O(n^3)$, replaces brute-force guessing with random sampling. Indeed, if a cluster contains $\mathrm{poly}(1/k)$ fraction of the points, then by randomly sampling $O(\epsilon^{-1}+\beta^{-1})$ points, the probability that all points belong to the same expensive cluster, and furthermore, their average can serve as a good empirical center, is at least $1/k^{\mathrm{poly}(1/\epsilon,1/\beta)}$. In contrast, if we have expensive clusters that contain few points (e.g. an expensive cluster of size $\sqrt{n}$, while $k = \mathrm{poly}(\log(n))$), then random sampling is unlikely to find good empirical centers for them. However, recall that our algorithm collects points and deletes them from our instance. So, it is possible that in the middle of the run, we are left with so few points, so that expensive clusters whose size is small in comparison to the original number of points, contain a $\mathrm{poly}(1/k)$ fraction of the *remaining* points.

Indeed, this is the motivation behind our second algorithm. We run the algorithm while interleaving the Population Stage of the algorithm with random sampling. Instead of running s from n to 1, we use $\left\{n, \frac{n}{k^2}, \frac{n}{k^4}, \frac{n}{k^6}, \ldots, 1\right\}$ as break points. Correspondingly, we define l_i to be the number of expensive clusters whose size is in the range $[n \cdot k^{-2i-2}, n \cdot k^{-2i})$. Whenever s reaches such a $n \cdot k^{-2i}$ break point, we randomly sample points in order to guess the l_{i+3} centers of the clusters that lie 3 intervals "ahead" (and so, initially, we guess all centers in the first 3 intervals). We prove that in every interval we are likely to sample good empirical centers. This is a simple corollary of Fact 7.14 along with the following two claims. First, we claim that at the end of each interval, the number of points remaining is at most $n \cdot k^{-2i+1}$. Secondly, we also claim that in each interval we do not remove even a single point from a cluster whose size is smaller than $n \cdot k^{-2i-6}$.

7.4.2 A Deterministic Algorithm for β-distributed k-Means Instances

Our algorithm is presented in Figure 7.2. The correctness is proved in a similar fashion to the proof of correctness presented in Section 7.3. Much like in Section 7.3, we call a cluster in the optimal k-means solution cheap if $\mathsf{OPT}_i = \sum_{x \in C_i^*} d^2(x, c_i^*) \leq \frac{\beta\epsilon\mathsf{OPT}}{4^6}$. First, observe that by the Markov inequality, for any cheap cluster C_i^*, we have that the set $\left\{x;\ d^2(x, c_i^*) > t\frac{\beta\mathsf{OPT}}{|C_i^*|}\right\}$ cannot contain more than $\epsilon/(4^6 t)$ fraction of the points in $|C_i^*|$. It follows that the inner ring of C_i^*, the set $\left\{x;\ d^2(x, c_i^*) \leq \frac{\beta\mathsf{OPT}}{256|C_i^*|}\right\}$, contains at least half of the points of C_i^*. The algorithm populates the list $\mathcal{Q}$ with *good* components corresponding to cheap clusters. Also from Fact 7.13, we know that for every expensive cluster, there exists a sample of $O(\frac{1}{\beta} + \frac{1}{\epsilon})$ data points whose center is a good substitute for the center of the cluster. In the analysis below, we assume that $\mathcal{Q}$ has been initialized correctly with $\mathcal{Q}_{init}$ containing these good substitutes. In general, the algorithm will be run multiple times for all possible guesses of samples from expensive clusters. We start with the following lemma which is similar to Lemma 7.8.

Lemma 7.15. *Let $T \in \mathcal{Q}$ be any component and let s be the stage in which we insert T to $\mathcal{Q}$. Let C_i^* be any cheap cluster s.t. $s \geq |C_i^*|$. Then (a) T does not contain any point z s.t. the distance $d^2(c_i^*, z)$ lies within the range $\left[\frac{\beta}{16}\frac{\mathsf{OPT}}{|C_i^*|}, \frac{\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}\right]$, and (b) T cannot contain both a point p_1 s.t. $d^2(c_i^*, p_1) \leq \frac{\beta}{16}\frac{\mathsf{OPT}}{|C_i^*|}$ and a point p_2 s.t. $d^2(c_i^*, p_2) > \frac{\beta}{4}\frac{\mathsf{OPT}}{|C_i^*|}$.*

Proof. Assume (a) does not hold. Let z be such point, and let $B(z, r)$ be the set of all points p s.t. $d^2(z, p) \leq r = \frac{\beta\mathsf{OPT}}{64s} \leq \frac{\beta\mathsf{OPT}}{64|C_i^*|}$. As $d^2(z, c_i^*) \geq \frac{\beta\mathsf{OPT}}{16|C_i^*|}$, we have that $d(z, p) \leq \frac{1}{2}d(z, c_i^*)$. It follows that $d^2(c_i^*, p) \geq \left(d(c_i^*, z) - d(z, p)\right)^2 \geq \left(d(c_i^*, z)/2\right)^2 = \frac{\beta\mathsf{OPT}}{64|C_i^*|}$. Sim-

1. **Initialization Stage:** Set $\mathcal{Q} \leftarrow \mathcal{Q}_{init}$.

2. **Population Stage:** For $s = n, n-1, n-2, \ldots, 1$ do:

   (a) Set $r = \frac{\beta \mathsf{OPT}}{64s}$.

   (b) Remove any point x such that $d^2(x, \mathcal{Q}) < 4r$.
   (Here, $d(x, \mathcal{Q}) = \min_{T \in \mathcal{Q}; y \in T} d(x, y)$.)

   (c) For any remaining data point x, denote the set of data points whose distance squared from x is at most r, by $B(x, r)$. Connect any two remaining points a and b if:
   (i) $d^2(a, b) \leq r$, (ii) $|B(a, r)| > \frac{s}{2}$ and (iii) $|B(b, r)| > \frac{s}{2}$.

   (d) Let T be a connected component of size $> \frac{s}{2}$. Then:

      i. Add T to $\mathcal{Q}$. (That is, $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{T\}$.)

      ii. Define the set $B(T) = \{x : \ d^2(x, y) \leq 4r \text{ for some } y \in T\}$. Remove the points of $B(T)$ from the instance.

3. **Centers-Retrieving Stage:** For any choice of k components $T_1, T_2, \ldots, T_k$ out of $\mathcal{Q}$

   (a) Find the best center c_i for $T_i \cup B(T_i)$.
   That is $c_i = \mu(T_i \cup B(T_i)) = \frac{1}{|T_i \cup B(T_i)|} \sum_{x \in T_i \cup B(T_i)} x$.

   (b) Partition all n points according to the nearest point among the k centers of the current k components.

   (c) If a clustering of cost at most $(1+\epsilon)\mathsf{OPT}$ is found – output these k centers and halt.

Figure 7.2: A deterministic PTAS for β-distributed instances of Euclidean k-means.

ilarly, $d^2(c_i^*, p) \leq (d(c_i^*, z) + d(z, p))^2 \leq (3d(c_i^*, z)/2)^2 \leq \frac{9\beta}{16} \frac{\mathsf{OPT}}{|C_i^*|}$. Thus $B(z, r)$ is contained in C_i^*, but falls outside the inner-ring of C_i^*, yet contains $s/2 \geq |C_i^*|/2$ many points. Contradiction.

Assume (b) does not hold. Let p_1 and p_2 the above mentioned points. As T is a connected components, it follows that along the path $p_1 \to p_2$, exists a pairs of neighboring nodes, x, y, s.t. $d^2(x, y) \leq r \leq \frac{\beta \mathsf{OPT}}{64|C_i^*|}$ yet $d^2(c_i^*, x) \leq \frac{\beta}{16} \frac{\mathsf{OPT}}{|C_i^*|}$ while $d^2(c_i^*, y) \geq \frac{\beta}{4} \frac{\mathsf{OPT}}{|C_i^*|}$. However, a simple computation gives that $d^2(c_i^*, y) \leq (3d(c_i^*, x)/2)^2 \leq \frac{9\beta}{64} \frac{\mathsf{OPT}}{|C_i^*|}$. Contradiction. □

Lemma 7.15 allows us to give the analogous claims to Claims 7.9 and 7.10. As before, call a component T *good* if it is contained within some target cluster C_i^* *and* $T \cup B(T)$ contains all of the inner ring points of C_i^*. Otherwise, the component is called *bad* provided it is not one of the initial centers present in $\mathcal{Q}_{init}$. We now show that each cheap target cluster will have a single, unique, good component.

Claim 7.16. *Let C_i^* be any cheap cluster in the target clustering. By stage $s = |C_i^*|$, the algorithm adds to $\mathcal{Q}$ a component T that contains a point from the inner ring of C_i^*.*

Claim 7.17. *Let T be a good connected component added to $\mathcal{Q}$, containing an inner ring point from cluster C_i^*. Then: (a) all points in T are of distance squared at most $\frac{\beta \mathsf{OPT}}{16|C_i^*|}$ from c_i^*, (b) $T \cup B(T)$ is fully contained in C_i^*, and (c) the entire inner ring of C_i^* is contained in $T \cup B(T)$, and (d) no other component $T' \neq T$ in $\mathcal{Q}$ contains an inner ring point from C_i^*.*

As the proofs of Claims 7.16 and 7.17 are identical to the Claims 7.9 and 7.10, we omit them.

Lemma 7.18. *We do not add to $\mathcal{Q}$ more than $1000/\beta$ bad components.*

Proof. Consider any bad component T that we add to $\mathcal{Q}$ and denote that stage in which we insert T to $\mathcal{Q}$ as s. So the size of this component is $> \frac{s}{2}$. Let y be an arbitrary point from T which belongs to cluster C^* in the optimal clustering. Let c^* be the center of C^*. We show that $d^2(c^*, y) > \frac{\beta \mathsf{OPT}}{500s}$.

We divide into cases.

Case 1: C^* is a cheap cluster and $s \geq |C^*|$. Recall that T must contain $s/2 \geq |C^*|/2$ points, so it follows that T contains some point x that does not belong to C^*. β-stability gives that this point has distance $d^2(c^*, x) > \beta \frac{\mathsf{OPT}}{|C^*|}$, and we apply Lemma 7.15 to deduce that all points in T are of distance squared of at least $\frac{\beta}{4} \frac{\mathsf{OPT}}{|C^*|}$.

Case 2: C^* is a cheap cluster and $s < |C^*|$. In this case we have that the entire inner ring of C^* already belongs to some $T' \in Q$. Let $x \in T'$ be any inner ring point from C^*, and we have that $d(c^*, x)^2 \leq \frac{\beta\mathsf{OPT}}{256|C^*|} \leq \frac{\beta\mathsf{OPT}}{256s}$, while $d^2(x, y) > \frac{\beta\mathsf{OPT}}{16s}$. It follows that $d^2(c^*, y) \geq (3d(x, y)/4)^2 > \frac{\beta\mathsf{OPT}}{500s}$.

Case 3: C^* is an expensive cluster and $s > 2|C^*|$. We claim that $d^2(c^*, y) > \frac{\beta\mathsf{OPT}}{32|C^*|}$. If, by contradiction, we have that $d^2(c^*, y) \leq \frac{\beta\mathsf{OPT}}{32|C^*|}$, then we show that the ball $B(y, r)$ contains only points from C_i^*, yet it must contains $s/2 > |C_i^*|$ points. This is because each $p \in B(y, r)$ satisfies that $d^2(c^*, p) \leq (d(c^*, y) + d(y, p))^2 \leq \left(\sqrt{\frac{\beta\mathsf{OPT}}{32|C^*|}} + \sqrt{\frac{\beta\mathsf{OPT}}{16s}}\right)^2 < \frac{\beta\mathsf{OPT}}{|C^*|}$.

Case 4: C^* is an expensive cluster and $s \leq 2|C^*|$. In this case, from Fact 7.13 we know that $\mathcal{Q}_{init}$ contains a a good empirical center c for the expensive cluster C^*, in the sense that $\|c - c^*\|^2 \leq \frac{\beta\mathsf{OPT}}{512|C^*|} \leq \frac{\beta\mathsf{OPT}}{256s}$. Then, similarly case 2 above we have $d^2(y, c^*) \geq (d(y, c) - d(c, c^*))^2 > \frac{\beta\mathsf{OPT}}{500s}$. It follows that every point in T has a large distance from its center. Therefore, the $s/2$ points in this component contribute at least $\beta\mathsf{OPT}/1000$ to the k-means cost. Hence, we can have no more than $1000/\beta$ such bad components. □

We now prove the main theorem.

Theorem 7.19. *The algorithm outputs a k-clustering whose cost is at most $(1 + \epsilon)\mathsf{OPT}$.*

Proof. Using Claim 7.17, it follows that there exists some choice of k components which has good components for all the cheap clusters and good substitutes for the centers of the expensive clusters. Fix that choice and consider a cluster C_i^* with center c_i^*. If C_i^* is an expensive cluster then from Section 7.4 we know that $\mathcal{Q}_{init}$ contains a point c_i such that $d^2(c_i, c_i^*) \leq \frac{\beta\epsilon}{\beta+\epsilon}\frac{\mathsf{OPT}_i}{|C_i^*|}$. Hence, the cost paid by the points in C_i^* will be atmost $(1+\epsilon)\mathsf{OPT}_i$. If C_i^* is a cheap cluster then denote by T the good component that resides within C_i^*. Denote $T \cup B(T)$ by A, and $C_i^* \setminus A$ by B. Let c_i be the center of A. We know that the entire inner-ring of C_i^* is contained in A, therefore, B cannot contain more than $\epsilon/16$ fraction of the points of C_i^*. Fact 7.14 dictates that in this case, $\|c_i^* - c_i\|^2 \leq \epsilon^2 \frac{\beta\mathsf{OPT}}{4^6|C_i^*|}$. We know every $x \in B$ contributes at least $\frac{\beta\mathsf{OPT}}{256|C_i^*|}$ to the cost of C_i^*, so $\|c_i^* - c_i\|^2 \leq \frac{\epsilon}{16}\|x - c_i^*\|^2$. Thus, for every $x \in B$, we have that $\|x - c_i\|^2 \leq (1+\epsilon)\|x - c_i^*\|^2$. It follows that $\sum_{x\in B}\|x - c_i\|^2 \leq (1 + \epsilon)\sum_{x\in B}\|x - c_i^*\|^2$, and obviously $\sum_{x\in A}\|x - c_i\|^2 \leq \sum_{x\in A}\|x - c_i^*\|^2$ as c_i is the center of mass of A. Therefore, when choosing the good k components out of $\mathcal{Q}$, we can assign them to the centers in such a way that costs no more than $(1+\epsilon)\mathsf{OPT}$. Obviously the assignment of each point to the nearest of the k-centers only yields a less costly clustering, and thus its cost is also at most $(1 + \epsilon)\mathsf{OPT}$. □

7.4.3 A Randomized Algorithm for β-distributed k-Means Instances

We now present a randomized algorithm which achieves a $(1+\epsilon)$ approximation to the k-means optimum of a β-distributed instance and runs in time $(k \log_k n)^{\mathrm{poly}(1/\epsilon, 1/\beta)} O(n^3)$. The algorithm is similar in nature to the one presented in the previous section, except that for expensive clusters we replace brute force guessing of samples with random sampling. Note that the straightforward approach of sampling the points right at the start of the algorithm might fail, if there exist expensive clusters which contain very few points. A better approach is to interleave the sampling step with the rest of the algorithm. In this way we sample points from an expensive cluster only when it contains a reasonable fraction of the total points remaining, hence our probability of success is noticeable (namely, $\mathrm{poly}(1/k)$). The alterations required in making the previous algorithm into a randomized one that also samples cluster centers are detailed in Figure 7.3.

The high-level approach of the algorithm is to partition the main loop of the Population Stage, in which we try all possible values of s (starting from n and ending at 1), into *intervals*. In interval i we run s on all values starting with $\frac{n}{k^{2i}}$ and ending with $\frac{n}{k^{2i+2}}$. So overall, we have no more than $t = \frac{1}{2}\log_k(n)$ intervals. Our algorithm begins by guessing l, the number of expensive clusters, then guessing $g_1, g_2, \ldots, g_t$ s.t. $\sum_i g_i = l$. Each g_i is a guess for the number of expensive clusters whose size lies in the range $\left[\frac{n}{k^{2i}}, \frac{n}{k^{2(i-1)}}\right)$. Note that $\sum_i g_i =$ # expensive clusters $\leq \frac{4^6}{\beta\epsilon}$. Hence, there are at most $(\log_k n)^{\frac{4^6}{\beta\epsilon}}$ number of possible assignments to g_i's and we run the algorithm for every such possible guess.

Fixing $g_1, g_2, \ldots, g_t$, we run the Population Stage of the previous algorithm. However, whenever s reaches a new interval, we apply random sampling to obtain good empirical centers for the expensive clusters whose size lies *three intervals "ahead"*. That is, in the beginning of interval i, the algorithm tries to collect centers for the clusters whose size $\geq \frac{n}{k^{6+2i}} = \frac{s}{k^6}$, yet $\leq \frac{n}{k^{4+2i}} = \frac{s}{k^4}$. We assume for this algorithm that k is significantly greater than $\frac{1}{\beta}$. Obviously, if k is a constant, then we can use the existing algorithm of Kumar et al [105].

In order to prove the correctness of the new algorithm, we need to show that the sampling step in the initialization stage succeeds with noticeable probability. Let l_i be the actual number of expensive clusters whose size belongs to the range $\left[\frac{n}{k^{2i}}, \frac{n}{k^{2(i-1)}}\right)$. In the proof which follows, we assume that the correct guess for l_i's has been made, i.e. $g_i = l_i$, for every i. We say that the algorithm *succeeds at the end of interval* i if the following conditions hold:

1. In the beginning of the interval, our guess for all clusters that belong to interval $(i+3)$ produces good empirical centers. That is, for every expensive cluster C^* of

1. Guess $l \leq \frac{4^6}{\beta\epsilon}$, the number of expensive clusters. Set $t = \frac{1}{2}(\log_k n)$. Guess non-negative integers $g_1, g_2, \ldots g_t$, such that $\sum_i g_i = l$.
2. Sample $g_1 + g_2 + g_3$ sets, by sampling independently and u.a.r $O(\frac{1}{\beta} + \frac{1}{\epsilon})$ points for each set. For each such set $\tilde{T}_j$, add the singleton $\{\mu(\tilde{T}_j)\}$ to $\mathcal{Q}$.
3. Modify the Population Stage from the previous algorithm, so that whenever $s = \frac{n}{k^{2i}}$ for some $i \geq 1$ (We call this the *interval* i)
   - Sample g_{i+3} sets, by sampling independently and u.a.r $O(\frac{1}{\beta} + \frac{1}{\epsilon})$ points for each set. For each such set $\tilde{T}_j$, add the singleton $\{\mu(\tilde{T}_j)\}$ to $\mathcal{Q}$.
4. Proceed with the Centers-Retrieving Stage as before.

Figure 7.3: A Randomized PTAS for β-distributed instances of Euclidean k-means, that succeeds w.p. $k^{-O(\frac{1}{\beta}+\frac{1}{\epsilon})}$.

size in the range $\left[\frac{n}{k^{6+2i}}, \frac{n}{k^{4+2i}}\right)$, the algorithm picks a sample $\tilde{T}$ such that the mean $\mu(\tilde{T})$ satisfies:

(a) $d^2(\mu(\tilde{T}), c^*) \leq \frac{\beta \mathsf{OPT}}{256|C^*|}$.

(b) $\sum_{x \in C^*} d^2(x, \mu(\tilde{T})) \leq (1+\epsilon) \sum_{x \in C^*} d^2(x, c^*)$.

2. During the interval, we do not delete any point p that belongs to some target cluster C^* of size $\leq \frac{n}{k^{4+2(i+1)}}$ points.

3. At the end of the interval, the total number of remaining points (points that were not added to some $T \in \mathcal{Q}$ or deleted from the instance because they are too close to some $T' \in \mathcal{Q}$) is at most $\frac{n}{k^{2i-1}}$.

Lemma 7.20. *For every $i \geq 1$, let S_i denote the event that the algorithm succeeds at the end of interval i. Then* $\mathbf{Pr}[S_i | S_1, S_2, \ldots, S_{i-1}] \geq k^{-l_{(i+3)} \cdot O(\frac{1}{\beta}+\frac{1}{\epsilon})}$

Before going into the proof we show that Lemma 7.20 implies that with noticeable probability, our algorithm returns a $(1+\epsilon)$-approximation of the k-means optimal clustering. First, observe the technical fact that for the first three intervals l_1, l_2, l_3, we need

to guess the centers of clusters of size $\geq \frac{n}{k^6}$ before we start our Population Stage. However, as these clusters contain k^{-6} fraction of the points, then using Fact 7.13, our sampling finds good empirical centers for all of these $l_1 + l_2 + l_3$ expensive clusters w.p. $\geq k^{-(l_1+l_2+l_3)O(\frac{1}{\beta}+\frac{1}{\epsilon})}$. Applying Lemma 7.20 we get that the probability our algorithm succeeds after all intervals is $\geq 1/k^{O(\frac{\beta+\epsilon}{\beta^2\epsilon^2})}$. Now, a similar analysis as in the previous section gives us that for the correct guess of the good components in $\mathcal{Q}$, we find a clustering of cost at most $(1+\epsilon)\mathsf{OPT}$.

Proof of Lemma 7.20. Recall that β is a constant, whereas k is not. Specifically, we assume throughout the proof that $k^2 > \frac{500}{\beta}$, and so we allow ourselves to use asymptotic notation.

We first prove that condition 2 holds during interval i. Assume for the sake of contradiction that for some cluster C^* whose size is less than $\frac{n}{k^{6+2i}}$, there exists some point $y \in C^*$, which was added to some component T during interval i, at some stage $s \in \left[\frac{n}{k^{2i+2}}, \frac{n}{k^{2i}}\right)$. This means that by setting the radius $r = \frac{\beta\mathsf{OPT}}{64s}$, the ball $B(y,r)$ contains $> s/2 \geq \frac{n}{2k^{2i+2}}$ points. Since C^* contains at most $\frac{n}{k^{6+2i}}$ many point, we have $|C^*| \ll s/2$, so at least $s/4$ points in $B(y,r)$ belong to other clusters. Our goal is to show that these $s/4$ points contribute more than OPT to the target clustering, thereby achieving a contradiction.

Let x be such point, and denote the cluster that x is assigned to in the target clustering by $C^*_j \neq C^*$. Since the instance is β-distributed we have that $d^2(c^*,x) > \frac{\beta\mathsf{OPT}}{|C^*|} \geq \beta\mathsf{OPT}\frac{k^{6+2i}}{n}$. On the other hand, $d^2(x,y) \leq r = \beta\mathsf{OPT}\frac{1}{64s} \leq \beta\mathsf{OPT}\frac{k^{2+2i}}{64n}$. Therefore, $d^2(c^*,x) = \Omega(k^4)\cdot r$, so $d^2(y,c^*) = \left(d(c^*,x) - d(x,y)\right)^2 = \Omega(k^4)\cdot r$. Recall that in the target clustering each point is assigned to its nearest center, so $d^2(c^*_j,y) \geq d^2(c^*,y) = \Omega(k^4)\cdot r$. So we have that $d^2(c^*_j,x) \geq \left(d(c^*_j,y) - d(x,y)\right)^2 = \Omega(k^4)\cdot r = \Omega(k^4)\cdot\frac{\beta\mathsf{OPT}}{64}\frac{k^{2i}}{n}$.

So, at least $s/4 = \Omega(\frac{n}{k^{2i+2}})$ points contribute $\Omega(k^4)\frac{\beta\mathsf{OPT}}{64}\frac{k^{2i}}{n}$ to the cost of the optimal clustering. Their total contribution is therefore $\Omega(k^2)\cdot\frac{\beta}{64}\mathsf{OPT} > \mathsf{OPT}$. Contradiction.

A similar proof gives that no point $y \in C^*$ is deleted from the instance because for some $x \in T$, where T is some component in $\mathcal{Q}$, we have that $d^2(y,x) < 4r$. Again, assume for the sake of contradiction that such y,x and T exist. Denote by $s \in \left[\frac{n}{k^{2i+2}}, \frac{n}{k^{2i}}\right)$ the stage in which we remove y, and denote by $s' \geq s$ the stage in which we insert T into $\mathcal{Q}$. By setting the radius $r' = \frac{\beta\mathsf{OPT}}{64s'} \leq r$, we have that the ball $B(x,r')$ contains at least $s'/2 \geq s/2$ points, and therefore, the ball $B(y,5r)$ contains at least $s/2$ points. We now continue as in the previous case.

We now prove condition 1. We assume the algorithm succeeded in all previous inter-

vals. Therefore, at the beginning of interval i, all points that belong to clusters of size $\leq \frac{n}{k^{2i+4}}$ remain in the instance, and in particular, the clusters we wish to sample from at interval i remain intact. Furthermore, by the assumption that the algorithm succeeded up to interval $(i-1)$, we have that each expensive cluster that should be sampled at the beginning of interval i, contains a $1/k^7$ fraction of the remaining points. We deduce that the probability that we pick a random sample of $O(\frac{1}{\beta}+\frac{1}{\epsilon})$ points from such expensive cluster is at least $k^{-O(\frac{1}{\beta}+\frac{1}{\epsilon})}$. Using Fact 7.13 we have that with probability $\geq k^{-O(\frac{1}{\beta}+\frac{1}{\epsilon})}$ this sample yields a good empirical center.

We now prove condition 3, under the assumption that 1 is satisfied. We need to bound the number of points left in the instance at the end of interval i. There are two types of remaining points: points that in the target clustering belong to clusters of size $> \frac{n}{2k^{2i}}$, and points that belong to clusters of size $\leq \frac{n}{2k^{2i}}$. To bound the number of points of the second type is simple – we have k clusters, so the overall number of points of the second type is at most $\frac{n}{2k^{2i-1}}$. We now bound the number of remaining points of the first type.

At the end of the interval $s = \frac{n}{k^{2i+2}}$, so we remove from the instance any point p whose distance (squared) from some point in $\mathcal{Q}$ is at most $4r = \frac{\mathsf{OPT}}{16}\frac{k^{2i+2}}{n}$. We already know that by the end of interval i, either by successfully sampling an empirical center or by adding an inner-ring point to a component in $\mathcal{Q}$, for every cluster C^* of size $> \frac{n}{2k^{2i}}$, exists some $T \in \mathcal{Q}$ with a point $c' \in T$, s.t. $d^2(c^*, c') \leq \frac{\beta\mathsf{OPT}}{256|C^*|} \leq \frac{\beta\mathsf{OPT}}{128}\frac{k^{2i}}{n}$. Thus, if $x \in C^*$ is a point that wasn't removed from the instance by the end of interval i, it must hold that $d^2(c^*, x) \geq \left(d(c', x) - d(c^*, c')\right)^2 = \Omega(k^{2i+2})\frac{\mathsf{OPT}}{n}$. Clearly, at most $n \cdot O(k^{-2i-2})$ points can contribute that much to the cost of the optimal k-means clustering, and so the number of points of the first type is at most $\frac{n}{2k^{2i-1}}$ as well. □

As we need to traverse all guesses g_is, the runtime of this algorithm takes $O(n^3(\log_k n)^{O(\frac{1}{\beta\epsilon})})$. Repeating this algorithm $k^{O(l(\frac{1}{\beta}+\frac{1}{\epsilon}))}$ many times, we increase the probability of success to be $\geq 1/2$, and incur runtime of $O(n^3(\log_k n)^{O(\frac{1}{\beta\epsilon})}k^{O(\frac{\beta+\epsilon}{\beta^2\epsilon^2})})$.

7.5 Discussion and Open Problems

The algorithm we present here for k-median has runtime of $\mathrm{poly}(n^{1/\beta}, n^{1/\epsilon}, k)$, and the algorithm for k-means has runtime $\mathrm{poly}(n, (k\log n)^{1/\epsilon}, (k\log n)^{1/\beta})$.[3] We comment that it is unlikely that we can obtain an algorithm of runtime $\mathrm{poly}(n^{1/\epsilon}, 1/\beta, k)$. Observe that for

[3]When dealing with k-means in a Euclidean space of dimension dim, we need to explicitly compute the distances, so we add $n^2\mathsf{dim}$ to the runtime.

any clustering instance and any $k > 1$ we have that $\frac{\mathsf{OPT}_{(k-1)}}{\mathsf{OPT}} > 1+\frac{1}{n}$, simply by considering the k-clustering that results from taking the optimal $(k-1)$-clustering, and setting the point which is the furthest from its center in a cluster of its own (as a new center). Hence, any k-median/k-means instance is β-distributed for $\beta = \Omega(\frac{1}{n})$. Recall from Section 7.2.4 the k-median problem restricted only to weakly-stable instances has no FPTAS. So the fact that our algorithm's runtime has super-polynomial dependence in *both* $1/\beta$ and $1/\epsilon$ is unavoidable. Nonetheless, one might still hope to do better. In particular, one major runtime expense of our algorithm comes from handling expensive clusters by brute-force guessing or sampling. Can one improve the runtime by doing something more clever for expensive clusters? It is worth noting that for the stability conditions of [25], Voevodski et al [140] develop an especially efficient implementation with good performance (in terms of both accuracy and speed) on real-world protein sequence datasets.

A different open problem lies in the relation to results of Ostrovsky et al [121]. Their motivating question was to analyze the performance of Lloyd-type methods over stable instances. Is it possible that weak deletion-stability is sufficient for some version of the k-means heuristic to converge to the optimal clustering?

Chapter 8

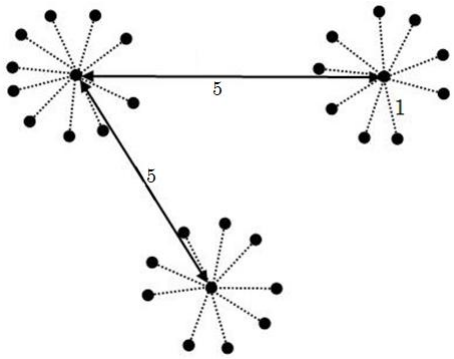
Center-based Clustering under Perturbation Stability

8.1 Introduction

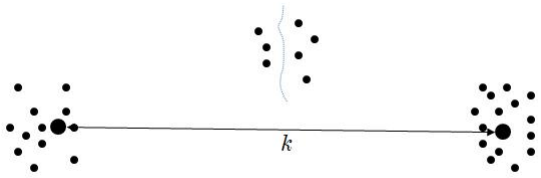
In the vast field of clustering, the recent work of Bilu and Linial [40] takes a refreshing approach to clustering. Bilu and Linial [40], focusing on the Max-Cut problem [71], proposed considering instances where the optimal clustering is optimal not only under the given metric, but also under any bounded multiplicative perturbation of the given metric. Bilu and Linial [40] analyze Max-Cut instances of this type and show that for instances that are stable to perturbations of multiplicative factor roughly $O(n^{1/2})$, one can retrieve the optimal Max-Cut in polynomial time. They conjecture that stability up to only *constant* magnitude perturbations should be enough to solve the problem in polynomial time. (Center-based clustering and the recent approach proposed by Bilu and Linial [40] were discussed in detail in Chapter 4.2.3.)

In this chapter we show that this conjecture is indeed true for k-median and k-means objectives and in fact for any well-behaved center-based objective function (see Definition 4.2). We comment that in the previous chapter, Chapter 7, we studied instances satisfying "weak deletion stability" – where merging any two clusters in the optimal k-solution increases the cost by a noticeable factor. This notion is motivated both as a relaxation of the separation condition of Ostrovsky et al [121], and also as a relaxation of the notion considered by Balcan et al [25]. However, there exists "clear-cut" instances, where the optimal k-clustering is obvious, which weak-deletion stability fails to capture. The notion of stability studied in this chapter, perturbation resilience, indeed capture such instances

(but fails to capture instances satisfying the weak-deletion stability notion of Chapter 7). Example is given in Figure 8.1.



(a) *Instance satisfying perturbation resilience but isn't weak-deletion stable*. The distance of any point to its cluster center is 1 and distances between two different centers is 5 and remaining distances are short-path distances. With k clusters, each with $\frac{n}{k}$ points, merging two clusters increases the cost by a sub-constant factor of $O(1/k)$.



(b) *Instance satisfying weak-deletion stability but isn't perturbation resilient*. The cluster centers are within distance k apart, but there are many "middle" points whose distance to both centers is roughly $k/2$. Perturbing the distances can cause the middle points to change clusters.

Figure 8.1: Instances satisfying one notion of stability but no the other.

8.1.1 Main Result

For clarity, let us formally redefine the notions of stability under multiplicative perturbations.

Definition 8.1. *Given a metric (S, d), and $\alpha > 1$, we say a function $d' : S \times S \to \mathbb{R}_{\geq 0}$ is an* α-perturbation *of d, if for any $x, y \in S$ it holds that*

$$d(x, y) \leq d'(x, y) \leq \alpha d(x, y)$$

Note that in this definition, much like in the definition of [40], d is a metric (satisfying reflexivity, symmetry and triangle inequality), yet d' may be any non-negative function. In particular, we allow d' to *not* satisfy the triangle inequality. We now give our main definitions and main theorem.

Definition 8.2. *Suppose we have a clustering instance composed of n points residing in a metric (S, d) and an objective function Φ we wish to optimize. We call the clustering instance* α-perturbation resilient *for Φ if for any d' which is an α-perturbation of d, the (only) optimal clustering of (S, d') under Φ is identical, as a partition of points into subsets, to the optimal clustering of (S, d) under Φ.*

We will in particular be concerned with *separable*, *center-based* clustering objectives Φ (which include k-median, k-means, and k-center among others).

Definition 8.3. *A clustering objective is* center-based *if the optimal solution can be defined by k points $c_1^*, \ldots, c_k^*$ in the metric space called* centers *such that every data point is assigned to its nearest center. Such a clustering objective is* separable *if it furthermore satisfies the following two conditions:*

- *The objective function value of a given clustering is either a (weighted) sum or the maximum of the individual cluster scores.*
- *Given a proposed single cluster, its score can be computed in polynomial time.*

Our main result is that we can efficiently find the optimal clustering for perturbation-resilient instances of separable center-based clustering objectives. In particular, we get an efficient algorithm for 3-perturbation-resilient instances when the metric S is defined only over data points, and for $(2 + \sqrt{3})$-perturbation-resiliant instances for general metrics.

Theorem 8.4. *For $\alpha \geq 3$ (in the case of finite metrics defined only over the data) or $\alpha \geq 2 + \sqrt{3}$ (for general metrics), there is a polynomial-time algorithm that finds the optimal clustering of α-perturbation resilient instances for any given separable center-based clustering objective.*

The algorithm, described in Section 8.2.2, turns out to be quite simple. As a first step, it runs the classic single-linkage algorithm, but unlike the standard approach of halting when k clusters remain, it runs the algorithm until *all* points have been merged into a single cluster and keeps track of the entire tree-on-clusters produced.[1] Then, the algorithm's second step is to apply dynamic programming to this hierarchical clustering to identify the best k-clustering that is present within the tree. Using a result of Balcan et al [27] we show that the resulting clustering obtained is indeed the optimal one. Albeit being very different, our approach resembles, in spirit, the work of Bartal [32], Abraham et al [2] and

[1]The example depicted in Figure 8.4 proves that indeed, halting the Single-Linkage algorithm once k clusters are formed may fail on certain α-perturbation resilient instances.

Räcke [124] in the sense that we reduce the problem of retrieving an optimal solution from a general instance to a tree-like instance (where it is poly-time solvable).

Our algorithms use only a weaker property, which we call center-proximity (see Section 8.2.1), that is implied by perturbation-resilience. We then complement these results with a lower bound showing that for the problem of k-median on general metrics, for any $\epsilon > 0$, there exist NP-hard instances that satisfy $(3-\epsilon)$-center proximity. We note that while our belief was that allowing Steiner points in the lower bound was primarily a technicality, Balcan and Liang [29] have recently shown this is not the case, giving a clever algorithm that finds the optimal clustering for k-median instances in finite metrics when $\alpha = 1+\sqrt{2}$, and Reyzin [127] gave a NP-hardness result for clustering instances satisfying $(2-\epsilon)$-center proximity.

8.2 Proof of Main Theorem

8.2.1 Properties of Perturbation Resilient Instances

We begin by deriving other properties which every 3-perturbation resilient clustering instance must satisfy.

Definition 8.5. *Let $p \in S$ be an arbitrary point, let c_i^* be the center p is assigned to in the optimal clustering, and let $c_j^* \neq c_i^*$ be any other center in the optimal clustering. We say a clustering instance satisfies the* α-center proximity *property if for any p it holds that*

$$d(p, c_j^*) > \alpha d(p, c_i^*)$$

Fact 8.6. *If a clustering instance satisfies the α-perturbation resilience property, then it also satisfies the α-center proximity property.*

Proof. Let C_i^* and C_j^* be any two clusters in the optimal clustering and pick any $p \in C_i^*$. Assume we blow up all the pairwise distances within cluster C_i^* by a factor of α. As this is a legitimate perturbation of the metric, it still holds that the optimal clustering under this perturbation is the same as the original optimum. Hence, p is still assigned to the same cluster. Furthermore, since the distances within C_i^* were all changed by the same constant factor, c_i^* will still remain an optimal center of cluster i. The same holds for cluster C_j^*. It follows that even in this perturbed metric, p prefers c_i^* to c_j^*. Hence $\alpha d(p, c_i^*) = d'(p, c_i^*) < d'(p, c_j^*) = d(p, c_j^*)$. □

Corollary 8.7. *For every point p and its center c_i^*, and for every point p' from a different cluster, it follows that $d(p, p') > (\alpha - 1)d(p, c_i^*)$.*

Proof. Denote by c_j^* the center of the cluster that p' belongs to. Now, consider two cases. Case (a): $d(p', c_j^*) \geq d(p, c_i^*)$. In this case, by traingle inequality we get that $d(p, p') \geq d(p', c_i^*) - d(p, c_i^*)$. Since the data instance is stable to α-perturbations, Fact 8.6 gives us that $d(p', c_i^*) > \alpha d(p', c_j^*)$. Hence we get that $d(p, p') > \alpha d(p', c_j^*) - d(p, c_i^*)$ $\geq (\alpha - 1)d(p, c_i^*)$. Case (b): $d(p', c_j^*) < d(p, c_i^*)$. Again by traingle inequality we get that $d(p, p') \geq d(p, c_j^*) - d(p', c_j^*) > \alpha d(p, c_i^*) - d(p', c_j^*) > (\alpha - 1)d(p, c_i^*)$. □

A key ingredient in the proof of Theorem 8.4 is the *tree-clustering* formulation of Balcan et. al [27]. In particular, we prove that if an instance satisfies α-center proximity for $\alpha \geq 3$ then it also satisfies the "min stability property" (defined below). This property, as shown in [27], is a (necessary and) sufficient condition for the Single-Linkage algorithm to produce a tree such that the optimal clustering is some pruning of this tree. In order to define the "min-stability" property, we first introduce the following notation. For any two subsets $A, B \subset S$, we denote the minimum distance between A and B as $d_{\min}(A, B) = \min\{d(a, b) \mid a \in A, b \in B\}$.

Definition 8.8. *A clustering instance satisfies the* min-stability property *if for any two clusters C and C' in the optimal clustering, and any two subsets $A \subsetneq C$, $A' \subseteq C'$, it holds that $d_{\min}(A, C \setminus A) \leq d_{\min}(A, A')$.*

In words, the min-stability property means that for any set A that is a strict subset of some cluster C in the optimal clustering, the closest point to A is a point from $C \setminus A$, and not from some other cluster. The next two lemmas lie at the heart of our algorithm.

Lemma 8.9. *A clustering instance (for a center-based clustering objective) in which centers are data points, that satisfies α-center proximity for $\alpha \geq 3$, also satisfies the min-stability property.*

Proof. Let C_i^*, C_j^* be any two clusters in the target clustering. Let A and A' be any two subsets s.t. $A \subsetneq C_i^*$ and $A' \subseteq C_j^*$. Let $p \in A$ and $p' \in A'$ be the two points which obtain the minimum distance $d_{\min}(A, A')$. Let $q \in C_i^* \setminus A$ be the nearest point to p. Also, denote by c_i^* and c_j^* the centers of clusters C_i^* and C_j^* respectively.

For the sake of contradiction, assume that $d_{\min}(A, C_i^* \setminus A) \geq d_{\min}(A, A')$. Suppose $c_i^* \notin A$. This means that $d(p, p') = d_{\min}(A, A') \leq d_{\min}(A, C_i^* \setminus A) \leq d(p, c_i^*)$. As $\alpha \geq 3$, this contradicts Corollary 8.7.

Thus we may assume $c_i^* \in A$. It follows that $d(q, c_i^*) \geq d(p, p') > (3-1)d(p, c_i^*) = 2d(p, c_i^*)$, so $d(p, c_i^*) < d(q, c_i^*)/2$. We therefore have that $d(p', c_i^*) \leq d(p, p') + d(p, c_i^*) \leq 3d(q, c_i^*)/2$. This implies that $d(p', c_j^*) < d(p', c_i^*)/\alpha < d(q, c_i^*)/2$, and thus $d(q, c_j^*) \leq d(q, c_i^*) + d(c_i^*, p) + d(p, p') + d(p', c_j^*) < 3d(q, c_i^*) \leq \alpha d(q, c_i^*)$. This contradicts Fact 8.6. □

Lemma 8.10. *A clustering instance (for a center-based clustering objective) in which centers need not be data points, that satisfies α-center proximity for $\alpha \geq 2 + \sqrt{3}$, also satisfies the min-stability property.*

Proof. As in the proof of Lemma 8.9, let C_i^*, C_j^* be any two clusters in the target clustering and let A and A' be any two subsets s.t. $A \subsetneq C_i^*$ and $A' \subseteq C_j^*$. Let $p \in A$ and $p' \in A'$ be the two points which obtain the minimum distance $d_{\min}(A, A')$ and let $q \in C_i^* \setminus A$ be the nearest point to p. Also, as in the proof of Lemma 8.9, let c_i^* and c_j^* denote the centers of clusters C_i^* and C_j^* respectively (though these need not be datapoints).

By definition of center-proximity, we have the following inequalities:

$$\begin{aligned} d(p, p') + d(p', c_j^*) &> \alpha d(p, c_i^*) \quad \text{[c.p. applied to } p\text{]} \\ d(p, p') + d(p, c_i^*) &> \alpha d(p', c_j^*) \quad \text{[c.p. applied to } p'\text{]} \\ d(p, p') + d(p', c_j^*) + d(p, q) &> \alpha(d(q, p) - d(p, c_i^*)) \\ &\text{[center proximity applied to } q \text{ and triangle ineq.]} \end{aligned}$$

Multiplying the first inequality by $1 - \frac{1}{\alpha+1} - \frac{1}{\alpha-1}$, the second by $\frac{1}{\alpha+1}$, the third by $\frac{1}{\alpha-1}$, and summing them together we get

$$d(p, p') > \tfrac{\alpha^2 - 4\alpha + 1}{\alpha - 1} d(p, c_i^*) + d(q, p),$$

which for $\alpha = 2 + \sqrt{3}$ implies $d(p, p') > d(q, p)$ as desired. □

8.2.2 The Algorithm

As mentioned, Balcan et al [27] proved (Theorem 2) that if an instance satisfies min-stability, then the tree on clusters produced by the single-linkage algorithm contains the optimal clustering as some k-pruning of it. I.e., the tree produced by starting with n clusters of size 1 (viewed as leaves), and at each step merging the two clusters C,C' minimizing $d_{\min}(C, C')$ (viewing the merged cluster as their parent) until only one cluster remains. Given the structural results proven above, our algorithm (see Figure 8.2) simply uses this clustering tree and finds the best k-pruning using dynamic programming.

1. Run Single-Linkage until only one cluster remains, producing the entire tree on clusters.

2. Find the best k-pruning of the tree by dynamic programming using the equality

$$\text{best-}k\text{-pruning}(T) = \min_{0<k'<k} \Big\{ \begin{array}{l} \text{best-}k'\text{-pruning}(T\text{'s left child}) \\ + \text{ best-}(k-k')\text{-pruning}(T\text{'s right child}) \Big\} \end{array}$$

Figure 8.2: Algorithm to find the optimal k-clustering of instances satisfying α-center proximity. The algorithm is described for the case (as in k-median or k-means) that Φ defines the overall score to be a sum over individual cluster scores. If it is a maximum (as in k-center) then replace "+" with "max" above.

Proof of Theorem 8.4. By Lemmas 8.9 and 8.10, the data satisfies the min-stability property, which as shown in [27] is sufficient to guarantee that some pruning of the single-linkage hierarchy is the target clustering. We then find the optimal clustering using dynamic programming by examining k-partitions laminar with the single-linkage clustering tree. The optimal k-clustering of a tree-node is either the entire subtree as one cluster (if $k = 1$), or the minimum over all choices of k_1-clusters over its left subtree and k_2-clusters over its right subtree (if $k > 1$). Here k_1, k_2 are positive integers, such that $k_1 + k_2 = k$. Therefore, we just traverse the tree bottom-up, recursively solving the clustering problem for each tree-node. By assumption that the clustering objective is separable, so each step including the base-case can be performed in polynomial time. For the case of k-median in a finite metric, for example, one can maintain a $n \times O(n)$ table for all possible centers and all possible clusters in the tree, yielding a running time of $O(n^2 + nk^2)$. For the case of k-means in Euclidean space, one can compute the cost of a single cluster by computing the center as just the average of all its points. In general, the overall running time is $O(n(k^2 + T(n)))$, where $T(n)$ denotes the time it takes to compute the cost of a single cluster. □

8.2.3 Some Natural Barriers

We complete this section with a discussion of barriers of our approach. First, our algorithm indeed fails on some finite metrics that are $(3 - \epsilon)$-perturbation resilient. For example, consider the instance shown in Figure 8.3. In this instance, the clustering tree produced by single-linkage is not laminar with the optimal k-median clustering. It is easy to check that this instance is resilient to α-perturbations for any $\alpha < 3$.

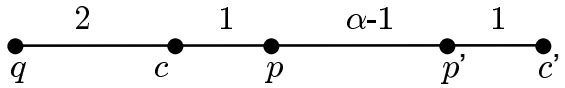


Figure 8.3: An example of a finite metric k-median instance with $2 < \alpha < 3$ where our algorithm fails. The optimal 2-median clustering is $\{c, p, q\}, \{c', p'\}$. In contrast, when we run our algorithm over on this instance, single linkage first connects $\{c, p\}$ with $\{c', p'\}$, and only then merges these 4 points with q.

Second, observe that our analysis, though emanating from perturbation resilience, only uses center proximity. We next show that for general metrics, one cannot hope to solve (in poly-time) k-median instances satisfying α-center proximity for $\alpha < 3$. This is close to our upper bound of $2 + \sqrt{3}$ for general metrics.

Theorem 8.11. *For any $\alpha < 3$, the problem of solving k-median instances over general metrics that satisfy α-center proximity is* NP*-hard.*

Proof. The proof of Theorem 8.11 follows from the classical reduction of Max-k-Coverage to k-median. In this reduction, we create a bipartite graph where the right-hand side vertices represent the elements in the ground set; the left-hand side vertices represent the given subsets; and the distance between the set-vertex and each element-vertex is 1, if the set contains that element. Using shortest-path distances, it follows that the distance from any element-vertex to a set-vertex to which it does not belong to is at least 3. Using the fact that the NP-hardness results for Max-k-Coverage holds for disjoint sets (i.e. the optimal solution of Yes-instances is composed of k disjoint sets, see [68]), the α-center proximity property follows. □

Lastly, we comment that using Single-Linkage in the usual way (namely, stopping when there are k clusters remaining) is *not* sufficient to produce a good clustering. We demonstrate this using the example shown in Figure 8.4. Observe, in this instance, since C contains significantly less points than A,B, or D, this instance is stable – even if we perturb distances by a factor of 3, the cost of any alternative clustering is higher than the cost of the optimal solution. However, because $d(A, C) > d(B, D)$, it follows that the usual version of Single-Linkage will unite B and D, and only then A and C. Hence, if we stop the Single-Linkage algorithm at $k = 3$ clusters, we will not get the desired clustering.

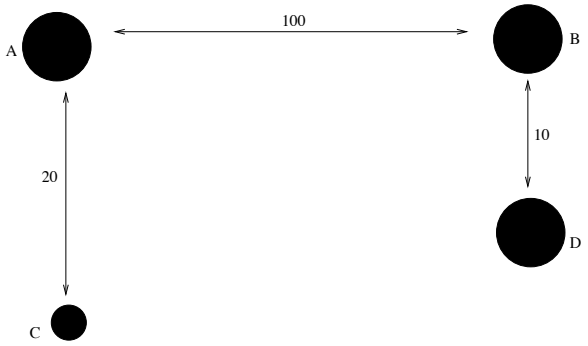


Figure 8.4: An example showing that the usual version of Single-Linkage fails. The instance is composed of 4 components, each with inner-distance ϵ and outer-distance as described in the figure. However, components A, B and D each contain 100 points, whereas component C has only 10 points. The optimal 3-median clustering consists of 3 clusters: $\{A, C\}, \{B\}, \{D\}$ and has cost $\mathsf{OPT} = 200 + 300\epsilon$.

8.3 Future Directions

There are several natural open questions left by this work. First, can one reduce the perturbation factor α needed for efficient clustering? As mentioned earlier, recently Balcan and Liang [29] have given a very interesting algorithm that reduces the $\alpha = 3$ factor needed by our algorithm for finite metrics to $1 + \sqrt{2}$. Additionally, they also extend our work in a different direction, giving algorithms for clustering instances that are "mostly resilient" to α-perturbations: having the property that under any α-perturbation of the underlying metric, no more than a δ-fraction of the points get mislabeled under the optimal solution. Reyzin [127] gives a NP-hardness for clustering 2-center proximity instances (with no Steiner points) as well as a single-pass algorithm that cluster instances which are ≈ 5.7 perturbation resilient.

Alternatively, one can consider a weaker notion of *resilience to perturbations on average*: a clustering instance whose optimal clustering is likely not to change, assuming the perturbation is *random* from some suitable distribution. Can this weaker notion be used to still achieve positive guarantees?

Chapter 9

Improved Spectral-Norm Bounds for Clustering

9.1 Introduction

In the long-studied field of clustering, there has been substantial work [56, 59, 13, 138, 4, 50, 96, 54, 47] studying the problem of clustering data from mixture of distributions under the assumption that the means of the distributions are sufficiently far apart. Each of these works focuses on one particular type (or family) of distribution, and devise an algorithm that successfully clusters datasets that come from that particular type. Typically, they show that w.h.p. such datasets have certain nice properties, then use these properties in the construction of the clustering algorithm.

The recent work of Kumar and Kannan [104] takes the opposite approach. First, they define a separation condition, deterministic and thus not tied to any distribution, and show that any set of data points satisfying this condition can be successfully clustered. Having established that, they show that many previously studied clustering problems indeed satisfy (w.h.p) this separation condition. These clustering problems include Gaussian mixture-models, the Planted Partition model of McSherry [110] and the work of Ostrovsky et al [121]. In this aspect they aim to unify the existing body of work on clustering under separation assumptions, proving that one algorithm applies in multiple scenarios.[1]

[1]We comment that, implicitly, Achlioptas and McSherry [4] follow a similar approach, yet they focus only on mixtures of Gaussians and log-concave distributions. In addition, the work of [53] studies a deterministic separation condition required for efficient clustering, extending the separation condition of [110]. The precise condition presented in [53] is technical but essentially assumes that the underlying graph over

However, the attempt to unify multiple clustering works is only successful in part. First, Kumar and Kannan's analysis is "wasteful" w.r.t the number of clusters k. Clearly, motivated by an underlying assumption that k is constant, their separation bound has linear dependence in k and their classification guarantee has quadratic dependence on k. As a result, Kumar and Kannan overshoot best known bounds for the Planted Partition Model and for mixture of Gaussians by a factor of $\sqrt{k}$. Similarly, the application to datasets considered by Ostrovsky et al only holds for constant k. Secondly, the analysis in Kumar-Kannan is far from simple – it relies on most points being "good", and requires multiple iterations of Lloyd steps before converging to good centers. Our work addresses these issues.

To formally define the separation condition of [104], we require some notation. Our input consists of n points in $\mathbb{R}^d$. We view our dataset as a $n \times d$ matrix, A, where each datapoint corresponds to a row A_i in this matrix. We assume the existence of a target partition, $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_k^*\}$, where each cluster's center is $\mu(C_r^*) = \frac{1}{n_r}\sum_{i \in C_r^*} A_i$, where $n_r = |C_r^*|$. Thus, the target clustering is represented by a $n \times d$ matrix of cluster centers, C, where the ith row of C equals $\mu(C_r^*)$ iff $i \in C_r^*$. Therefore, the k-means cost of this partition is the squared Frobenius norm $\|A-C\|_F^2$, but the focus of this paper is on the spectral (L_2) norm of the matrix $A-C$. Indeed, the deterministic equivalent of the maximal variance in any direction is, by definition, $\frac{1}{n}\|A - C\|^2 = \max_{\{v:\ \|v\|=1\}} \frac{1}{n}\|(A - C)v\|^2$.

Definition. *Fix $i \in C_r^*$. We say a datapoint A_i satisfies the* Kumar-Kannan proximity condition *if for any $s \neq r$, when projecting A_i onto the line connecting μ_r and μ_s, the projection of A_i is closer to $\mu(C_r^*)$ than to $\mu(C_s^*)$ by an additive factor of $\Omega\left(k(\frac{1}{\sqrt{n_r}} + \frac{1}{\sqrt{n_s}})\|A - C\|\right)$.*

Kumar and Kannan proved that if all but at most ϵ-fraction of the data points satisfy the proximity condition, they can find a clustering which is correct on all but an $O(k^2\epsilon)$-fraction of the points. In particular, when $\epsilon = 0$, their algorithm clusters all points correctly. Observe, the Kumar-Kannan proximity condition gives that the distance $\|\mu_r - \mu_s\|$ is also bigger than the above mentioned bound. The opposite also holds – one can show that if $\|\mu_r - \mu_s\|$ is greater than this bound then only few of the points do not satisfy the proximity condition.

the set of points has a "low rank structure" and presents an algorithm to recover this structure which is then enough to cluster well.

9.1.1 Our Contribution

Our Separation Condition. In this work, the bulk of our analysis is based on the following quantitatively weaker version of the proximity condition, which we call *center separation*. Formally, we define $\Delta_r = \frac{1}{\sqrt{n_r}} \min\{\sqrt{k}\|A - C\|, \|A - C\|_F\}$ and we assume throughout the paper that for a large constant[2] c we have that the means of any two clusters C_r^* and C_s^* satisfy

$$\|\mu(C_r^*) - \mu(C_s^*)\| \geq c(\Delta_r + \Delta_s) \tag{9.1}$$

Observe that this is a simpler version of the Kumar-Kannan proximity condition, scaled down by a factor of $\sqrt{k}$. Even though we show that (9.1) gives that only a few points do not satisfy the proximity condition, our analysis (for the most part) does not partition the dataset into good and bad points, based on satisfying or non-satisfying the proximity condition. Instead, our analysis relies on basic tools, such as the Markov inequality and the triangle inequality. In that sense one can view our work as "aligning" Kumar and Kannan's work with the rest of clustering-under-center-separation literature – we show that the bulk of Kannan and Kumar's analysis can be simplified to rely merely on center-separation.

Our results. We improve upon the results of [104] along several axes. In addition to the weaker condition of Equation (9.1), we also weaken the Kumar-Kannan proximity condition by a factor of k, and still retrieve the target clustering, if all points satisfy the (k-weaker) proximity condition. Secondly, if at most ϵn points do not satisfy the k-weaker proximity condition, we show that we can correctly classify all but a $(\epsilon + O(1/c^4))$-fraction of the points, improving over the bound of [104] of $O(k^2\epsilon)$. Note that our bound is meaningful even if ϵ is a constant whereas $k = \omega(1)$. Furthermore, we prove that the k-means cost of the clustering we output is a $(1 + O(1/c))$-approximation of the k-means cost of the target clustering.

Once we have improved on the main theorem of Kumar and Kannan, we derive immediate improvements on its applications. In Section 9.3.1 we show our analysis subsumes the work of Ostrovsky et al [121], and applies also to non-constant k. Using the fact that Equation (9.1) "shaves off" a $\sqrt{k}$ factor from the separation condition of Kumar and Kannan, we obtain a separation condition of $\Omega(\sigma_{\max}\sqrt{k})$ for learning a mixture of Gaussians,

[2]We comment that throughout the paper, and much like Kumar and Kannan, we think of c as a large constant ($c = 100$ will do). However, our results also hold when $c = \omega(1)$, allowing for a $(1 + o(1))$-approximation. We also comment that we think of $d \gg k$, so one should expect $\|A - C\|_F^2 \geq k\|A - C\|^2$ to hold, thus the reader should think of Δ_r as dependent on $\sqrt{k}\|A - C\|$. Still, including the degenerate case, where $\|A - C\|_F^2 < k\|A - C\|$, simplifies our analysis in Section 9.3. One final comment is that (much like all the work in this field) we assume k is given, as part of the input, and not unknown.

and we also match the separation results of the Planted Partition model of McSherry [110]. These results are described in Section 9.5.

Comparison with previous stability notions. In Chapter 7 we studied clustering instances satisfying weak-deletion stability – where merging two clusters in the optimal k-means clustering increases the cost significantly. In Chapter 8 we studied the notion of instances which are perturbation resilient – where any 3-multiplicative change to the distances doesn't change the optimal k-means clustering. Both of these notions are very different from the center-separation notion studied here. Think of the motivating example of learning a mixture of k-Gaussians in a high-dimensional space $\mathbb{R}^d$ (with $d > k$), and with $\Delta_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A - C\|$. In this example, merging clusters r and s by assigning the n_r points in cluster r to μ_s increases the cost by a factor of $n_r\Delta_r^2 = k\|A - C\|^2$. Assuming $k\|A-C\|^2 \ll \|A-C\|_F^2$, we have that the increase in cost is negligible and so the instance doesn't satisfy weak-deletion stability. Similarly, such mixture of Gaussians doesn't satisfy perturbation resilient, as the distance between cluster centers (under the right choice of parameters) is even smaller than the average distance of a point to its own cluster center. As we show, it is only after we project the instance onto the subspace spanned by its top k singular vectors that we get an instance where distances between centers are large in comparison to distances inside a cluster.

We comment that indeed, in the case where $\Delta_r = \frac{1}{\sqrt{n_r}}\|A - C\|_F$, then (as we discuss in Section 9.3.1) we do have an instance which is of the type considered by Ostrovsky et al [121] – and therefore the instance is also weak-deletion stable. Still, the algorithm we propose here is deterministic (as opposed to the randomized algorithm of [121]) and its running time is much smaller than the running time of the algorithm detailed in Chapter 7 (no exponential dependency on a parameter β). Finally, we comment that one can design instances in small dimension (even $d = 2$) satisfying perturbation resilience and yet they do not satisfy center separation. An example of such instance was given in Figure 8.1.

Organization. To formally detail our results, we first define some notations and discuss a few preliminary facts. The next section (Section 9.2) contains the details of the preliminaries, as well as a formal statement of our algorithm and its guarantees, and an overview of the proof. The first part of the analysis of our algorithms is in Section 9.3, and it is enough for us to give a "one-line" proof in Section 9.3.1 showing how the work of Ostrovsky et al falls into our framework. The second part of the analysis of our algorithm is in Section 9.4. The improved guarantees we get by applying the algorithm to the Planted Partition model and to the Gaussian mixture model are discussed in Section 9.5. We conclude

with an open problem in Section 9.6.

9.2 Notations and Preliminaries

9.2.1 Notation

The Frobenius norm of a $n \times m$ matrix M, denoted as $\|M\|_F$ is defined as $\|M\|_F = \sqrt{\sum_{i,j} M_{i,j}^2}$. The spectral norm of M is defined as $\|M\| = \max_{x:\|x\|=1} \|Mx\|$. It is a well known fact that if the rank of M is t, then $\|M\|_F^2 \leq t\|M\|^2$. Recall the Singular Value Decomposition (SVD) of M, denoted $M = U\Sigma V^T$, where U is a $n \times n$ unitary matrix, V is a $m \times m$ unitary matrix, Σ is a $n \times m$ diagonal matrix whose entries are nonnegative real numbers, and its diagonal entries satisfy $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min\{m,n\}}$. The columns of U and V, denoted u_i and v_i resp., are called the left- and right-singular vectors. As a convention, when referring to singular vectors, we mean the right-singular vectors. Projecting M onto its top t singular vectors means taking $\hat{M} = \sum_{i=1}^{t} \sigma_i u_i v_i^T$. It is a known fact that for any t, the t-dimensional subspace which best fits the rows of M, is obtained by projecting M onto the subspace spanned by the top t singular vectors (corresponding to the top t singular values). Another way to phrase this result is by saying that $\hat{M} = \arg\min_{N:\text{rank}(N)=t}\{\|M - N\|_F\}$. (For a proof, see [97].) The same matrix, $\hat{M}$, also minimizes the spectral norm of this difference, meaning $\hat{M} = \arg\min_{N:\text{rank}(N)=t}\{\|M - N\|\}$ (see [73] for a proof).

As previously defined, $\|A - C\|$ denotes the spectral norm of $A - C$. We abbreviate, and denote $\mu_r = \mu(C_r^*)$. From this point on, we denote the projection of A onto the subspace spanned by its top k-singular vectors as $\hat{A}$, and for any vector v, we denote $\hat{v}$ as the projection of v onto this subspace. Throughout the paper, we abuse notation and use i to iterate over the rows of A, whereas r and s are used to iterate over *clusters* (or submatrices). So A_i represents the ith row of A whereas A_r represents the submatrix $[A_i]_{\{i \in C_r^*\}}$.

9.2.2 Basic Facts.

The analysis of our main theorem makes use of the following facts, from [110, 97, 104]. We advise the reader to go over the proofs, which are short and elegant.

The first fact bounds the cost of assigning the points of $\hat{A}$ to their original centers.

Fact 9.1 (Lemma 9 from [110]). $\|\hat{A} - C\|_F^2 \leq 8\min\{k\|A - C\|^2, \|A - C\|_F^2\}$ $\Big(= 8n_r\Delta_r^2$ *for every* $r\Big)$.

Proof.

$$\|\hat{A} - C\|_F^2 \leq 2k\|\hat{A} - C\|^2 \leq 2k\left(\|\hat{A} - A\| + \|A - C\|\right)^2 \leq 2k\left(2\|A - C\|\right)^2$$

where the first inequality holds because $\text{rank}(\hat{A} - C) \leq 2k$, and the last inequality follows from the fact that $\hat{A} = \arg\min_{N:\text{rank}(N)=k}\{\|A - N\|\}$. For the same reason, $\|\hat{A} - C\|_F \leq \|A - \hat{A}\|_F + \|A - C\|_F \leq 2\|A - C\|_F$. □

Next, we show that we can match each target center μ_r to a unique, relatively close, center ν_r that we get in Part I of the algorithm (see Figure 9.1).

Fact 9.2 (Claim 1 in Section 3.2 of [97]). *For every μ_r there exists a center ν_s s.t. $\|\mu_r - \nu_s\| \leq 6\Delta_r$, so we can match each μ_r to a unique ν_r.*

Proof. Observe that by taking $\hat{A} - \hat{C}$, we project $A - C$ to a k-dimensional subspace, so we have that $\|\hat{A} - \hat{C}\|_F^2 \leq k\|\hat{A} - \hat{C}\|^2 \leq k\|A - C\|^2$. Similarly, $\|\hat{A} - \hat{C}\|_F^2 \leq \|A - C\|_F^2$.

Assume for the sake of contradiction that $\exists r$ s.t. $\|\mu_r - \nu_s\| > 6\Delta_r$ for all s. Since $\|\hat{A} - \hat{C}\|_F^2 \leq n_r\Delta_r^2$, then our 10-approximation algorithm yields a clustering of cost $\leq 10n_r\Delta_r^2$. In contrast, as each $\hat{A}_i$ is assigned to some $\nu_{c(i)}$, the contribution of only the points in C_r^* to the k-means cost of the clustering is more than

$$\sum_{i\in C_r^*}\left\|(\mu_r - \nu_{c(i)}) - (\hat{A}_i - \mu_r)\right\|^2 > \frac{n_r}{2}(6\Delta_r)^2 - \sum_{i\in C_r^*}\|\hat{A}_i - \mu_r\|^2 \geq 18n_r\Delta_r^2 - \|\hat{A} - C\|_F^2 \geq 10n_r\Delta_r^2$$

where the first inequality follows from the fact that $(a - b)^2 \geq \frac{1}{2}a^2 - b^2$. □

Finally, we exhibit the following fact, which is detailed in the analysis of [104].

Fact 9.3. *Fix a target cluster C_r^* and let S_r be a set of points created by removing $\rho_{out}n_r$ points from C_r^* and adding $\rho_{in}(s)n_r$ points from each cluster $s \neq r$, s.t. every added point x satisfies $\|x - \mu_s\| \geq \frac{2}{3}\|x - \mu_r\|$. Assume $\rho_{out} < \frac{1}{4}$ and $\rho_{in} \stackrel{\text{def}}{=} \sum_{s\neq r}\rho_{in}(s) < \frac{1}{4}$. Then*

$$\|\mu(S_r) - \mu_r\| \leq \frac{1}{\sqrt{n_r}}\left(\sqrt{\rho_{out}} + \tfrac{3}{2}\sum_{s\neq r}\sqrt{\rho_{in}(s)}\right)\|A - C\| \leq \left(\sqrt{\frac{\rho_{out}}{n_r}} + \tfrac{3}{2}\sqrt{k}\sqrt{\frac{\rho_{in}}{n_r}}\right)\|A - C\|$$

In order to prove Fact 9.3 we use the following Fact.

Fact 9.4 (Lemma 5.2 and Corollary 5.3 from [104])**.** *Fix any cluster C_r^* and a subset $X \subset C_r^*$. Then*

$$|X| \, \|\mu(X) - \mu_r\| = (|C_r^*| - |X|) \, \|\mu(C_r^* \setminus X) - \mu_r\| \leq \sqrt{|X|} \, \|A_r - C_r\|$$

Proof. Let u_X be the indicator vector of X. Then

$$\| \, |X| \, (\mu(X) - \mu_r) \, \| = \|(A_r - C_r)^T \, u_X\| \leq \|(A_r - C_r)^T\| \, \|u_X\| = \|A_r - C_r\| \sqrt{|X|}$$

and the fact that $|X| \, \|\mu(X) - \mu_r\| = |C_r^* \setminus X| \, \|\mu(C_r^* \setminus X) - \mu_r\|$ is simply because $\mu_r = \frac{|X|}{|C_r^*|}\mu(X) + \frac{|C_r^* \setminus X|}{|C_r^*|}\mu(C_r^* \setminus X)$. □

Proof of Fact 9.3. We break $\|\mu(S_r) - \mu_r\|$ into its components and deduce

$$\begin{aligned} \|\mu(S_r) - \mu_r\| &\leq \frac{(1-\rho_{out})n_r}{n_r}\|\mu(S_r \cap C_r^*) - \mu_r\| + \sum_{s \neq r} \frac{\rho_{in}(s) n_r}{n_r}\|\mu(S_r \cap C_s^*) - \mu_r\| \\ &\leq \frac{(1-\rho_{out})n_r}{n_r}\|\mu(S_r \cap C_r^*) - \mu_r\| + \tfrac{3}{2}\sum_{s \neq r} \frac{\rho_{in}(s) n_r}{n_r}\|\mu(S_r \cap C_s^*) - \mu_s\| \end{aligned}$$

Plugging in Fact 9.4 we have $\|\mu(S_r) - \mu_r\| \leq \frac{1}{n_r}\left(\sqrt{\rho_{out} n_r} + \frac{3}{2}\sum_{s \neq r}\sqrt{\rho_{in}(s) n_r}\right)\|A - C\|$. The last inequality comes from maximizing the sum of square-roots by taking each $\rho_{in}(s) = \rho_{in}/k$. □

9.2.3 Formal Description of the Algorithm and Our Theorems

Having established notation, we now present our algorithm, in Figure 9.1. Our algorithm's goal is three fold: (a) to find a partition that identifies with the target clustering on the majority of the points, (b) to have the k-means cost of this partition comparable with the target, and (c) output k centers which are close to the true centers. It is partitioned into 3 parts. Each part requires stronger assumptions, allowing us to prove stronger guarantees.

- Assuming only the center separation of (9.1), then **Part I** gives a clustering which (a) is correct on at least $1 - O(c^{-2})$ fraction of the points *from each target cluster* (Theorem 9.5), and (b) has k-means cost smaller than $(1 + O(1/c))\|A - C\|_F^2$ (Theorem 9.6).

Part I: Find initial centers:

- Project A onto the subspace spanned by the top k singular vectors.
- Run a 10-approximation algorithm[a] for the k-means problem on the projected matrix $\hat{A}$, and obtain k centers $\nu_1, \nu_2, \ldots, \nu_k$.

Part II: Set $S_r \leftarrow \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|, \text{ for every } s\}$ and $\theta_r \leftarrow \mu(S_r)$.

Part III: Repeatedly run Lloyd steps until convergence.

- Set $\Theta_r \leftarrow \{i : \|A_i - \theta_r\| \leq \|A_i - \theta_s\|, \text{ for every } s\}$.
- Set $\theta_r = \mu(\Theta_r)$.

[a]Throughout the paper, we assume the use of a 10-approximation algorithm. Clearly, it is possible to use *any* t-approximation algorithm, assuming c/t is a large enough constant.

Figure 9.1: Algorithm ∼Cluster

- Assuming also that $\Delta_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A-C\|$, i.e. assuming the *non-degenerate* case where $\|A - C\|_F^2 \geq k\|A - C\|^2$, then **Part II** finds centers that are $O(1/c)\frac{\|A-C\|}{\sqrt{n_r}}$ close to the true centers (Theorem 9.7). As a result (see Section 9.4.1), if $(1 - \epsilon)n$ points satisfy the proximity condition (weakened by a k factor,), then we misclassify no more than $(\epsilon + O(c^{-4}))n$ points.
- Assuming all points satisfy the proximity condition (weakened by a k-factor), **Part III** finds *exactly* the target partition (Theorem 9.14).

9.2.4 Proofs Overview

Proof outline for Section 9.3. The first part of our analysis is an immediate application of Facts 9.1 and 9.2. Our assumption dictates that the distance between any two centers is big ($\geq c(\Delta_r + \Delta_s)$). Part I of the algorithm assigns each projected point $\hat{A}_i$ to the nearest ν_r instead of the true center μ_r and Fact 9.2 assures that the distance $\|\mu_r - \nu_r\|$ is small ($< 6\Delta_r$). Consider a misclassified point A_i, where $\|A_i - \mu_r\| < \|A_i - \mu_s\|$ yet $\|\hat{A}_i - \nu_s\| < \|\hat{A}_i - \nu_r\|$. The triangle inequality assures that $\hat{A}_i$ has a fairly big distance to its true center ($> (\frac{c}{2} - 12)\Delta_r$). We deduce that each misclassified point contributes $\Omega(c^2\Delta_r^2)$ to the k-means cost of assigning all projected points to their true centers. Fact 9.1 bounds

this cost by $\|\hat{A} - C\|_F^2 \leq 8n_r\Delta_r^2$, so the Markov inequality proves only a few points are misclassified. Additional application of the triangle inequality for misclassified points gives that the distance between the original point A_i and a true center μ_r is comparable to the distance $\|A_i - \mu_s\|$, and so assigning A_i to the cluster s only increases the k-means cost by a small factor.

Proof outline for Section 9.4. In the second part of our analysis we compare between the true clustering $\mathcal{C}^*$ and some proposed clustering $\mathcal{S}$, looking *both* at the number of misclassified points *and* at the distances between the matching centers $\|\mu_r - \theta_r\|$. As Kumar and Kannan show, the two measurements are related: Fact 9.3 shows how the distances between the means depend on the number of misclassified points, and the main lemma (Lemma 9.11) essentially shows the opposite direction. These two relations are how Kumar and Kannan show that Lloyd steps converge to good centers, yielding clusters with few misclassified points. They repeatedly apply (their version of) the main lemma, showing that with each step the distances to the true means decrease and so fewer of the good points are misclassified.

To improve on Kumar and Kannan analysis, we improve on the two above-mentioned relations. Lemma 9.11 is a simplification of a lemma from Kumar and Kannan, where instead of projecting into a k-dimensional space, we project only into a 4-dimensional space, thus reducing dependency on k. However, the dependency of Fact 9.3 on k is tight[3]. So in Part II of the algorithm we devise sub-clusters S_r s.t. $\rho_{in}(s) = \rho_{out}/k^2$. The crux in devising S_r lies in Proposition 9.10 – we show that any misclassified projected point $i \in C_s^* \cap S_r$ is essentially misclassified by $\hat{\mu_r}$. And since (see [4]) $\|\mu_r - \hat{\mu_r}\| \leq \frac{1}{\sqrt{k}}\Delta_r$ (compared to the bound $\|\mu_r - \nu_r\| \leq 6\Delta_r$), we are able to give a good bound on $\rho_{in}(s)$.

Recall that we rely only on center separation rather than a large batch of points satisfying the Kumar-Kannan separation, and so we do not apply iterative Lloyd steps (unless all points are good). Instead, we apply the main lemma only once, w.r.t to the misclassified points in $C_s^* \cap S_r$, and deduce that the distances $\|\mu_r - \theta_r\|$ are small. In other words, Part II is a single step that retrieve centers whose distances to the original centers are $\sqrt{k}$-times better than the centers retrieved by Kumar and Kannan in numerous Lloyd iterations.

[3]In fact, Fact 9.3 is exactly why the case of $k = \omega(1)$ is hard – because the L_1 and L_2 norms of the vector $(\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{k}}, \ldots, \frac{1}{\sqrt{k}})$ are not comparable for non-constant k.

9.3 Part I of the Algorithm

In this section, we look only at Part I of our algorithm. Our approximation algorithm defines a clustering $\mathcal{T}$, where $T_r = \{i : \|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\| \text{ for every } s\}$. Our goal in this section is to show that $\mathcal{T}$ is correct on all but a small constant fraction of the points, and furthermore, the k-means cost of $\mathcal{T}$ is no more than $(1 + O(1/c))$ times the k-means cost of the target clustering.

Theorem 9.5. *There exists a matching (given by Fact 9.2) between the target clustering $\mathcal{C}^*$ and the clustering $\mathcal{T} = \{T_r\}_r$ where $T_r = \{i : \|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\|$ for every $s\}$ that satisfies the following properties:*

- *For every cluster $C^*_{s_0}$ in the target clustering, no more than $O(1/c^2)|C^*_{s_0}|$ points are misclassified.*
- *For every cluster T_{r_0} in the clustering that the algorithm outputs, we add no more than $O(1/c^2)|C^*_{r_0}|$ points from other clusters.*
- *At most $O(1/c^2)|C^*_{r_2}|$ points are misclassified overall, where $C^*_{r_2}$ is the second largest cluster.*

Proof. Let us denote $T_{s\to r}$ as the set of points $\hat{A}_i$ that are assigned to C^*_s in the target clustering, yet are closer to ν_r than to any other ν'_r. From triangle inequality we have that $\|\hat{A}_i - \mu_s\| \geq \|\hat{A}_i - \nu_s\| - \|\mu_s - \nu_s\|$. We know from Fact 9.2 that $\|\mu_s - \nu_s\| \leq 6\Delta_s$. Also, since $\hat{A}_i$ is closer to ν_r than to ν_s, the triangle inequality gives that $2\|\hat{A}_i - \nu_s\| \geq \|\nu_r - \nu_s|$. So,

$$\|\hat{A}_i - \mu_s\| \geq \frac{1}{2}\|\nu_r - \nu_s\| - 6\Delta_s \geq \frac{1}{2}\|\mu_r - \mu_s\| - 12(\Delta_r + \Delta_s) \geq \frac{c}{4}(\Delta_r + \Delta_s)$$

Thus, we can look at $\|\hat{A} - C\|_F^2$, and using Fact 9.1 we immediately have that for every fixed r'

$$\sum_r \sum_{s\neq r} |T_{s\to r}| \frac{c^2}{16}(\Delta_r + \Delta_s)^2 \leq \sum_r \sum_{i\in C^*_r} \|\hat{A}_i - \mu_r\|^2 = \|\hat{A} - C\|_F^2 \leq 8n_{r'}\Delta^2_{r'}$$

The proof of the theorem follows from fixing some r_0 or some s_0 and deducing:

$$\Delta^2_{s_0} \sum_{r\neq s_0} |T_{s_0\to r}| \leq \sum_{r\neq s_0} |T_{s_0\to r}|(\Delta_r + \Delta_{s_0})^2 \leq \sum_r \sum_{s\neq r} |T_{s\to r}|(\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{s_0}\Delta^2_{s_0}$$

$$\Delta_{r_0}^2 \sum_{s \neq r_0} |T_{s \to r_0}| \leq \sum_{s \neq r_0} |T_{s \to r_0}| (\Delta_{r_0} + \Delta_s)^2 \leq \sum_r \sum_{s \neq r} |T_{s \to r}| (\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{r_0} \Delta_{r_0}^2$$

Observe that for every $r \neq s$ we have that $\Delta_r + \Delta_s \geq \Delta_{r_2}$ (where r_2 is the cluster with the second largest number of points), so we have that

$$\Delta_{r_2}^2 \sum_r \sum_{s \neq r} |T_{s \to r}| \leq \sum_r \sum_{s \neq r} |T_{s \to r}| (\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{r_2} \Delta_{r_2}^2 \qquad \square$$

We now show that the k-means cost of $\mathcal{T}$ is close to the k-means cost of $\mathcal{C}^*$. Observe that the k-means cost of $\mathcal{T}$ is computed w.r.t the best center of each cluster (i.e., $\mu(T_r)$), and *not* w.r.t the centers ν_r.

Theorem 9.6. *The k-means cost of $\mathcal{T}$ is at most $(1 + O(1/c))\|A - C\|_F^2$.*

Proof. Given $\mathcal{T}$, it is clear that the centers that minimize its k-means cost are $\mu(T_r) = \frac{1}{|T_r|} \sum_{i \in T_r} A_i$. Recall that the majority of points in each T_r belong to a unique C_r^*, and so, throughout this section, we assume that all points in T_r were assigned to μ_r, and not to $\mu(T_r)$. (Clearly, this can only increase the cost.) We show that by assigning the points of T_r to μ_r, our cost is at most $(1 + O(1/c))\|A - C\|_F^2$, and so Theorem 9.6 follows. In fact, we show something stronger. We show that by assigning all the points in T_r to μ_r, each point A_i pays no more than $(1 + O(1/c))\|A_i - C_i\|^2$. This is clearly true for all the points in $T_r \cap C_r^*$. We show this also holds for the misclassified points.

Because $i \in T_{s \to r}$, it holds that $\|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\|$. Observe that for every s we have that $\|A_i - \nu_s\|^2 = \|A_i - \hat{A}_i\|^2 + \|\hat{A}_i - \nu_s\|^2$, because $\hat{A}_i - \nu_s$ is the projection of $A_i - \nu_s$ onto the subspace spanned by the top k-singular vectors of A. Therefore, it is also true that $\|A_i - \nu_r\| \leq \|A_i - \nu_s\|$. Because of Fact 9.2, we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$ and $\|\mu_s - \nu_s\| \leq 6\Delta_s$, so we apply the triangle inequality and get

$$\|A_i - \mu_r\| \leq \|A_i - \mu_s\| + \|\mu_r - \nu_r\| + \|\mu_s - \nu_s\| \leq \|A_i - \mu_s\| \left(1 + \frac{6(\Delta_r + \Delta_s)}{\|A_i - \mu_s\|}\right)$$

So all we need to do is to lower bound $\|A_i - \mu_s\|$. As noted, $\|A_i - \nu_s\| \geq \|\hat{A}_i - \nu_s\|$. Thus

$$\|A_i - \mu_s\| \geq \|A_i - \nu_s\| - 6\Delta_r \geq \|\hat{A}_i - \nu_s\| - 6\Delta_r \geq \frac{1}{2}\|\nu_s - \nu_r\| - 6\Delta_r \geq \frac{1}{4} c(\Delta_r + \Delta_s)$$

and we have the bound $\|A_i - \mu_r\| \leq \left(1 + \frac{24}{c}\right)\|A_i - \mu_s\|$, so $\|A_i - \mu_r\|^2 \leq \left(1 + \frac{49}{c}\right)\|A_i - \mu_s\|^2$. $\square$

9.3.1 Application: The ORSS-Separation

One straight-forward application of Theorem 9.6 is for the datasets considered by Ostrovsky et al [121], where the optimal k-means cost is an ϵ-fraction of the optimal $(k-1)$-means cost. Ostrovsky et al proved that for such datasets a variant of the Lloyd method converges to a good solution in polynomial time. Kumar and Kannan have shown that datasets satisfying the ORSS-separation, also have the property that most points satisfy their proximity-condition. Their analysis is not immediate, and gives a $(1+O(\sqrt{k\epsilon}))$-approximation. Here, we provide a "one-line" proof that Part I of Algorithm $\sim$Cluster yields a $(1+O(\sqrt{\epsilon}))$-approximation, for any k.

Suppose we have a dataset satisfying the ORSS-separation condition, so any $(k-1)$-partition of the dataset have cost $\geq \frac{1}{\epsilon}\|A-C\|_F^2$. For any r and any $s \neq r$, by assigning all the points in C_r^* to the center μ_s, we get some $(k-1)$-partition whose cost is exactly $\|A-C\|_F^2 + n_r\|\mu_r - \mu_s\|^2$, so $\|\mu_r - \mu_s\| \geq \frac{\sqrt{\frac{1}{\epsilon}-1}}{\sqrt{n_r}}\|A-C\|_F$. Setting $c = O(1/\sqrt{\epsilon})$, Theorem 9.6 is immediate.

9.4 Part II of the Algorithm

In this section, our goal is to show that Part II of our algorithm gives centers that are very close to the target clusters. We should note that from this point on, we assume we are in the non-degenerate case, where $\|A-C\|_F^2 \geq k\|A-C\|^2$. Therefore, $\Delta_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A-C\|$.

Recall, in Part II we define the sets $S_r = \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|,\ \forall s \neq r\}$. Observe, these set do not define a partition of the dataset! There are some points that are not assigned to any S_r. However, we only use the centers of S_r. We prove the following theorem.

Theorem 9.7. *Denote $S_r = \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|,\ \forall s \neq r\}$. Then for every r it holds that $\|\mu(S_r) - \mu_r\| = O(1/c)\frac{1}{\sqrt{n_r}}\|A-C\| = O(\frac{1}{c\sqrt{k}}\Delta_r)$.*

The proof of Theorem 9.7 is an immediate application of Fact 9.3 combined with the following two lemmas, that bound the number of misclassified points. Observe that for every point that belongs to C_s^* yet is assigned to S_r (for $s \neq r$) is also assigned to T_r in the clustering $\mathcal{T}$ discussed in the previous section. Therefore, any misclassified point $i \in C_s^* \cap S_r$ satisfies that $\|A_i - \mu_r\| \leq (1+O(c^{-1}))\|A_i - \mu_s\|$ as the proof of Theorem 9.6 shows. So all conditions of Fact 9.3 hold.

Lemma 9.8. *Assume that for every r we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$. Then at most $\frac{512}{c^2} n_r$ points of C_r^* do not belong to S_r.*

Lemma 9.9. *Redefine $T_{s\to r}$ as the set $C_s^* \cap S_r$. Assume that for every r we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$. Then for every r and every $s \neq r$ we have that $|T_{s\to r}| = \left(\frac{48^2}{c^4 k^2}\right) n_r$.*

Proof of Lemma 9.8. First, we claim that if i is such that $\|\hat{A}_i - \mu_r\| \leq \frac{c}{8}\Delta_r$, then it must be the case that $i \in S_r$.

This is a simple consequence of the triangle inequality, bounding $\|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \mu_r\| + \|\mu_r - \nu_r\| \leq ((c/8) + 6)\Delta_r$. Yet, for every $s \neq r$, the triangle inequality gives that $\|\hat{A}_i - \nu_s\| \geq \|\mu_r - \mu_s\| - \|\hat{A}_i - \mu_r\| - \|\mu_s - \nu_s\| \geq (c - \frac{c}{8} - 6)(\Delta_r + \Delta_s)$. Assuming $c > 48$, we have that $\|\hat{A}_i - \nu_s\| \geq 3\|\hat{A}_i - \nu_r\|$.

All that's left is to show that the number of $i \in C_r^*$ s.t. $\|\hat{A}_i - \mu_r\| > \frac{c}{8}\Delta_r$ is small. This again follows from the Markov inequality: Since $\|\hat{A} - C\|_F^2 \leq 8k\|A - C\|^2$, then the number of such points is at most $\frac{8k\|A-C\|^2}{(c^2/64)k\|A-C\|^2} n_r$. □

We now turn to proving Lemma 9.9. The general outline of the proof of Lemma 9.9 resembles to the outline of the proof of Lemma 9.8. Proposition 9.10 exhibit some property that every point in $T_{s\to r}$ must satisfy, and then we show that only few of the points in C_s^* satisfy this property. Recall that $\hat{\mu_r}$ indicates the projection of μ_r onto the subspace spanned by the top k-singular vectors of A.

Proposition 9.10. *Fix $i \in C_s^*$ s.t. $\|\hat{A}_i - \hat{\mu}_s\| \leq 2\|\hat{A}_i - \hat{\mu}_r\|$. Then $\|\hat{A}_i - \nu_s\| < 3\|\hat{A}_i - \nu_r\|$, so $i \notin S_r$.*

Proof. First, for every r we have that $\|\hat{\mu_r} - \nu_r\| \leq \|\mu_r - \nu_r\| \leq 6\Delta_r$, as $\hat{\mu_r} - \nu_r$ is a projection of $\mu_r - \nu_r$.

Let us fiddle with the triangle inequality, in order to obtain a lower bound on $\|\hat{A}_i - \nu_r\|$. We have that $3\|\hat{A}_i - \hat{\mu}_r\| \geq \|\hat{\mu}_r - \hat{\mu}_s\| \geq \|\mu_r - \mu_s\| - \big(\|\mu_r - \nu_r\| + \|\nu_r - \hat{\mu}_r\|\big) - \big(\|\mu_s - \nu_s\| + \|\nu_s - \hat{\mu}_s\|\big) \geq (c - 12)(\Delta_r + \Delta_s)$, thus $\|\hat{A}_i - \nu_r\| \geq \left(\frac{c-12}{3} - 6\right)(\Delta_r + \Delta_s)$.

Assume for the sake of contradiction that $\|\hat{A}_i - \nu_s\| \geq 3\|\hat{A}_i - \nu_r\|$, and let us show this yields an upper bound on $\|\hat{A}_i - \nu_r\|$, which contradicts our lower bound. We have that

$$6\Delta_s \geq \|\hat{A}_i - \nu_s\| - \|\hat{A}_i - \hat{\mu}_s\| \geq 3\|\hat{A}_i - \nu_r\| - 2\|\hat{A}_i - \hat{\mu}_r\| \geq \|\hat{A}_i - \nu_r\| - 2 \cdot 6\Delta_r$$

It follows that $12(\Delta_r + \Delta_s) \geq \|\hat{A}_i - \nu_r\| \geq \left(\frac{c-12}{3} - 6\right)(\Delta_r + \Delta_s)$. Contradiction ($c > 60$). □

Proposition 9.10, shows that in order to bound $|T_{s\to r}|$ it suffices to bound the number of points in C_s^* satisfying $\|\hat{A}_i - \hat{\mu}_s\| \geq 2\|\hat{A}_i - \hat{\mu}_r\|$. The major tool in providing this bound is the following technical lemma. This lemma is a variation on the work of [104], on which we improve on the dependency on k and simplify the proof.

Lemma 9.11 (Main Lemma)**.** *Fix $\alpha, \beta > 0$. Fix $r \neq s$ and let ζ_r and ζ_s be two points s.t. $\|\mu_r - \zeta_r\| \leq \alpha\Delta_r$ and $\|\mu_s - \zeta_s\| \leq \alpha\Delta_s$. We denote $\tilde{A}_i$ as the projection of A_i onto the line connecting ζ_r and ζ_s. Define $X = \left\{ i \in C_s^* \,:\, \|\tilde{A}_i - \zeta_s\| - \|\tilde{A}_i - \zeta_r\| \geq \beta\|\zeta_s - \zeta_r\| \right\}$. Then $|X| \leq 256\frac{\alpha^2}{\beta^2}\,\frac{1}{c^4 k}\big(\min\{n_r, n_s\}\big)$.*

Proof. Let $\mathcal{V}$ be the subspace spanned by the following 4 vectors: $\{\mu_r, \mu_s, \zeta_r, \zeta_s\}$. Denote $P_{\mathcal{V}}$ as the projection onto $\mathcal{V}$. We denote $v_i = P_{\mathcal{V}}(A_i)$, and observe that $P_{\mathcal{V}}(\mu_r) = \mu_r$, and the same goes for μ_s, ζ_r and ζ_s. Observe also that, as a projection, $\|P_{\mathcal{V}}(A-C)\| \leq \|A-C\|$ (alternatively, $\|P_{\mathcal{V}}\| = 1$).

We now make a simple observation. Let $\bar{A}_i$ denote the projection of A_i onto the line connecting μ_r and μ_s. Now, the inequality $\|A_i - \mu_s\| < \|A_i - \mu_r\|$ holds iff the inequality $\|\bar{A}_i - \mu_s\| \leq \|\bar{A}_i - \mu_r\|$ holds (because $\|A_i - \mu_r\|^2 = \|A_i - \bar{A}_i\|^2 + \|\bar{A}_i - \mu_r\|^2$). Furthermore, such relation holds for any point whose projection on the line connecting μ_r and μ_s is identical to $\bar{A}_i$. In particular, if $\mathcal{W}$ is *any* subspace containing μ_r and μ_s, then the projection of A_i onto $\mathcal{W}$ is closer to μ_r than to μ_s iff A_i is closer to μ_r than to μ_s. Thus, since $\|A_i - \mu_s\| \leq \|A_i - \mu_r\|$ then $\|v_i - \mu_s\| \leq \|v_i - \mu_r\|$. Furthermore, as ζ_r and ζ_s also belong to $\mathcal{V}$, then the projection of A_i onto the line connecting ζ_s and ζ_r is identical to the projection of v_i onto the same line (meaning, $\tilde{A}_i = \tilde{v}_i$). So v_i also satisfies the inequality: $\|\tilde{v}_i - \zeta_s\| - \|\tilde{v}_i - \zeta_r\| \geq \beta\|\zeta_s - \zeta_r\|$, and, of course, $\|v_i - \zeta_r\|^2 = \|v_i - \tilde{v}_i\|^2 + \|\tilde{v}_i - \zeta_r\|^2$.

The proof follows from upper- and lower-bounding the term $\|v_i - \zeta_s\|^2 - \|v_i - \zeta_r\|^2$. We've just shown a lower bound, as we have that

$$\|v_i - \zeta_s\|^2 - \|v_i - \zeta_r\|^2 = \left(\|\tilde{v}_i - \zeta_s\| - \|\tilde{v}_i - \zeta_r\|\right)\,\left(\|\tilde{v}_i - \zeta_s\| + \|\tilde{v}_i - \zeta_r\|\right) \geq \beta\|\zeta_s - \zeta_r\|^2$$

The triangle inequality gives that $\|v_i - \zeta_s\| \leq \|v_i - \mu_s\| + \alpha(\Delta_r + \Delta_s)$, and that $\|v_i - \zeta_r\| \geq \|v_i - \mu_r\| - \alpha(\Delta_r + \Delta_s)$, so we have the upper bound of

$$\begin{aligned}
\|v_i - \zeta_s\|^2 - \|v_i - \zeta_r\|^2 &\leq \left(\|v_i - \mu_s\| + \alpha(\Delta_r + \Delta_s)\right)^2 - \left(\|v_i - \mu_r\| - \alpha(\Delta_r + \Delta_s)\right)^2 \\
&\leq \left(\|v_i - \mu_r\| + \alpha(\Delta_r + \Delta_s)\right)^2 - \left(\|v_i - \mu_r\| - \alpha(\Delta_r + \Delta_s)\right)^2 \\
&\leq 4\alpha(\Delta_r + \Delta_s)\|v_i - \mu_r\|
\end{aligned}$$

Comparing the upper and the lower bound, we have that for any $i \in X$ the distance

$\|v_i - \mu_r\| \geq \frac{\beta}{4\alpha}\frac{(c-\alpha)^2(\Delta_r+\Delta_s)^2}{\Delta_r+\Delta_s}$. As $X \subset C_s^*$, the Markov inequality concludes the proof

$$|X|\left(\frac{c^2}{8}\frac{\beta}{\alpha}\sqrt{k}\|A-C\|\right)^2\frac{1}{\min\{n_r,n_s\}} \leq \sum_{i\in C_s^*}\|v_i-\mu_s\|^2 \leq \|P_{\mathcal{V}}(A-C)\|_F^2 \leq 4\|A-C\|^2$$

□

Proof of Lemma 9.9. Every $i \in T_{s\to r}$ must satisfy that $\|\hat{A}_i - \hat{\mu}_s\| \geq 2\|\hat{A}_i - \hat{\mu}_r\|$ (Proposition 9.10). Therefore, we must have that $\|\tilde{A}_i - \hat{\mu}_s\| \geq 2\|\tilde{A}_i - \hat{\mu_r}\|$, where we denote $\tilde{A}_i$ as the projection of A onto the line connecting $\hat{\mu}_r$ with $\hat{\mu}_s$ (simply because $\|\hat{A}_i - \hat{\mu}_s\|^2 = \|\hat{A}_i - \tilde{A}_i\|^2 + \|\tilde{A}_i - \hat{\mu}_s\|^2$.) Therefore, $\|\hat{\mu}_r - \hat{\mu}_s\| \leq \frac{3}{2}\|\tilde{A}_i - \hat{\mu}_s\|$, so $\|\tilde{A}_i - \hat{\mu}_s\| - \|\tilde{A}_i - \hat{\mu}_r\| > \frac{1}{3}\|\hat{\mu}_r - \hat{\mu}_s\|$.

Thus, every $i \in T_{s\to r}$ satisfies the conditions of Lemma 9.11 with $\zeta_r = \hat{\mu}_r, \zeta_s = \hat{\mu}_s$, and $\beta = 1/3$. We deduce the $|T_{s\to r}| \leq \alpha^2\,\frac{256\cdot 9}{c^4 k}\min\{n_r, n_s\}$, where α is the bound s.t. for every r, $\|\mu_r - \hat{\mu}_r\| \leq \alpha\frac{\sqrt{k}}{\sqrt{n_r}}\|A-C\|$. Since $\alpha \leq \frac{1}{\sqrt{k}}$, we conclude the proof.

The fact that α is small was proven by Achlioptas and McSherry (Theorem 1 of [4]). Denote u_r as the indicator vector of C_r^*. Since $\text{rank}(C) \leq k$, we get

$$\|\mu_r - \hat{\mu}_r\| = \frac{1}{n_r}\|(A-\hat{A})^T u_r\| \leq \frac{1}{n_r}\|u_r\|\,\|A-\hat{A}\| \leq \frac{1}{\sqrt{n_r}}\|A-C\| \qquad \square$$

As an interesting corollary, Theorem 9.7 dictates that for every r we have that $\|\mu_r - \theta_r\| = O(1/c)\|\mu_r - \hat{\mu}_r\|$.

9.4.1 The Proximity Condition – Part III of the Algorithm

Part II of our algorithm returns centers $\theta_1, \ldots, \theta_k$ which are $O(\frac{1}{c\sqrt{n_r}})\|A-C\|$ close to the true centers. Suppose we use these centers to cluster the points: $\Theta_s = \{i : \forall s',\ \|A_i - \theta_s\| \leq \|A_i - \theta_{s'}\|\}$. It is evident that this clustering correctly classifies the majority of the points. It correctly classifies any point $i \in C_s^*$ with $\|A_i - \mu_r\| - \|A_i - \mu_s\| = \Omega(\frac{1}{c\sqrt{n_r}})\|A - C\|$ for every $r \neq s$, and the analysis of Theorem 9.5 shows that at most $O(c^{-2})$-fraction of the points do not satisfy this condition. In order to have a direct comparison with the Kumar-Kannan analysis, we now bound the number of misclassified points w.r.t the fraction of points satisfying the Kumar-Kannan proximity condition.

Definition 9.12. *Denote $gap_{r,s} = (\frac{1}{\sqrt{n_r}} + \frac{1}{\sqrt{n_s}})\|A-C\|$. Call a point $i \in C_s^*$ γ-*good*, if for every $r \neq s$ we have that the projection of A_i onto the line connecting μ_r and μ_s, denoted $\bar{A}_i$, satisfies that $\|\bar{A}_i - \mu_r\| - \|\bar{A}_i - \mu_s\| \geq \gamma\, gap_{r,s}$; otherwise we say the point is γ-*bad.

Corollary 9.13. *If the number of γ-bad points is ϵn, then (a) the clustering $\{\Theta_1, \ldots, \Theta_k\}$ misclassifies no more than $\left(\epsilon + \frac{O(1)}{\gamma^2 c^4}\right) n$ points, and (b) $\epsilon < O\left((c - \frac{\gamma}{\sqrt{k}})^{-2}\right)$, assuming $\gamma < c\sqrt{k}$.*

Proof. Clearly, all ϵn bad points may be misclassified. In addition, for every r and $s \neq r$, Lemma 9.11 (setting $\zeta_r = \theta_r$, $\zeta_s = \theta_s$, $\alpha = 1/c\sqrt{k}$ and $\beta = \Omega(\gamma/(c\sqrt{k}))$) proves that no more than $O(\gamma^{-2}c^{-2}k^{-1})n_s$ good points can be misclassified. Summing $\sum_{s \neq r} \frac{1}{k} n_s \leq n$, we conclude (a).

The proof of (b) is similar to the proof of Theorem 9.5. We look at the k-means cost of $\|\hat{A} - C\|_F^2$. We show that all γ-bad points contribute a large amount to this cost.

Take A_i to be a γ-bad point from C_s^*. Projecting it down to the line connecting μ_r and μ_s, we denote the projection as $\bar{A}_i$. Clearly, $\|\mu_r - \mu_s\| = \|\mu_r - \bar{A}_i\| + \|\bar{A}_i - \mu_s\| \geq c\sqrt{k} gap_{r,s}$ whereas $\|\mu_r - \bar{A}_i\| - \|\bar{A}_i - \mu_s\| \leq \gamma gap_{r,s}$. It follows that $\|\hat{A}_i - \mu_s\| \geq \|\bar{A}_i - \mu_s\| \geq \frac{1}{2}(c\sqrt{k} - \gamma) gap_{r,s} \geq \frac{c\sqrt{k}-\gamma}{2\sqrt{n_s}} \|A - C\|$. Again, the Markov inequality gives that

$$\#\{\text{bad points from } C_s^*\} \frac{(c\sqrt{k} - \gamma)^2}{4n_s} \|A - C\|^2 \leq \|\hat{A} - C\|_F^2 \leq 8k\|A - C\|^2$$

so from each cluster, only a fraction of $32\left(\frac{\sqrt{k}}{c\sqrt{k}-\gamma}\right)^2$ of the points can be bad. □

Observe that Corollary 9.13 allows for multiple scaled versions of the proximity condition, based on the magnitude of γ. In particular, setting $\gamma = 1$ we get a proximity condition whose bound is independent of k, and still our clustering misclassifies only a small fraction of the points – at most $O(c^{-2})$ fraction of all points might be misclassified because they are 1-bad, and no more than a $O(c^{-4})$-fraction of 1-good points may be misclassified. In addition, if there are no 1-bad points we show the following theorem. The proof (omitted) merely follows the Kumar-Kannan proof, plugging in the better bounds, provided by Lemma 9.11.

Theorem 9.14. *Assume all data points are 1-good. That is, for every point A_i that belongs to the target cluster $T_{c(i)}$ and every $s \neq c(i)$, by projecting A_i onto the line connecting $\mu_{c(i)}$ with μ_s we have that the projected point $\bar{A}_i$ satisfies $\|\bar{A}_i - \mu_{c(i)}\| - \|\bar{A}_i - \mu_s\| = \Omega\left((\frac{1}{\sqrt{n_{c(i)}}} + \frac{1}{\sqrt{n_s}})\right)\|A - C\|$, whereas $\|\mu_{c(i)} - \mu_s\| = \Omega\left(\sqrt{k}(\frac{1}{\sqrt{n_{c(i)}}} + \frac{1}{\sqrt{n_s}})\right)\|A - C\|$. Then the Lloyd method, starting with $\theta_1, \ldots, \theta_k$, converges to the true centers.*

9.5 Applications

Clustering a mixture of Gaussians For a mixture of k Gaussians, we quote the suitable results without proof, as the proof is identical to the proof in [104]. We are given a mixture of k Gaussians, $F_1, \ldots, F_k$, where the standard deviation of each distribution in any direction is at most σ_r, and the weight of each distribution is w_r. We denote $\sigma_{\max} = \max_r\{\sigma_r\}$ and $w_{\min} = \min_r\{w_r\}$.

Theorem 9.15. *Suppose we are given a set of $n \gg \frac{d}{w_{\min}}$ samples from a mixture of k Gaussians, such that for every $r \neq s$ it holds that $\|\mu_r - \mu_s\| \geq c\sigma_{\max}\sqrt{\frac{k}{w_{\min}}}$ poly$\log\left(\frac{d}{w_{\min}}\right)$. Then w.h.p. these points satisfy the proximity condition.*

For Gaussians, the best known separation bound is Achlioptas and McSherry's bound [4] of $\Omega(\sigma_{\max}(w_{\min}^{-1/2} + \sqrt{k\log(k \cdot \min\{n, 2^k\})}\ ))$. As we assume k is large, this separation condition is $\tilde{\Omega}(\sigma_{\max}(w_{\min}^{-1/2} + \sqrt{k})) = \tilde{\Omega}(\sigma_{\max}/\sqrt{w_{min}})$. Therefore, the separation bound of Theorem 9.15 is $\sqrt{k}$ times worse than the best known bound. However, applying Kumar and Kannan's boosting technique (Section 7 in [104]), that replaces the polynomial dependency in $w_{\min}$ with a logarithmic one, we get:

Theorem 9.16. *Suppose we are given a set of $n \gg \frac{d}{w_{\min}}$ samples from a mixture of k Gaussians, such that for every $r \neq s$ it holds that*

$$\|\mu_r - \mu_s\| \geq c\sigma_{\max}\sqrt{k}\ \text{poly}\log\left(\frac{d}{w_{\min}}\right)$$

Then there exists an algorithm that w.h.p. correctly classifies all points.

Therefore, if for any r and r', both $\sigma_r \approx \sigma_{r'}$ and $w_r \approx w_{r'}$, then both [4] and Theorem 9.16 give roughly the same bound. If for any r and r' we have that $\sigma_r \approx \sigma_{r'}$, yet $w_{\min} \ll \frac{1}{k}$, then Theorem 9.16 provides a better bound. If for any r and r' we have that $w_r \approx w_{r'}$, yet the directional standard deviations of the distributions vary, then the bound of [4], in which the distance between any two cluster centers depends only the parameters of these two distributions, is the better bound. If both the standard deviations and the weights vary significantly between the different distributions, then better bound is determined on a case by case basis.

McSherry's Planted Partition Model. In the Planted Partition Model [110, 10, 11] our instance is a random n-vertex graph generated by using an implicit partition of the n

points into k clusters. There exists an unknown $k \times k$ matrix of probabilities P, and for every pair of vertices u, v there exists an edge connecting u and v w.p. P_{rs} (assuming u belongs to cluster r and v to cluster s). The goal here is to recover the partition of the points (thus – recover P). Viewing this graph as a $n \times n$ matrix, each row is taken from a special distribution F_r over $\{0,1\}^n$ – where each coordinate j is an independent Bernoulli r.v. with mean $P_{r,C(j)}$, denoting $C(j)$ as the cluster j belongs to. Thus, the mean of this distribution, μ_r, is a vector with its j-coordinate set to $P_{r,C(j)}$. Denote $w_{\min} = \min_r\{\frac{n_r}{n}\}$ and $\sigma_{\max} = \max_{r,s}\sqrt{P_{rs}}$. The result of [110] is that if for every $r \neq s$

$$\|\mu_r - \mu_s\| = \Omega\left(\sigma_{\max}\sqrt{k}\left(\frac{1}{w_{\min}} + \log(n/\delta)\right)\right) \tag{9.2}$$

then it is possible to retrieve the partition of the vertices w.p. at least $1-\delta$.

Kumar and Kannan were not able to match the distance bounds of McSherry, and required centers to be $\sqrt{k}$ factor greater then the bound of (9.2). Here we match the bound of McSherry exactly. Following the proof in Kumar-Kannan (with few changes), we prove:

Theorem 9.17. *Assuming that $\sigma_{\max} \geq \frac{3\log(n)}{n}$ and that the planted partition model satisfies equation 9.2 for every $r \neq s$, then w.p. at least $1-\delta$, every point satisfies the proximity condition.*

Proof. We follow the proof of Kumar-Kannan, making the suitable changes. McSherry (Theorem 10 of [110]) showed that w.h.p. $\|A - C\| \leq 4\sigma_{\max}\sqrt{n}$. So our goal is to show that, w.h.p., all points are $\sqrt{k}$-good. I.e., denoting u as a unit-length vector connecting μ_r and μ_s, we show that w.h.p. that for every $i \in C_r^*$ we have

$$|(A_i - \mu_r)\cdot u| = O(\sqrt{k}\sigma_{\max}\left(\frac{1}{w_{\min}} + \log(n/\delta)\right))$$

Observe $u = \frac{\mu_s - \mu_r}{\|\mu_s - \mu_r\|}$, and due to the special structure of the means in this model, we have that $(\mu_r - \mu_s)_j = P_{rt} - P_{st}$ where $j \in T_t$. It follows that

$$\|\mu_r - \mu_s\|^2 = \sum_{t=1}^{k} n_t (P_{rt} - P_{st})^2$$

We therefore have

$$|(A_i - \mu_r)\cdot u| \leq \frac{1}{\|\mu_r - \mu_s\|}\left(\sum_{t=1}^{k} |P_{rt} - P_{st}| \left|\sum_{j \in T_t} A_{ij} - P_{rt}\right|\right)$$

Observe, A_{ij} are i.i.d 0-1 random variables with mean P_{rt}, so we expect their sum to deviate from its expectation by no more than a few standard deviations. Indeed, Kumar and Kannan prove that w.h.p. it holds that for every t we have

$$\left|\sum_{j\in T_t} A_{ij} - P_{rt}\right| \leq B\sqrt{n_t}\sigma_{\max}\left(\frac{1}{w_{\min}} + \log(n/\delta)\right)$$

where B is some sufficiently large constant. This allows us to deduce that

$$\begin{aligned}|(A_i - \mu_r)\cdot u| &\leq B\sigma_{\max}\left(\frac{1}{w_{\min}} + \log(n/\delta)\right)\frac{\sum_{t=1}^k \sqrt{n_t}|P_{rt} - P_{st}|}{\sqrt{\sum_{t=1}^k n_t(P_{rt} - P_{st})^2}} \\ &\leq B\sqrt{k}\sigma_{\max}\left(\frac{1}{w_{\min}} + \log(n/\delta)\right)\end{aligned}$$

where the last inequality is simply the power-mean inequality. □

9.6 An Open Problem

Our work presents an algorithm which successfully clusters a dataset, provided that the distance between any two cluster centers meets a certain lower bound. We would like to point out one particular direction to improve this bound. Note that our center separation bound depends on $\|A - C\|$, a property of the entire dataset. It would be nice to handle the case where the separation condition between μ_r and μ_s depends solely on C_r^* and C_s^*. That is, if we define $\tilde{\Delta}_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A_r - C_r\|$, is it possible to successfully separate clusters s.t $\|\mu_r - \mu_s\| \geq c(\tilde{\Delta}_r + \tilde{\Delta}_s)$? We comment that most of our analysis (and particularly Lemma 9.11) builds only on the ratio between $\|\mu_r - \nu_r\|$ and $\|\mu_r - \mu_s\|$ – we assume the first is no greater than $\alpha\Delta_r$ and that the latter is no less than $c(\Delta_r + \Delta_s)$. In fact, one can revise the proofs of Theorems 9.5 and 9.6 so that they will hold based on this assumption alone (without using the properties of the SVD). The problem therefore boils down to finding *initial* centers $\{\nu_r\}$ that are sufficiently close to the true centers $\{\mu_r\}$, under the assumption that $\forall r \neq s,\ \|\mu_r - \mu_s\| \geq c(\tilde{\Delta}_r + \tilde{\Delta}_s)$. But this is an intricate task, mainly because such separation condition does *not* imply that $\{\mu_1, \mu_2, \ldots, \mu_k\}$ are the centers minimizing the k-means cost! (Nor do $\{\hat{\mu_1}, \hat{\mu_2}, \ldots, \hat{\mu_k}, \}$ minimize the k-means cost of $\hat{A}$.) Consider the case, for example, where cluster r has very few points (say $n_r = \sqrt{n}$) and very small variance, and cluster s is very big (say $n_s = n/5$), and is essentially composed of two sub-components with distance $\frac{1}{2\sqrt{n_s}}\|A_s - C_s\|$ between the

centers of the two sub-components. The k-means cost of placing two centers within C_s is smaller than placing one center at μ_s and one center at μ_r. This relates to the question of designing a t-approximation algorithm for k-means, guaranteeing that *each cluster's cost* cannot increase by more than a factor of t.

Bibliography

"What burns me up is that the answer is right here somewhere, staring us in the face."

[1] Facebook newsroom: Statistics. `http://newsroom.fb.com/Key-Facts`, 2012. Retrieved 6/29/2012. 6.4.2

[2] Ittai Abraham, Yair Bartal, T-H. Hubert Chan, Kedar Dhamdhere Dhamdhere, Anupam Gupta, Jon Kleinberg, Ofer Neiman, and Aleksandrs Slivkins. Metric embeddings with relaxed guarantees. In *Proc. 46th Annual IEEE Symp. Foundations of Computer Science (FOCS)*, 2005. 8.1.1

[3] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4), June 2003. 5.1.1, 5.5

[4] Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In *COLT*, 2005. 1.1.2, 4.3.1, 4.3.1, 9.1, 1, 9.2.4, 9.4, 9.5, 9.5

[5] Pankaj K. Agarwal and Cecilia Magdalena Procopiuc. Exact and approximation algorithms for clustering (extended abstract). In *SODA*, 1998. 4.1

[6] Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *APPROX-RANDOM*, 2009. 4.2.2

[7] Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, STOC '06, New York, NY, USA, 2006. ACM. 5.1

[8] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In *NIPS*, 2009. 4.2.2

[9] N. Alon, D. Moshkovitz, and S. Safra. Algorithmic construction of sets for k-restrictions. *ACM Transactions on Algorithms (TALG)*, 2(2):153–177, 2006. 6.4.2.2

[10] Noga Alon and Nabil Kahale. A spectral technique for coloring random 3-colorable graphs. In *SIAM Journal on Computing*, pages 346–355, 1994. 9.5

[11] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. pages 457–466, 1998. 9.5

[12] Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 1992. 2.3.2

[13] Sanjeev Arora and Ravi Kannan. Learning mixtures of arbitrary gaussians. In *STOC*, 2001. 1.1.2, 4.3.1, 9.1

[14] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for Euclidean k-medians and related problems. In *STOC*, 1998. 1

[15] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *SODA*, 2007. 4.2.2

[16] David Arthur, Bodo Manthey, and Heiko Roglin. k-means has polynomial smoothed complexity. *Foundations of Computer Science, Annual IEEE Symposium on*, 0, 2009. 4.2.3

[17] David Arthur and Sergei Vassilvitskii. Worst-case and smoothed analysis of the icp algorithm, with an application to the k-means method. In *FOCS*, 2006. 4.2.3

[18] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *STOC*, 2001. 4.1

[19] Pranjal Awasthi, Maria-Florina Balcan, Avrim Blum, Or Sheffet, and Santosh Vempala. On nash-equilibria of approximation-stable games. In *SAGT*, 2010. 4.2.3

[20] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Stability yields a ptas for k-median and k-means clustering. In *Proc. 51st Annual IEEE Symp. Foundations of Computer Science (FOCS)*, 2010. (document), 1.2.2, 1.3

[21] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Inf. Process. Lett.*, 112(1-2), 2012. (document), 1.2.2, 1.3

[22] Pranjal Awasthi and Or Sheffet. Improved spectral-norm bounds for clustering. In *APPROX-RANDOM*, pages 37–49, 2012. (document), 1.2.2, 1.3

[23] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*, pages 181–190. ACM, 2007. 6.1

[24] Mihai Bādoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via coresets. In *STOC*, 2002. 4.1

[25] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *SODA*, pages 1068–1077, 2009. 1.2.2, 1.3, 4.2.1, 4.5, 4.2.1, 4.2.4, 7.1, 7.1, 7.2, 7.2, 7.2.2, 7.2.4, 7.5, 8.1

[26] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1), 2006. 5.1, 5.1.1

[27] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *STOC*, 2008. 1.1.2, 8.1.1, 8.2.1, 8.2.2, 8.2.2

[28] Maria-Florina Balcan and Mark Braverman. Finding low error clusterings. In *COLT*, 2009. 4.2.1, 7.2.2, 7.3

[29] Maria Florina Balcan and Yingyu Liang. Clustering under perturbation resilience. In *ICALP*. Springer-Verlag, 2012. 4.2.3, 8.1.1, 8.3

[30] Maria-Florina Balcan, Heiko Röglin, and Shang-Hua Teng. Agnostic clustering. In *ALT*, 2009. 4.2.1, 7.2.2

[31] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008. 5.1

[32] Yair Bartal. On approximating arbitrary metrices by tree metrics. In *Proc. 30th Annual ACM Symp. Theory of Computing (STOC)*, 1998. 8.1.1

[33] Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-ramanujan sparsifiers. In *STOC*, 2009. 5.3.2.3

[34] P.S. Bearman and J. Moody. Suicide and friendships among american adolescents. *Journal Information*, 94(1), 2004. 6.5.1

[35] Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. *Computing Research Repository*, abs/1004.4:103–112, 2010. 4.3.1

[36] Shai Ben-David, Dávid Pál, and Hans-Ulrich Simon. Stability of k-means clustering. In *COLT*, 2007. 4.2.3

[37] Shai Ben-David, Ulrike von Luxburg, and Dvid Pl. A sober look at clustering stability. In *COLT*, Lecture Notes in Computer Science, 2006. 4.2.3

[38] András A. Benczúr and David R. Karger. Approximating s-t minimum cuts in $\tilde{o}(n^2)$ time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, STOC '96, pages 47–55, New York, NY, USA, 1996. ACM. 5.3.2.3

[39] Yonatan Bilu, Amit Daniely, Nati Linial, and Michael Saks. On the practically interesting instances of maxcut. In *STACS*, 2013. 4.2.3

[40] Yonatan Bilu and Nati Linial. Are stable instances easy? In *1st Symp. Innovations in Computer Science (ICS)*, 2010. 1.2.2, 1.3, 4.2.3, 4.2.3, 4.10, 4.2.4, 8.1, 8.1.1

[41] J. Blocki, A. Blum, A. Datta, and O. Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. In *53rd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2012. (document), 1.2.1, 1.3, 6.1.1

[42] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private analysis of social networks via restricted sensitivity. In *ITCS*, 2013. (document), 1.2.1, 1.3

[43] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618. ACM, 2008. 3.2.1, 3.6, 3.2.1, 14, 5.1.1, 5.3.2.3, 6.6

[44] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the sulq framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '05, pages 128–138, New York, NY, USA, 2005. ACM. 1.1.1, 5.1.1

[45] J Bourgain. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52(1-2):46–52, 1985. 5.1

[46] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer networks*, 33(1):309–320, 2000. 6.4.2

[47] S. Charles Brubaker and Santosh Vempala. Isotropic pca and affine-invariant clustering. In *FOCS*, 2008. 4.3.1, 9.1

[48] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *STOC*, 1999. 4.1

[49] Kamalika Chaudhuri and Satish Rao. Beyond gaussians: Spectral methods for learning mixtures of heavy-tailed distributions. In *COLT*, 2008. 4.3.1

[50] Kamalika Chaudhuri and Satish Rao. Learning mixtures of product distributions using correlations and independence. In *COLT*, 2008. 4.3.1, 9.1

[51] Shuchi Chawla, Cynthia Dwork, Frank Mcsherry, Adam Smith, and Larry Joseph Stockmeyer. Toward privacy in public databases. In *In TCC*, pages 363–385, 2005. 1.1.1

[52] William W. Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *KDD*, pages 475–480, 2002. 4.1

[53] Amin Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Comb. Probab. Comput.*, 19:227–284, 2010. 1

[54] Anirban Dasgupta, John Hopcroft, Ravi Kannan, and Pradipta Mitra. Spectral clustering with limited independence. In *SODA*, 2007. 4.3.1, 9.1

[55] Anirban Dasgupta, Ravi Kumar, and Tamás Sarlos. A sparse johnson: Lindenstrauss transform. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC '10, pages 341–350, New York, NY, USA, 2010. ACM. 5.1.1, 5.5

[56] Sanjoy Dasgupta. Learning mixtures of gaussians. In *FOCS*, 1999. 1.1.2, 4.3.1, 4.3.1, 9.1

[57] Sanjoy Dasgupta. The hardness of k-means clustering. Technical report, University of California at San Diego, 2008. 4.1

[58] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Struct. Algorithms*, 22(1), January 2003. 5.1.1

[59] Sanjoy Dasgupta and Leonard Schulman. A probabilistic analysis of em for mixtures of separated, spherical gaussians. *J. Mach. Learn. Res.*, 2007. 4.3.1, 4.11, 9.1

[60] W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation schemes for clustering problems. In *STOC*, 2003. 4.1

[61] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *PODS*, 2003. 1.1.1

[62] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, 2006. 1.1.1, 3.1, 3.3, 6.1

[63] Cynthia Dwork, Frank Mcsherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *In Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284. Springer, 2006. 1.1.1, 1.1.1, 3, 3.2, 3.3, 6.1

[64] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, 2009. 3.2.1

[65] Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *FOCS*, 2010. 3.4, 0, 5.3.2.1, 0, 5.4.2.1

[66] D. Easley and J. Kleinberg. *Networks, crowds, and markets*. Cambridge Univ Press, 2010. 6.1, 6.1.1, 6.4.2, 6.5.1

[67] Michelle Effros and Leonard J. Schulman. Deterministic clustering with data nets. *ECCC*, 50, 2004. 4.1

[68] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *JACM*, 45:314–318, 1998. 8.2.3

[69] P. Frankl and H. Maehara. The johnson-lindenstrauss lemma and the sphericity of some graphs. *J. Comb. Theory Ser. A*, 44(3), June 1987. 5.1.1

[70] L.C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979. 6.5.1

[71] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990. 4.2.3, 8.1

[72] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6), 1995. 4.2.3

[73] Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996. 9.2.1

[74] Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38, 1985. 4.1

[75] M.S. Granovetter. *Getting a job: A study of contacts and careers*. University of Chicago Press, 1995. 6.5.1

[76] Sudipto Guha and Samir Khuller. Greedy strikes back: Improved facility location algorithms. In *Journal of Algorithms*, 1998. 4.1

[77] Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *TCC*, 2012. 3.2.2, 3.8, 14, 5.1.1, 5.3.2, 5.3.2.2, 5.3.2.4, 5.4.2.2

[78] Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. 5.1.1

[79] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, 2004. 1, 4.1

[80] M. Hardt and G.N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 61–70. IEEE, 2010. 3.2.3, 3.8, 5.3.2.4, 6.6

[81] Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. *CoRR*, abs/1012.4763, 2010. 14

[82] Moritz Hardt and Aaron Roth. Beating randomized response on incoherent matrices. In *STOC*, 2012. 1, 5.1.1, 5.2

[83] John A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 99th edition, 1975. 4.1.1

[84] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM*, pages 169–178, 2009. 5.1.1, 6.1.1

[85] Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3), 1986. 4.1

[86] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990. 2.2.2

[87] W. L. Hsu and G. L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1, 1979. 4.1

[88] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *Proc. 10th Symp. Comp. Geom.*, 1994. 7.2.3, 7.4.1

[89] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*. ACM, 1998. 5.1

[90] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems (extended abstract). In *STOC*, 2002. 1.1.2, 4.1

[91] Ragesh Jaiswal and Nitin Garg. Analysis of k-means++ for separable data. In *APPROX-RANDOM*, 2012. 4.2.2

[92] C. Jernigan and B.F.T. Mistree. Gaydar: Facebook friendships expose sexual orientation. *First Monday*, 14(10), 2009. 6.1

[93] W. Johnson and J. Lindenstauss. Extensions of Lipschitz maps into a Hilbert space. *Contemporary Mathematics*, 1984. 5.1, 5.3

[94] Petri Juutinen. Absolutely minimizing lipschitz extensions on a metric space. *Annales Academiae Scientiarum Fennicae. Mathematica*, 27:57–67, 2002. 6.4.1

[95] Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two gaussians. In *STOC'10*, pages 553–562, 2010. 4.3.1

[96] Ravindran Kannan, Hadi Salmasian, and Santosh Vempala. The spectral method for general mixture models. *SIAM J. Comput.*, 2008. 4.3.1, 9.1

[97] Ravindran Kannan and Santosh Vempala. Spectral algorithms. *Found. Trends Theor. Comput. Sci.*, March 2009. 9.2.1, 9.2.2, 9.2

[98] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Proc. 18th Symp. Comp. Geom.*, 2002. 1.1.2, 4.1

[99] Michael Kapralov and Kunal Talwar. On differentially private low rank approximation. In *SODA*, 2013. 5.1.1

[100] Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *PVLDB*, 4(11), 2011. 5.1.1, 5.2, 6.1.1

[101] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS, 2008. 3.2.2

[102] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In *TCC*, 2013. 6.1.1

[103] Jon M. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *STOC*, 1997. 5.1

[104] A. Kumar and R. Kannan. Clustering with spectral norm and the k-means algorithm. In *FOCS*, 2010. 1.2.2, 4.3.3, 9.1, 9.1.1, 9.2.2, 9.2.2, 9.4, 9.4, 9.5, 9.5

[105] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time (1+ ϵ)-approximation algorithm for k-means clustering in any dimensions. In *FOCS*, 2004. 4.1, 7.3, 7.4.3

[106] Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In *STOC*. ACM, 2013. 4.1

[107] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, SFCS '94, pages 577–591, Washington, DC, USA, 1994. IEEE Computer Society. 5.1

[108] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2), 1994. 5

[109] Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Bilulinial stable instances of max cut. *CoRR*, abs/1305.1681, 2013. 4.2.3

[110] F. McSherry. Spectral partitioning of random graphs. In *FOCS*, 2001. 1.2.2, 4.3.2, 9.1, 1, 9.1.1, 9.2.2, 9.1, 9.5, 9.5

[111] Frank McSherry and Ilya Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *KDD*, pages 627–636, 2009. 1, 5.1.1

[112] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007. 3.2.1, 3.5, 5.3.2.3

[113] Marina Meilă. The uniqueness of a good optimum for k-means. In *ICML*, 2006. 4.2.3

[114] Matús Mihalák, Marcel Schöngens, Rastislav Srámek, and Peter Widmayer. On the complexity of the metric tsp under stability considerations. In *SOFSEM*, 2011. 4.2.3

[115] K.S. Miller. *Multidimensional Gaussian distributions*. SIAM series in applied mathematics. Wiley, 1964. 2.3.1.3

[116] Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of gaussians. In *FOCS'10*, 2010. 4.3.1

[117] A G Murzin, S E Brenner, T Hubbard, and C Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247(4):536–540, 1995. 4.1

[118] M.E.J. Newman. The structure and function of complex networks. *SIAM review*, pages 167–256, 2003. 6.4.2, 6.5.1

[119] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, pages 75–84. ACM, 2007. Full version in: `http://www.cse.psu.edu/~asmith/pubs/NRS07`. 3.3, 3.10, 3.11, 5.1.1, 5.2, 6.1, 6.1.1, 6.4.2.2

[120] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric k-clustering. In *FOCS*, 2000. 4.1

[121] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *FOCS*, 2006. 1.2.2, 1.3, 4.2.2, 4.7, 4.2.4, 7.1, 7.1, 7.2, 7.2, 7.2.1, 7.2.4, 7.3, 7.4.1, 7.5, 8.1, 9.1, 9.1.1, 9.1.1, 9.3.1

[122] Mallesh Pai and Aaron Roth. Privacy and mechanism design. *SIGecom Exchanges*, 12(1), 2013. 3.2.1

[123] Christos H. Papadimitriou, Prabhakar Raghavan, Hisao Tamaki, S. Vempala, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *PODS*. ACM press, 1998. 5.1

[124] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proc. 40th Annual ACM Symp. Theory of Computing (STOC)*, 2008. 8.1.1

[125] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: output perturbation for queries with joins. In *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 107–116. ACM, 2009. 6.1.1

[126] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 475–484. ACM, 1997. 6.4.2.2

[127] Lev Reyzin. Data stability in clustering: A closer look. *CoRR*, 2011. 4.2.3, 8.1.1, 8.3

[128] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 765–774. ACM, 2010. 3.2.3

[129] Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *FOCS*, 2006. 5.1

[130] F. Schalekamp, M. Yu, and A. van Zuylen. Clustering with or without the Approximation. In *COCOON*, 2010. 4.2.1

[131] Leonard J. Schulman. Clustering for edge-cost minimization (extended abstract). In *STOC*, 2000. 5.1

[132] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *STOC*, 2008. 5.3.2.3

[133] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC*, 2004. 5.3.2.3

[134] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the facebook social graph. *Arxiv preprint arXiv:1111.4503*, 2011. 6.1, 6.4.2

[135] Jonathan Ullman. Answering $n^{\{2+o(1)\}}$ counting queries with differential privacy is hard. In *STOC*, 2013. 3.2.1

[136] Jonathan Ullman and Salil P. Vadhan. Pcps and the hardness of generating private synthetic data. In *TCC*, 2011. 3.2.1

[137] Andrea Vattani. k-means requires exponentially many iterations even in the plane. In *Symposium on Computational Geometry*, 2009. 4.2.3

[138] Santosh Vempala and Grant Wang. A spectral algorithm for learning mixtures of distributions. In *Journal of Computer and System Sciences*, 2002. 1.1.2, 4.3.1, 9.1

[139] S.S. Vempala. *The Random Projection Method*. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 2005. 5.1, 5.1.1

[140] Konstantin Voevodski, Maria Florina Balcan, Heiko Roglin, ShangHua Teng, and Yu Xia. Efficient clustering with limited distance information. In *Proc. 26th UAI*, 2010. 7.5

[141] Stanley L. Warner. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association*, 60(309):63+, March 1965. 3.2.2

[142] D.J. Watts and S.H. Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442, 1998. 6.5.1