2012 IEEE 53rd Annual Symposium on Foundations of Computer Science

The Johnson-Lindenstrauss Transform Itself Preserves Differential Privacy

Jeremiah Blocki, Avrim Blum, Anupam Datta, Or Sheffet
Carnegie Mellon University
{jblocki@cs, avrim@cs, danupam@andrew, osheffet@cs}.cmu.edu

***Abstract*—This paper proves that an "old dog", namely – the classical Johnson-Lindenstrauss transform, "performs new tricks" – it gives a novel way of preserving differential privacy. We show that if we take two databases, D and D', such that (i) $D' - D$ is a rank-1 matrix of bounded norm and (ii) all singular values of D and D' are sufficiently large, then multiplying either D or D' with a vector of iid normal Gaussians yields two statistically close distributions in the sense of differential privacy. Furthermore, a small, deterministic and *public* alteration of the input is enough to assert that all singular values of D are large.**

We apply the Johnson-Lindenstrauss transform to the task of approximating cut-queries: the number of edges crossing a $(S, \bar{S})$-cut in a graph. We show that the JL transform allows us to *publish a sanitized graph* that preserves edge differential privacy (where two graphs are neighbors if they differ on a single edge) while adding only $O(|S|/\epsilon)$ random noise to any given query (w.h.p). Comparing the additive noise of our algorithm to existing algorithms for answering cut-queries in a differentially private manner, we outperform all others on small cuts ($|S| = o(n)$).

We also apply our technique to the task of estimating the variance of a given matrix in any given direction. The JL transform allows us to *publish a sanitized covariance matrix* that preserves differential privacy w.r.t bounded changes (each row in the matrix can change by at most a norm-1 vector) while adding random noise of magnitude independent of the size of the matrix (w.h.p). In contrast, existing algorithms introduce an error which depends on the matrix dimensions.

I. Introduction

The celebrated Johnson Lindenstrauss transform [22] is widely used across many areas of Computer Science. A very non-exhaustive list of related applications include metric and graph embeddings, computational speedups, machine learning, information retrieval, nearest-neighbor search, and compressed sensing. This paper unveils a new application of the Johnson Lindenstrauss transform – it also preserves 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* [11] 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 (see formal definition in Section II). By itself, preserving differential privacy is not 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*.

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. This simple technique 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 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. This work takes 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 proving privacy guarantees is non-trivial.

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 is 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

Supported in part by the National Science Foundation under grants CCF-1101215, CCF-1116892, the NSF Science and Technology Center TRUST, and an NSF Graduate Fellowship, as well as by the MSR-CMU Center for Computational Thinking.

[1]This notion of neighboring inputs, also considered in [24], [18], is weaker than allowing any individual to change her attributes arbitrarily, but is natural in a graph or matrix context.

0272-5428/12 $26.00 © 2012 IEEE
DOI 10.1109/FOCS.2012.67

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 II), 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 users wish to know how many edges cross a $(S, \bar{S})$-cut. Such a query can be formalized by the length of the product $E_G \mathbf{1}_S$, where E_G is the *edge-matrix* of G and $\mathbf{1}_S$ is the indicator vector of S (see Section II). We prove that the JL transform allows us to publish a perturbed Laplacian of G while preserving (ϵ, δ)-differential privacy, w.r.t two graphs being neighbors if they differ only on a single edge. Comparing our 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 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. After all, if we multiply the edge matrix of a graph G with a vector of iid normal Gaussians, we get a vector with each entry distributed like a Gaussian; and if we replace G with a neighboring G', we affect only two entries in this vector. Presumably, applying the previous analysis to both entries suffices to prove we preserve differential privacy. But 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 $E_G \mathbf{1}_S$ is the zero-vector, and no matter what random projection we pick, $Y^\intercal E_G \mathbf{1}_S = 0$. In contrast, by adding a single edge that crosses the $(S, \bar{S})$-cut, we get a graph G' s.t. $\mathbf{Pr}[Y^\intercal E_{G'} 1_S \neq 0] = 1$.

Organization. Next we detail related work. Section II details important notations and preliminaries. In Sections III and IV we convert the above univariate intuition to the multivariate Gaussian case. Section III describes our results for graphs and cut-queries, and Section IV details the result for directional queries (the general case). Due to space limitations, the comparison of our algorithms with existing algorithms is deferred to the full version of this work [3]. Even though there are clear similarities between the analyses in Sections III and IV, 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 is 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 V concludes

[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 III and IV, the idea of changing the input will be the key ingredient in getting noise bounds that are independent of n and d.

with a discussion and open problems.

A. Related Work

Differential privacy was developed through a series of papers [8], [11], [6], [4]. Dwork et al [11] gave the first formal definition and the description of the basic Laplace mechanism. Its Gaussian equivalent was defined in [10]. Other mechanisms for preserving differential privacy include the Exponential Mechanism of McSherry and Talwar [25], [5]; the recent Multiplicative Weights mechanism of Hardt and Rothblum [19] and its various extensions [17], [14], [15]; the Median Mechanism [28] and a boosting mechanism of Dwork et al [12]. In addition, the classical Randomized Response (see [30]) preserves differential privacy as discussed in recent surveys [13], [9]. The task of preserving differential privacy when the given database is a graph or a social network was studied by Hay et al [20] 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 [27] (see full version) studied the case of estimating the number of triangles in a graph, and Karwa et al [23] extended this result to other graph structures. Gupta et al [15] 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 [4] approximate the input matrix with the PCA construction by adding $O(d^2)$ noise to the input. The work of McSherry and Mironov [24] (inspired by the Netflix prize competition) 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. The work of Hardt and Roth [18] gives a low-rank approximation of a given input matrix while adding $\min\{\sqrt{d},\sqrt{n}\}$ noise by following the elegant framework of Halko et al [16]. According to [18], a recent and not-yet-published work of Kapralov, McSherry and Talwar preserves rank-1 approximations of a given PSD matrix with error $O(n)$.

The body of work on the JL transform is by now so extensive that only a book may survey it properly [29]. 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 [5] gave an algorithm that outputs a sanitized dataset for learning large-margin classifiers by appealing to JL related results of [1]. Hardt and Roth [18] gave a privacy preserving version of an algorithm of [16] 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.

II. 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 [27], [23], 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 [18], 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.

Definition II.1. *An algorithm ALG which maps inputs into some range $\mathcal{R}$ maintains* (ϵ,δ)-differential privacy *if for all pairs of neighboring inputs $\mathcal{I},\mathcal{I}'$ and for all subsets $\mathcal{S} \subset \mathcal{R}$ it holds that*

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

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 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 II.2. *We say an algorithm ALG gives a (η,τ,ν)-approximation for cut queries, if for every nonempty S w.p. $\geq 1-\nu$ we have that*

$$[(1-\eta)\Phi_G(S) - \tau \leq \mathsf{ALG}(S) \leq (1+\eta)\Phi_G(S) + \tau]$$

For $n \times d$ matrices, we are interested in *directional variance queries*: given a unit-length direction x, we wish to know what is the variance of A along the x direction: $\Phi_A(x) = x^{\intercal}A^{\intercal}Ax$. (Our algorithm normalizes A s.t. the mean of its n rows is 0.)

Definition II.3. *We say an algorithm ALG gives a (η,τ,ν)-approximation for directional variance queries, if for every unit-length vector x w.p. $\geq 1-\nu$ we have that*

$$[(1-\eta)\Phi_A(x) - \tau \leq \mathsf{ALG}(x) \leq (1+\eta)\Phi_A(x) + \tau]$$

Some Linear Algebra. 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 $\tilde{\det}(M) = \prod_{i=1}^{rank(M)} \sigma_i(M)$. 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, see [21].

Gaussian distribution. Given a r.v. X, we denote by $X \sim \mathcal{N}(\mu, \sigma^2)$ the fact that X has normal distribution with mean μ and variance σ^2. Recall that $\mathsf{PDF}_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-(x-\mu)^2/2\sigma^2)$. We repeatedly apply 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 $R' \sim \mathcal{N}(0, 1)$. Classic concentration bounds on Gaussians give that $Pr[|x - \mu|^2 > \log(1/\delta)\sigma^2] \leq 2\delta$.

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 [26].

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

Theorem II.4 (The Johnson Lindenstrauss transform [22])**.** *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 $\forall x \in \mathbb{R}^m$.*

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

Laplacians and edge-matrices. An undirected weighted graph $G = (V(G), E(G))$ can be represented in various ways. One representation is by the *adjacency matrix* A, where $A_{u,v} = w_{u,v}$. Another way is by the $\binom{n}{2} \times n$ *edge matrix* of the graph, E_G. We assume that the vertices of G are ordered arbitrarily, and for each pair of vertices $\{u, v\}$ where $u < v$, there exists a row in E_G. The entries of E_G are $\left(E_G\right)_{(\{u,v\},x)} = \begin{cases} \sqrt{w_{u,v}}, & \text{if } u \sim_G v \text{ and } x = u \\ -\sqrt{w_{u,v}}, & \text{if } u \sim_G v \text{ and } x = v \\ 0, & \text{o/w} \end{cases}$. where $u \sim_G v$ denotes that (u, v) is an edge in G. Alternatively, one can represent G using the *Laplacian* of the graph $L_G = E_G^\intercal E_G$. Formally, the matrix L_G is the matrix whose diagonal entries are $(L_G)_{u,u} = \sum_{x \sim_G u} w_{x,u}$ and non diagonal entries are $(L_G)_{u,v} = -w_{u,v}$. It is simple to verify that for any x, the following equality holds: $x^\intercal L_G x = \sum_{u \sim_G v} w_{u,v}(x_u - x_v)^2$. As a corollary, if we take any nonempty $S \subsetneq V(G)$ and denote its $\{0,1\}^n$-indicator vector as $\mathbf{1}_S$, then $\mathbf{1}_S^\intercal\ L_G\ \mathbf{1}_S = \|E_G\mathbf{1}_S\|^2 = \sum_{u \in S, v \notin S} w_{u,v} = \Phi_G(S)$.

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: $L_{a,b} = e_{a,b}e_{a,b}^\intercal = E_{a,b}^\intercal E_{a,b}$.

III. Publishing a Perturbed Laplacian

A. The Johnson-Lindenstrauss Algorithm

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

Theorem III.1. *Algorithm 1 preserves (ϵ, δ)-differential privacy w.r.t to edge changes in G.*

Algorithm 1: 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 2: Approximating $\Phi_G(S)$

Input: A non empty $S \subsetneq V(G)$, parameters n, w and Laplacian $\tilde{L}$ from Algorithm 1.

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 III.2. *For every $\eta, \nu > 0$ and a nonempty S of size s, Algorithm 2 gives a (η, τ, ν)-approximation for cut queries, for $\tau = O\big(s \cdot \frac{\sqrt{\ln(1/\delta)\ln(1/\nu)}}{\epsilon}\left(\ln(1/\delta) + \ln(\ln(1/\nu)/\eta^2)\right)\big)$.*

Clearly, once Algorithm 1 publishes $\tilde{L}$, any user interested in estimating $\Phi_G(S)$ for some nonempty $S \subsetneq V(G)$ can run Algorithm 2 on her own. We comment that $\tilde{L}$ is a Laplacian of graph which might have negative edge-weight. Also, observe that w is independent of n, which we think of as large number, so we assume throughout the proofs of both theorems that both $\frac{w}{n}, \frac{1}{w}$ are $< 1/2$. The proof of Theorem III.2 is immediate from the JL lemma. Since it is no more than a mere computation we omit it, deferring the interested reader to the full version of this paper [3].

Comment. The guarantee of Theorem III.2 is not to be mistaken with a weaker guarantee of providing a good approximation to *most* cut-queries. Theorem III.2 guarantees that any set of k predetermined cuts is well-approximated by Algorithm 2, assuming Algorithm 1 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). After all, most cuts are concentrated about the expectation.

We turn our attention to the proof of Theorem III.1. 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 is not present in G, and in the proof of Theorem III.1, we identify G with the manipulation Algorithm 1 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 1 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 III.1 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 III.3. *Set $\epsilon_0 = \frac{\epsilon}{\sqrt{4r\ln(2/\delta)}}, \delta_0 = \frac{\delta}{2r}$. Then,*

$$\forall x, \quad \mathsf{PDF}_{E_G^\intercal Y}(x) \leq \quad e^{\epsilon_0}\mathsf{PDF}_{E_{G'}^\intercal Y}(x) \tag{1}$$

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

$$\mathbf{Pr}[S] \geq 1 - \delta_0 \tag{2}$$

Proof of Theorem III.1 based on Claim III.3: Apply the composition theorem of [12] for r iid samples each preserving (ϵ_0, δ_0)-differential privacy. ∎

To prove Claim III.3, 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 III.3.

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 III.4. *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 III.3 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 allows us to deduce the following fact. Its and other facts' proofs appear in the full version of this work [3].

Fact III.5. *Since $L_G \preceq L_{G'}$ then for every i we have that $\lambda_i^2 \geq \sigma_i^2$.*

In addition, since Algorithm 1 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 III.5 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

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

Another fact we can deduce from $L_G \preceq L_{G'}$, is the following.

Fact III.6. *Since the kernels of L_G and of $L_{G'}$ are identical, then for every x it holds that $x^\intercal L_{G'}^\dagger x \leq x^\intercal L_G^\dagger x$. Symbolically, $L_G \preceq L_{G'} \Rightarrow L_{G'}^\dagger \preceq L_G^\dagger$.*

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

Proof of Claim III.3: We first prove the upper bound in (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 $\forall i, \sigma_i^2 \geq w > \epsilon_0^{-1}$ it holds that

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

We now turn to the lower bound of (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.

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

Therefore, if we show that

$$\mathbf{Pr}_{x\sim E_G^\intercal Y}\left[x^\intercal L_G^\dagger e_{a,b} \ \cdot \ e_{a,b}^\intercal L_{G'}^\dagger x > \frac{2}{1-\frac{w}{n}}\epsilon_0\right] < \delta_0 \quad (3)$$

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

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

which proves the lower bound of (2). We turn to proving (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 $E_{G'}$.

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

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 (3) and of Claim III.3. ■

To conclude this section, we attach a summarized comparison between our own technique and various other techniques in Table I. Due to space limitations, we do not elaborate further on this comparison, and refer the interested reader to the full version of this paper [3].

IV. Publishing a Covariance Matrix

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. We detail the algorithms here, but prove their privacy and utility in Appendix A, along with comparing Algorithm 3 to existing algorithms.

Theorem IV.1. *Algorithm 3 preserves (ϵ, δ)-differential privacy.*

Theorem IV.2. *Algorithm 4 is a (η, τ, ν)-approximation for directional variance queries, where $\tau = O\left(\frac{\ln(1/\delta)\ln(1/\nu)}{\epsilon^2\eta}\ln^2\left(\frac{\ln(1/\nu)}{\delta\eta^2}\right)\right)$.*

Again, the proof of Theorem IV.2 is straight-forward, and so it is deferred to the full version of this paper [3].

Method	Additive Error for any k	Additive Error for all Cuts	Multi-plicative Error?	Inter-active?	Tract-able?	Comments
Laplace Noise [11]	$O(\sqrt{k}/\epsilon)$	$O(2^{n/2}\epsilon)$	✗	✓	✓	
Randomized Response	$O(\sqrt{sn\log(k)}/\epsilon)$	$O(n\sqrt{s}/\epsilon)$	✗	✗	✓	Can be distributed; answers (S,T)-cut queries
Exponential Mecha-nism [25], [5]	$O(n\log(n)/\epsilon)$	$O(n\log(n)/\epsilon)$	✓	✗	✗	Error ind. of k
MW [19] IDC [15]	$\tilde{O}(\sqrt{\lvert E\rvert}\log(k)/\epsilon)$ $\tilde{O}(\sqrt{\lvert E\rvert}\log(k)/\epsilon)$	$\tilde{O}(n\sqrt{\lvert E\rvert}/\epsilon)$ $\tilde{O}(\sqrt{n\lvert E\rvert}/\epsilon)$	✗	✓	✓	Answers (S,T)-cut queries
JL	$O(s\sqrt{\log(k)}/\epsilon)$	$\tilde{O}(s\sqrt{n}/\epsilon)$	✓	✗	✓	Can be distributed

Table I
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.

Comment. We wish to clarify that Theorem IV.2 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 V.)

Comment. Comparing Algorithms 1 and 3, 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.

We conclude this section as well with a comparison of our technique to other techniques, summarize in Table II. As before, the details of the comparison can be found in the full version of this paper [3].

Algorithm 3: 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** $\tilde{C} = \frac{1}{r}A^\intercal M^\intercal M A$.

V. 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 paper. Below we detail a few of the open questions we find most compelling.

Error dependency 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

Algorithm 4: Approximating $\Phi_A(x)$

Input: A unit-length vector x, parameter w and a Covariance matrix $\tilde{C}$ from Algorithm 3.

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

Method	Additive Error	Multi-plicative Error?	Inter-active?	Tract-able?
Laplace Noise [11]	$O(\sqrt{k}/\epsilon)$	✗	✓	✓
Randomized Response	$\tilde{O}(\sqrt{d\log(k)}/\epsilon)$	✗	✗	✓
MW [19] IDC [15]	$\tilde{O}(d\sqrt{n}\log(k)/\epsilon)$ $\tilde{O}(d\sqrt{n\log(k)}/\epsilon)$	✗	✓	✓
JL	$O(\log(k)/\epsilon^2)$	✓	✗	✓

Table II
COMPARISON BETWEEN MECHANISMS FOR ANSWERING DIRECTIONAL VARIANCE QUERIES.

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.

Other Versions of JL. The analysis in this works deals with the most basic JL transform, using normal Gaussians. We believe that qualitatively the same results should apply for other versions of the JL transform (e.g., with entries taken in $U_{[-1,1]}$). However, we are not certain whether the same results hold for *sparse* transforms (see [7]).

ACKNOWLEDGMENTS

We would like to the thank the anonymous referees for their helpful suggestions.

REFERENCES

[1] M.-F. Balcan, A. Blum, and S. Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1):79–94, 2006.

[2] R. Bhatia. *Perturbation Bounds for Matrix Eigenvalues (Classics in Applied Mathematics)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2007.

[3] J. Blocki, A. Blum, A. Datta, and O. Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. *CoRR*, abs/1204.2136, 2012.

[4] A. Blum, C. Dwork, F McSherry, and K. Nissim. Practical privacy: the sulq framework. PODS '05, pages 128–138. ACM, 2005.

[5] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, pages 609–618. ACM, 2008.

[6] S. Chawla, C. Dwork, F. Mcsherry, A. Smith, and L. Stockmeyer. Toward privacy in public databases. In *In TCC*, pages 363–385, 2005.

[7] Anirban Dasgupta, Ravi Kumar, and Tamás Sarlos. A sparse johnson: Lindenstrauss transform. STOC '10, pages 341–350. ACM, 2010.

[8] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210, 2003.

[9] C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, 2011.

[10] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[11] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284. Springer, 2006.

[12] C. Dwork, G. Rothblum, and S. Vadhan. Boosting and differential privacy. In *FOCS*, 2010.

[13] C. Dwork and A. Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):2, 2010.

[14] A. Gupta, M. Hardt, A. Roth, and J. Ullman. Privately releasing conjunctions and the statistical query barrier. In *STOC*, pages 803–812, 2011.

[15] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. In *TCC*, pages 339–356, 2012.

[16] N. Halko, P-G. Martinsson, and J. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

[17] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. *CoRR*, abs/1012.4763, 2010.

[18] M. Hardt and A. Roth. Beating randomized response on incoherent matrices. In *STOC*, 2012.

[19] M. Hardt and G.N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70. IEEE, 2010.

[20] M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM*, pages 169–178, 2009.

[21] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[22] W. Johnson and J. Lindenstauss. Extensions of Lipschitz maps into a Hilbert space. *Contemporary Mathematics*, 1984.

[23] V. Karwa, S. Raskhodnikova, A. Smith, and G. Yaroslavtsev. Private analysis of graph structure. *PVLDB*, 4(11), 2011.

[24] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *KDD*, pages 627–636, 2009.

[25] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.

[26] K.S. Miller. *Multidimensional Gaussian distributions*. SIAM series in applied mathematics. Wiley, 1964.

[27] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, pages 75–84. ACM, 2007. Full version in: http://www.cse.psu.edu/~asmith/pubs/NRS07.

[28] A. Roth and T. Roughgarden. Interactive privacy via the median mechanism. In *STOC*, pages 765–774. ACM, 2010.

[29] S. Vempala. *The Random Projection Method*. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 2005.

[30] S. Warner. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association*, 60(309), March 1965.

APPENDIX A.
COVARIANCE MATRICES: MISSING PROOF

Proof of Theorem IV.1: 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 $E = e_i v^\mathsf{T}$ (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 $\{1, 0, \ldots, 0\}$.[3]

[3]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 $\tilde{e_i}^\mathsf{T} v$ 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.

The proof of the theorem is composed of two stages. The first stage is the simpler one. We ignore step 4 of Algorithm 3 (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 III.1, the proof follows from the assumption that Algorithm 3 outputs $O^\mathsf{T} = A^\mathsf{T} M$ (which clearly allows us to reconstruct $\tilde{C} = \frac{1}{r} O^\mathsf{T} O$). Again O^T is composed of r columns each is an iid sample from $A^\mathsf{T} Y$ where $Y \sim \mathcal{N}(0, I_{n\times n})$. We now give the analogous claim to Claim III.3.

Claim A.1. *Fix $\epsilon_0 = \frac{\epsilon}{\sqrt{4r\ln(2/\delta)}}$ and $\delta_0 = \frac{\delta}{2r}$. Denote $S = \{x : e^{-\epsilon_0}\mathsf{PDF}_{A'^\mathsf{T}Y}(x) \le PDF_{A^\mathsf{T}Y}(x) \le e^{-\epsilon_0}\mathsf{PDF}_{A'^\mathsf{T}Y}(x)\}$. Then $\mathbf{Pr}[S] \ge 1 - \delta_0$.*

Again, the composition theorem of [12] along with the choice of r gives that overall we preserve (ϵ, δ)-differential privacy. ■

Proof of Claim A.1: The proof mimics the proof of Claim III.3, but there are two subtle differences. First, the problem is simpler notation-wise, because A and A' both have full rank due to Algorithm 3. 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 $A^\mathsf{T} Y$ and $A'^\mathsf{T} Y$ are linear transformations of $\mathcal{N}(0, I_{n\times n})$.

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

Our proof proceeds as follows. First, we show

$$e^{-\epsilon_0/2} \le \sqrt{\frac{\det(A'^\mathsf{T}A')}{\det(A^\mathsf{T}A)}} \le e^{\epsilon_0/2} \tag{4}$$

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

$$\mathbf{Pr}_x \left[\frac{1}{2} \left| x^\mathsf{T} \left((A^\mathsf{T}A)^{-1} - (A'^\mathsf{T}A')^{-1} \right) x \right| \ge \epsilon_0/2 \right] \le \delta_0 \tag{5}$$

Clearly, combining both (4) and (5) proves the claim.

Let us prove (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 $\sigma_1 \ge \sigma_2 \ge \ldots \ge \sigma_d > 0$ and the singular values of A' are $\lambda_1 \ge \lambda_2 \ge \ldots \ge \lambda_d > 0$. Therefore we have $A^\mathsf{T}A = V\Sigma^2 V^\mathsf{T}$, $A'^\mathsf{T}A' = V'\Lambda^2 V'^\mathsf{T}$ 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 is not sufficient to use the trace of the matrices. Instead, we invoke an application of Lindskii's theorem (Theorem 9.4 in [2]).

Fact A.2 (Linskii)**.** *For every k and every $1 \le i_1 < i_2 < \ldots < i_k \le n$ we have that $\sum_{j=1}^k \lambda_{i_j} \le \sum_{j=1}^k \sigma_{i_j} + \sum_{i=1}^k sv_i(E)$ 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 $\sum_{i\in Big} \lambda_i - \sigma_i \le 1$.

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 (4) is straight-forward: $\sqrt{\prod_i \frac{\lambda_i^2}{\sigma_i^2}}\leq \prod_{i\in Big}\left(1+\frac{\lambda_i-\sigma_i}{\sigma_i}\right)\leq e^{\left(\frac{1}{w}\sum_{i\in Big}\lambda_i-\sigma_i\right)}\leq e^{\frac{1}{w}}$ and similarly, $\sqrt{\prod_i \frac{\sigma_i^2}{\lambda_i^2}}\leq e^{\frac{1}{w}}\leq e^{\epsilon_0/2}$.

We turn to proving (5). We start with the following derivation.

$$\begin{aligned} & x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}x-x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x \\ = \;& x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}(A'^{\mathsf{T}}A')(A'^{\mathsf{T}}A')^{-1}x-x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x \\ = \;& x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}((A+E)^{\mathsf{T}}(A+E))(A'^{\mathsf{T}}A')^{-1}x \\ & -x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x \\ = \;& x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}(A^{\mathsf{T}}E+E^{\mathsf{T}}A')(A'^{\mathsf{T}}A')^{-1}x \end{aligned}$$

and using the SVD and denoting $E=e_iv^{\mathsf{T}}$, we get

$$\begin{aligned} & x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}x-x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x \\ = \;& x^{\mathsf{T}}\left(V\Sigma^{-1}U^{\mathsf{T}}\right)e_i\cdot v^{\mathsf{T}}\left(V'\Lambda^{-2}V'^{\mathsf{T}}\right)x \\ & +x^{\mathsf{T}}\left(V\Sigma^{-2}V^{\mathsf{T}}\right)v\cdot e_i^{\mathsf{T}}\left(U'\Lambda^{-1}V'^{\mathsf{T}}\right)x \end{aligned}$$

So now, assume x is sampled from $A^{\mathsf{T}}Y$. (The case of $A'^{\mathsf{T}}Y$ 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 $x=A^{\mathsf{T}}y=V\Sigma U^{\mathsf{T}}y$ and equivalently $x=(A'^{\mathsf{T}}-E^{\mathsf{T}})y=V'\Lambda U'^{\mathsf{T}}y-ve_i^{\mathsf{T}}y$. The above calculation shows that

$$\left|x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}x-x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x\right|\leq t_1\cdot t_2+t_3\cdot t_4$$

where for $i=1,2,3,4$ we have $t_i=|\bar{v}_i\cdot y|$ and

$$\begin{aligned} &\bar{v}_1=U\Sigma V^{\mathsf{T}}V\Sigma^{-1}Ue_i=e_i, && \text{so } \|\bar{v}_1\|=1 \\ &\bar{v}_2=U'\Lambda^{-1}V'^{\mathsf{T}}v-e_iv^{\mathsf{T}}V'\Lambda^{-2}V'^{\mathsf{T}}v, && \text{so } \|\bar{v}_2\|\leq\frac{\lambda_d+1}{\lambda_d^2} \\ &\bar{v}_3=U\Sigma^{-1}V^{\mathsf{T}}v, && \text{so } \|\bar{v}_3\|\leq\tfrac{1}{\sigma_d} \\ &\bar{v}_4=e_i-e_iv^{\mathsf{T}}V'\Lambda^{-1}U'^{\mathsf{T}}e_i, && \text{so } \|\bar{v}_4\|\leq 1+\tfrac{1}{\lambda_d} \end{aligned}$$

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. $\geq 1-\delta_0$ we have that for every i it holds that $t_i\leq\sqrt{\ln(4/\delta_0)}\|\bar{v}_i\|$. So to conclude the proof of the first stage, we have that

$$\begin{aligned} \left|x^{\mathsf{T}}(A^{\mathsf{T}}A)^{-1}x-x^{\mathsf{T}}(A'^{\mathsf{T}}A')^{-1}x\right| &\leq 2(\frac{1}{w}+\frac{1}{w^2})\ln(4/\delta_0) \\ &\leq \frac{4\ln(4/\delta_0)}{w}\leq\epsilon_0 \end{aligned}$$

Stage 2.

We assume step 4 was applied, and denote $B=U(\sqrt{\Sigma^2+w^2I})V^{\mathsf{T}}$ and $B'=U'(\sqrt{\Lambda^2+w^2I})V'^{\mathsf{T}}$. 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 $O^{\mathsf{T}}=B^{\mathsf{T}}Y$, and compare $X=B^{\mathsf{T}}Y$ to $X'=B'^{\mathsf{T}}Y$. The theorem merely requires Claim A.1 to hold, and they, in turn, depend on the following two conditions.

$$e^{-\epsilon_0/2}\leq\sqrt{\frac{\det(B'^{\mathsf{T}}B')}{\det(B^{\mathsf{T}}B)}}\leq e^{\epsilon_0/2} \tag{6}$$

$$\mathbf{Pr}_x\left[\frac{1}{2}\left|x^{\mathsf{T}}\left((B^{\mathsf{T}}B)^{-1}-(B'^{\mathsf{T}}B')^{-1}\right)x\right|\geq\epsilon_0/2\right]\leq\delta_0 \tag{7}$$

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 A.1 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. (6) holds. The analysis is a simple variation on the proof of Eq. (4). Fact A.2 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. (7). We start with an observation regarding $A'^{\mathsf{T}}A$ and $B'^{\mathsf{T}}B'$.

$$\begin{aligned} A'^{\mathsf{T}}A' &= A'^{\mathsf{T}}(A+E)=A^{\mathsf{T}}A+A'^{\mathsf{T}}E+E^{\mathsf{T}}A \\ B^{\mathsf{T}}B &= V(\Sigma^2+w^2I)V^{\mathsf{T}}=A^{\mathsf{T}}A+w^2I \\ B'^{\mathsf{T}}B' &= V'(\Lambda^2+w^2I)V'^{\mathsf{T}}=A'^{\mathsf{T}}A'+w^2I \end{aligned}$$

So $B'^{\mathsf{T}}B'-B^{\mathsf{T}}B=A'^{\mathsf{T}}E+E^{\mathsf{T}}A$. Now we can follow the same outline as in the proof of (5). Fix x, then:

$$\begin{aligned} & x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}x-x^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}x \\ = \;& x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}(B'^{\mathsf{T}}B')(B'^{\mathsf{T}}B')^{-1}x-x^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}x \\ = \;& x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}\left[B^{\mathsf{T}}B+A'^{\mathsf{T}}E+E^{\mathsf{T}}A\right](B'^{\mathsf{T}}B')^{-1}x \\ & -x^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}x \\ = \;& x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}\left[A'^{\mathsf{T}}E+E^{\mathsf{T}}A\right](B'^{\mathsf{T}}B')^{-1}x \\ = \;& x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}(A^{\mathsf{T}}+E^{\mathsf{T}})e_i\;\cdot\;v^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}x \\ & +x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}v\;\cdot\;e_i^{\mathsf{T}}\left(A'-E\right)(B'^{\mathsf{T}}B')^{-1}x \end{aligned}$$

It is straight-forward to see that the i-th spectral values of $(B^{\mathsf{T}}B)^{-1}A$ 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 $(B'^{\mathsf{T}}B')^{-1}A'$. 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 $x=B^{\mathsf{T}}y$:

$$\begin{aligned} & x^{\mathsf{T}}(B^{\mathsf{T}}B)^{-1}x-x^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}x \\ & =y^{\mathsf{T}}\left[B(B^{\mathsf{T}}B)^{-1}(A^{\mathsf{T}}+E^{\mathsf{T}})e_i\right]\;\cdot\;\left[v^{\mathsf{T}}(B'^{\mathsf{T}}B')^{-1}B^{\mathsf{T}}\right]y \\ & \quad +y^{\mathsf{T}}\left[B(B^{\mathsf{T}}B)^{-1}v\;\right]\cdot\left[e_i^{\mathsf{T}}\left(A'-E\right)(B'^{\mathsf{T}}B')^{-1}B^{\mathsf{T}}\right]y \end{aligned}$$

Lastly, we need to bound all terms that contain the multiplication $(B'^{\mathsf{T}}B')^{-1}B^{\mathsf{T}}y$ in comparison to $(B'^{\mathsf{T}}B')^{-1}B'^{\mathsf{T}}y=B'^{\dagger}y$. For instance, take the $term=|vec^{\mathsf{T}}y|$ for $vec^{\mathsf{T}}=e_i^{\mathsf{T}}\left(A'-E\right)(B'^{\mathsf{T}}B')^{-1}B^{\mathsf{T}}$, and define it as $vec^{\mathsf{T}}=z^{\mathsf{T}}B^{\mathsf{T}}$. 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 $B^{\mathsf{T}}y=(B'-\Delta)^{\mathsf{T}}y$. 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 $B^{\mathsf{T}}B-B'^{\mathsf{T}}B'=-A'^{\mathsf{T}}E-E^{\mathsf{T}}A$, a simple derivation gives that $\|Bz\|^2\leq\left(\|B'z\|+\|z\|\right)^2\leq\left(1+\frac{1}{w}\right)^2\|B'z\|^2$, and vice-versa. So if y is s.t. $\frac{|z^{\mathsf{T}}B^{\mathsf{T}}y|}{(1+\frac{1}{w})\|B'z\|}>Threshold$ then $\frac{|z^{\mathsf{T}}B^{\mathsf{T}}y|}{\|Bz\|}>Threshold$. Observe that $z^{\mathsf{T}}B^{\mathsf{T}}y$ is distributed like $\mathcal{N}(0,\|Bz\|^2)=\|Bz\|\mathcal{N}(0,1)$, and so we have that for every $\delta'>0$

$$\begin{aligned} & \mathbf{Pr}\left[|z^{\mathsf{T}}B^{\mathsf{T}}y|\geq\sqrt{\log(1/\delta')}\left(1+\frac{1}{w}\right)\|B'z\|\right] \\ = \;& \mathbf{Pr}\left[\left(\left(1+\frac{1}{w}\right)\|B'z\|\right)^{-1}|z^{\mathsf{T}}B^{\mathsf{T}}y|\geq\sqrt{\log(1/\delta')}\right] \\ \leq \;& \mathbf{Pr}\left[(\|Bz\|)^{-1}|z^{\mathsf{T}}B^{\mathsf{T}}y|\geq\sqrt{\log(1/\delta')}\right]\leq\delta' \end{aligned}$$

■