Differentially private vector sums for normally distributed data

Tim Sehested Poulsen
tpw705@alumni.ku.dk

January 22, 2023

Supervisor: Rasmus Pagh

UNIVERSITY OF COPENHAGEN

Abstract

This project aims at, reviewing some current methods to achieve differentially private estimates of sums of vectors. One such method which utilizes the variance of each dimension is then analysed further in an attempt to add less noise whilst still achieving differential privacy. I did so in a restricted setting in which data is normally distributed with priory knowledge about the distribution. I then analysed the noise in this setting and ways in which the mechanism could be used optimally. I then proceed to give bounds on the noise and lastly evaluate it empirically on synthetic datasets. This shows great promise as the empirical improvement is between 1 to d times better than a naive approach. Lastly the limitations of the proposed method and analysis is discussed and ways in further research could be carried out are reviewed.

Contents

1 Introduction

In recent years there has been an increased focus on the privacy of individuals, as we live in the age of data. More and more data is collected about us to be used for solving real world problems, be that through old school statistics or advanced algorithmic systems, such as deep neural networks. Often it is also the situation that to solve the problem at hand, using sensitive data is a necessity. In fact that is often the case, as the more sensitive the data is, the more useful it is. Many attempts have been made to combat this dilemma, preserve the privacy of the individuals involved whilst harvesting the fruitful information which the sensitive data contains. It has been shown repeatedly that de-anonymizing the data by removing "personally identifiable information", such as names, PIN codes (Personal Identification Number) or the alike is not enough to protect individuals from adversaries [14, 17]. Even publicizing aggregate statistics from sensitive data, has been shown to break privacy [8].
Many optimization problems rely upon calculating the sum, or equivalently the mean, of a set of vectors. If the data which is used contains sensitive information it is a necessity that calculating and using this aggregate statistic is done privately. The focus of this project will be to calculate the sum of a set of vectors whilst maintaining the privacy of individuals.

In an effort to mathematically define what is meant by privacy and giving a basis for algorithms to be compared and measured on their degree of privacy preservation the term *Differential Privacy* was born. It is merely a mathematical definition (which will be covered shortly), but has been the golden standard on which algorithms can be proofed against. In essence it encapsulates that the effect of any individual on the output of a statistic, should be limited; preferably to such an extend that the individual is not recognizable.
In the context of this project it specifically means that it should, to some extent, not be possible to deduce whether any one individual was part of a given sum of vectors.

2 Definitions & Preliminaries

Before I mathematically introduce the concept of Differential Privacy, there are some prerequisites in order to understand the definition. Firstly, I must introduce:

The dataset which will mostly be written as $X \in \mathbb{R}^{n\times d}$. I have that n denotes the number of entries in the dataset and d is the number of dimensions of the dataset. I will throughout the report refer to a single entry of the dataset as x_i and dimension $j \in [d]$ of entry $i \in [n]$ as $x_{i,j}$. I will also be using the notation $[n]$ which is the set $[n] = \{1, 2, \ldots, n-1, n\}$.

The following two definitions are essential to the definition of differential privacy, and are used when arguing about privacy and proving that an algorithm satisfies differential privacy.

Definition 2.1 (Neighboring dataset [7])**.** *Two dataset $X, X' \in \mathbb{R}^{n\times d}$ are said to be neighboring if they differ in at most a single entry. Neighboring dataset are denoted with*

the relation $X \sim X'$ and defined as follows:

$$X \sim X' \iff |\{i \in [n] \mid x_i \neq x_i'\}| \leq 1$$

Definition 2.2 (Sensitivity [7, Definition 3.8])**.** *Let $f(X) : \mathbb{R}^{n \times d} \to \mathbb{R}^k$ be a function, in which $k \in \mathbb{N}$. The l_p-sensitivity of f is the maximal possible l_p-norm of the difference between the output of f on two neighboring dataset. We denote the sensitivity as*

$$\Delta_p(f) = \max_{X \sim X'} \|f(X) - f(X')\|_p$$

Throughout the report, I will only be working with l_2-sensitivity and will just denote this as $\Delta(f)$ for ease of notation.

Differential privacy has multiple slightly different formal definitions, all of which are very similar, and there exist ways of defining one of them in terms of another. I will be working with the most standard one, which is called (ε, δ)-Differential Privacy, referred to as (ε, δ)-DP .

Definition 2.3 ((ε, δ)-Differential Privacy [7, Definition 2.4])**.** *A randomized algorithm $\mathcal{M} : \mathbb{R}^{n \times d} \to \mathcal{R}$ is (ε, δ)-differentially private if for all possible subsets of outputs $S \subseteq \mathcal{R}$ and all pairs of neighboring dataset $X \sim X'$ we have that*

$$\Pr[M(X) \in S] \leq e^{\varepsilon} \cdot \Pr[M(X') \in S] + \delta$$

The definition of differential privacy lends itself to possess some very useful properties. I will cover two of these, which are of importance to this project.

Lemma 2.1 ((ε, δ)-DP under post-processing [7, Proposition 2.1])**.** *Let $\mathcal{M} : \mathbb{R}^{n \times d} \to \mathcal{R}$ be an (ε, δ)-differentially private algorithm. Let $f : \mathcal{R} \to \mathcal{R}'$ be an arbitrary randomized mapping, then $f \circ \mathcal{M} : \mathbb{R}^{n \times d} \to \mathcal{R}'$ is (ε, δ)-DP .*

Proof

Fix any pair of neighbouring datasets $X \sim X'$ and let $S \subseteq \mathcal{R}'$ be an arbitrary event. We then define $T = \{r \in \mathcal{R} \mid f(r) \in S\}$. We thus have that

$$\begin{aligned} &\Pr[f(\mathcal{M}(X)) \in S] = \Pr[\mathcal{M}(X) \in T] \\ &\leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(X') \in T] + \delta = e^{\varepsilon} \cdot \Pr[f(\mathcal{M}(X')) \in S] + \delta \end{aligned}$$

■

Lemma 2.2 ((ε, δ)-DP under composition [7, Theorem 3.16])**.** *Let $\mathcal{M}_i : \mathbb{R}^{n \times d} \to \mathcal{R}_i$ be an $(\varepsilon_i, \delta_i)$-differentially private algorithm for $i \in [k]$. Then if $\mathcal{M}_{[k]} : \mathbb{R}^{n \times d} \to \prod_{i=1}^{k} \mathcal{R}_i$ is defined to be $\mathcal{M}_{[k]} = (\mathcal{M}_1(X), \mathcal{M}_2(X), \ldots, \mathcal{M}_k(X))$ then we have that $\mathcal{M}_{[k]}$ is $\left(\sum_{i \in [k]} \varepsilon_i, \sum_{i \in [k]} \delta_i\right)$-differentially private.*

The proof is a bit long and is omitted here, but can be found in Appendix B of [7].

Error Measure As this report concerns itself exclusively with the sum of entries in a dataset, error will be defined as the expected squared l_2-norm between the true sum and the output of a randomized algorithm. So let $X \in \mathbb{R}^{n\times d}$ be the dataset and $f(X) = \sum_i^n x_i$ be the true sum of all entries. The error of a randomized algorithm $M : \mathbb{R}^{n\times d} \to \mathbb{R}^d$ which estimates $f(X)$ is then

$$\mathrm{Err}(M) := \mathbb{E}\left[\|M(X) - f(X)\|^2\right]$$

2.1 Gaussian Random Variables

As this project will be working mainly with gaussian random variables I will lay a firm foundation of a few important properties which they possess. I will however omit the proofs as they are of no importance to the project.

Lemma 2.3. *Let $X \sim \mathcal{N}(\mu, \sigma^2)$ be a gaussian random variable and let $r \in \mathbb{R}$ be a constant. We then have that*

$$rX \sim \mathcal{N}(r\mu, (r\sigma)^2)$$
$$X - r \sim \mathcal{N}(\mu - r, \sigma^2)$$

Lemma 2.4 (Expectation of a quadratic random variable [2, Theorem 6])**.** *Let X be a d-dimensional random vector with expected value $\mathbb{E}[X] = \boldsymbol{\mu_X}$ and covariance matrix $Var[X] = \boldsymbol{\Sigma_X}$. Let also A be a constant $d \times d$ symmetric matrix, then*

$$\mathbb{E}\left[X^T A X\right] = tr\left(A\boldsymbol{\Sigma_X}\right) + \boldsymbol{\mu}^T A \boldsymbol{\mu}$$

Corollary 2.1. *Let $X \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma_X})$ be a d-dimensional gaussian vector with expected value μ_j for dimension j, and let σ_j^2 denote the variance of the j'th dimension where $1 \leq j \leq d$. By lemma 2.4 we have that the expected l_2-norm of such a vector is given by*

$$\mathbb{E}\left[\|X\|^2\right] = tr(\boldsymbol{\Sigma_X}) + \|\boldsymbol{\mu}\|^2 = \sum_{j=1}^{d} \sigma_j^2 + \mu_j^2$$

3 Algorithms

3.1 The Gaussian Mechanism

One of the most fundamental algorithms for achieving (ε, δ)-DP is the Gaussian Mechanism [7]. It computes the real value of a statistic, where the l_2-sensitivity is known. That is it produces a (ε, δ)-DP estimate of a function $g : \mathbb{R}^{n\times d} \to \mathbb{R}^d$ where the l_2-sensitivity $\Delta(g)$ is known. It does so by computing the value of $g(X)$ and then adding noise to each dimension drawn from the normal distribution $\mathcal{N}(0, \sigma^2_{\varepsilon,\delta})$. The variance $\sigma^2_{\varepsilon,\delta}$ is chosen specifically to satisfy (ε, δ)-DP , and it will shortly be covered how to choose this. This can be seen as adding a noise vector η which is then distributed according to the multivariate normal distribution $\mathcal{N}(\mathbf{0}, \sigma^2_{\varepsilon,\delta} I)$. Pseudo code for the algorithm can be seen in Algorithm 1.

Algorithm 1 The Gaussian Mechanism

Input

$\sigma_{\varepsilon,\delta}$ Standard deviation required to achieve (ε, δ)-DP

$X \in \mathbb{R}^{n \times d}$ Dataset

Output

(ε, δ)-DP estimate of $g(X)$

for $j \in [d]$ **do**

$\eta_j \leftarrow$ sample from $\mathcal{N}(0, \sigma^2_{\varepsilon,\delta})$

end for

return $g(X) + \boldsymbol{\eta}$

The algorithm quite intuitively produces error which is purely given by the norm of the noise added, and the expected error can be calculated to be

$$\mathbb{E}\left[\|(g(X) + \boldsymbol{\eta}) - g(X)\|^2\right] = \mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right]$$

Which by corollary 2.1 is

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] = \sum_i^d \sigma^2_{\varepsilon,\delta} = d \cdot \sigma^2_{\varepsilon,\delta} \tag{1}$$

It is apparent that the main difficulty of the mechanism lies in determining a $\sigma_{\varepsilon,\delta}$ which achieves (ε, δ)-DP , and preferably the smallest such one.

The following theorem was initially proven

Theorem 1. [7, Theorem 3.22] *Let $g : \mathbb{R}^{n \times d} \to \mathbb{R}^d$ be an arbitrary d-dimensional function which by definition 2.2 has l_2-sensitvity $\Delta(g) = \max_{X \sim X'} \|g(X) - g(X')\|$, and let $\varepsilon \in (0, 1)$. The Gaussian Mechanism with $\sigma_{\varepsilon,\delta} = \Delta(g)\sqrt{2\ln(1.25/\delta)}/\varepsilon$ is (ε, δ)-DP .*

The proof is rather long and is therefore omitted here. A few years later it was shown in [1] how to compute the minimal $\sigma_{\varepsilon,\delta}$.

Theorem 2. [1, Theorem 8] *The Gaussian Mechanism is differentially private if and only if $\sigma_{\varepsilon,\delta} \geq \Delta(g) \cdot \sigma_{opt}$ where $\sigma_{opt} \in \mathbb{R}$ is the smallest value greater than 0, which satisfies*

$$\Phi\left(\frac{1}{2\sigma_{opt}} - \varepsilon\sigma_{opt}\right) - e^{\varepsilon}\Phi\left(-\frac{1}{2\sigma_{opt}} - \varepsilon\sigma_{opt}\right) \leq \delta \tag{2}$$

Where Φ is the cumulative density function of a standard normal distribution, thus

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt$$

In [1] it is also shown how to compute this value, but since this is of no importance to this project it will therefore not be covered here. For the rest of the report, I will only be referring to σ_{opt} as the minimal value given by equation (2).

The main downside of the Gaussian Mechanism is that it adds equal noise to all dimensions, even if additional information about the skewness of the data was known. As this report focuses on giving differentially private estimates of sums of vectors, it is

natural to ask whether adding equal noise in all dimensions is optimal in all settings. Let us define the function of interest $f(X) = \sum_{i\in[n]} x_i$ which is the sum of vectors/entries in a dataset $X \in \mathbb{R}^{n\times d}$. The sensitivity of this function must be the largest possible l_2 distance between two vectors in the dataset. Therefore if the largest difference between any two neighbouring datasets in the j'th dimension, can be limited as follows

$$\Delta_j := \max_{X\sim X'} |X^{(j)} - X'^{(j)}| \tag{3}$$

where $X^{(j)}$ denotes the j'th dimension over the dataset. We then have the sensitivity

$$\Delta(f) = \max_{X\sim X'} \|f(X) - f(X')\| = \|\boldsymbol{\Delta}\| = \sqrt{\sum_{j\in[d]} \Delta_j^2} \tag{4}$$

where $\boldsymbol{\Delta} = (\Delta_1, \Delta_2, \ldots, \Delta_d)$.
This means that by equation (1) the expected error of the Gaussian Mechanism when estimating $f(X)$ is given by

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] = d \cdot \sigma_{\varepsilon,\delta}^2 = d \cdot \left(\Delta(f) \cdot \sigma_{opt}\right)^2 = d \cdot \sigma_{opt}^2 \cdot \|\boldsymbol{\Delta}\|^2 \tag{5}$$

3.2 The Elliptical Gaussian Mechanism

A logical next step would be to add noise to each dimension such that it is given as a monotone function of the sensitivity of that dimension. This has been studied in [15] and lays the foundation for this report. Their mechanism, called the *Elliptical Gaussian Mechanism* works very similarly to the Gaussian Mechanism as described by algorithm 1, though instead of sampling all η_j from $\mathcal{N}(0, \sigma_{\varepsilon,\delta}^2)$, here they are instead drawn from $\mathcal{N}\left(0, \left(\sigma_{opt} \cdot \frac{\Delta_j}{b_j}\right)^2\right)$. The algorithm is described in detail in Algorithm 2.

Algorithm 2 The Elliptical Gaussian Mechanism

Input

σ_{opt}	Standard deviation as defined by theorem 2
$X \in \mathbb{R}^{n\times d}$	Dataset
$\boldsymbol{b} \in \mathbb{R}^d$	Scaling vector, where $\|\boldsymbol{b}\| = 1$
$\boldsymbol{\Delta} \in \mathbb{R}^d$	Sensitivies of all dimensions

Output

(ε, δ)-DP estimate of $f(X) = \sum_{i\in[n]} x_i$

for $j \in [d]$ **do**
  $\sigma_j \leftarrow \sigma_{opt} \cdot \frac{\Delta_j}{b_j}$
  $\eta_j \leftarrow$ sample from $\mathcal{N}(0, \sigma_j^2)$
end for
return $f(X) + \eta$

Though the algorithm seems short and simple, the only thing which distinguishes it from The Gaussian Mechanism is to draw samples from $\mathcal{N}(0, \left(\sigma_{opt} \cdot \frac{\Delta_j}{b_j}\right)^2)$, it can be seen

as a transformation of space. Consider transforming the data by the factor $\frac{b_j}{\Delta_j}$ in each dimension. Then you would have that each dimension is bounded

$$\tilde{x}_{i,j} := x_{i,j} \cdot \frac{b_j}{\Delta_j}$$
$$\forall i, i' \in [n] : |x_{i,j} - x_{i',j}| \leq \Delta_j \implies |\tilde{x}_{i,j} - \tilde{x}_{i',j}| \leq b_j$$

and in this space the sensitivity of $f(\tilde{X}) = \sum_{i \in [n]} \tilde{x}_i$ is then

$$\Delta(f) = \|\boldsymbol{b}\| = \sqrt{\sum_{j \in [d]} b_j^2} = 1$$

This means that all points in X are mapped to lie within the unit ball, and in this space, one can simply add noise to each dimension drawn from $\eta_j \sim \mathcal{N}(0, \sigma_{opt}^2)$. Then the inverse transformation $\frac{\Delta_j}{b_j}$ can then be applied to each dimension of $f(\tilde{X})$ as well as η, giving a differentially private estimate of $f(X)$, because differential privacy is preserved under post-processing as per lemma 2.1.

By this interpretation, it can also be seen that the mechanism ensures privacy for vectors lying with a d-dimensional ellipsoid with radii $(b_1/\Delta_1, \ldots, b_d/\Delta_d)$, whereas the Gaussian mechanism, ensures privacy for vectors lying within a sphere with radius $\|\boldsymbol{\Delta}\|$. It can therefore be seen that the Gaussian mechanism is a special instance of the Elliptical Gaussian mechanism in which b_j is chosen such that $\Delta_j/b_j = \|\boldsymbol{\Delta}\|$.
Thus the thing that distinguishes the two mechanisms is the introduction of the scaling vector $\boldsymbol{b}$. This is the scaling of how much weight should be attributed to each dimension when adding noise. It is shown how to determine the optimal values for b_j in [15], and what the expected error of the mechanism then is.

Theorem 3 (Optimality of the Elliptical Gaussian Mechanism [15, Lemma 3.2 & 3.3])**.** *The value for b_j which minimizes the expected squared l_2 error $\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right]$ of Algorithm 2 is as follows*

$$b_j = \sqrt{\frac{\Delta_j}{\sum_{j \in [d]} \Delta_j}}$$

which leads the error to be

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] = \sigma_{opt}^2 \cdot \|\boldsymbol{\Delta}\|_1^2 \tag{6}$$

where $\|\cdot\|_1$ is the l_1 norm.

The proof will be omitted here, but it is very similar to the proofs for lemma 4.1 and Theorem 4 which are in section 4.1. Additionally, a generalized version of the theorem is proofed in [15].

Comparing the expected error between Algorithm 1 and Algorithm 2 we have the following ratio

$$\frac{d \cdot \sigma_{opt}^2 \cdot \|\boldsymbol{\Delta}\|^2}{\sigma_{opt}^2 \cdot \|\boldsymbol{\Delta}\|_1^2} = \left(\frac{\sqrt{d}\|\boldsymbol{\Delta}\|}{\|\boldsymbol{\Delta}\|_1}\right)^2 = \frac{d \sum_{j \in [d]} \Delta_j^2}{\left(\sum_{j \in [d]} \Delta_j\right)^2}$$

in which it can be seen that they are equal when all entries of $\boldsymbol{\Delta}$ are the same. Otherwise, the error for the Elliptical Gaussian Mechanism is lower when $\boldsymbol{\Delta}$ is skewed. As argued in [15] Algorithm 2 improves Algorithm 1 by a factor in $[1, d)$.

4 Gaussian Data

4.1 Clipping points

As previously mentioned the problem investigated here consists of releasing the sum of vectors in a dataset under differential privacy. More formally we wish to release the value of $f(X) = \sum_{i=1}^{n} x_i$ under (ε, δ)-DP .
The common factor for achieving (ε, δ)-DP in both the Gaussian Mechanism and the Elliptical Gaussian Mechanism is the knowledge that data lie within some bounds. Specifically for the Elliptical Gaussian Mechanism data is required to lie within some hyperrectangle. It is formally described by equation (3) essentially saying that there is an upper and lower bound on each dimension. This requirement is needed to know the l_2-sensitivity $\Delta(f)$ as shown in equation (4). In this project, I will change this assumption and instead look at the case where each dimension is normally distributed. This means that for each $j \in [d]$ we have that $x_{i,j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$. An equivalent formulation is that the data is multivariate gaussianly distributed but with no correlation between dimensions. This means that $x_i \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, where Σ is a diagonal matrix with the variance of each dimension along its diagonal. It is quite apparent that determining a limit Δ_j is impossible in this setting as the support of the gaussian distribution is $(-\infty, \infty)$. Several recent papers have combatted this by doing something called *clipping* [3, 11]. Clipping is the process of limiting all points to lie within a sphere of radius C. To perform clipping two things are required, (1) a center of the sphere which we call $\hat{\boldsymbol{\mu}}$ which will become obvious why later, and (2) the radius C. This means that every vector is transformed as such

$$\overline{x}_i := \hat{\boldsymbol{\mu}} + \min\left\{\frac{C}{\|x_i - \hat{\boldsymbol{\mu}}\|}, 1\right\} \cdot (x_i - \hat{\boldsymbol{\mu}})$$

Clipping entries by a factor C thus means that $\Delta(f) = 2C$ as any one entry cannot have more impact on the summation than C. If the summation $f(X)$ is performed on a clipped dataset $\overline{X}$ it is equivalent to defining the summation function $\overline{f} : \mathbb{R}^{n \times d} \to \mathbb{R}^d$ as

$$\overline{f}(X) = \sum_{i}^{n} \hat{\boldsymbol{\mu}} + \min\left\{\frac{C}{\|x_i - \hat{\boldsymbol{\mu}}\|}, 1\right\} \cdot (x_i - \hat{\boldsymbol{\mu}})$$

Then by Theorem 2 the gaussian mechanism with the function $\overline{f}$ is (ε, δ)-DP with $\sigma_{\varepsilon,\delta} = \Delta\left(\overline{f}\right)\sigma_{opt} = 2C\sigma_{opt}$. Though the mechanism is still (ε, δ)-DP it will now have a larger error when regarding the true sum $f(X) = \sum_i^n x_i$ as the actual answer. If the probability of clipping is set so low that we don't expect to clip any entries, and $\hat{\boldsymbol{\mu}}$ estimates $\boldsymbol{\mu}$ well, we can use $\overline{f}$ as an approximation of f. Say there are n points in a dataset, I will thus set the probability of clipping to be less than $\frac{1}{n}$ and get that

$$\mathbb{E}\left[\|\left(\overline{f}(X) + \eta\right) - f(X)\|^2\right] \approx \mathbb{E}\left[\|\left(f(X) + \eta\right) - f(X)\|^2\right] = \mathbb{E}\left[\|\eta\|^2\right]$$

This is of course not correct as some error will come from clipping points but in this report I will solely focus on the error introduced by the noise added. Another source of error will come from determining a good estimate for the center of the sphere, which optimally is $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$, i.e. the mean of the distribution which the data originates from. As lemma 2.2

states that (ε, δ)-DP methods can be combined, one can acquire such an estimate using some of the privacy budget. However, doing so is in essence almost the same problem as I am attempting to solve. The difference lies in the fact that this estimate could be given quite loosely and I am interested in the sum (or mean) of a specific dataset not that of the original distribution.

4.2 Noise scaling

In this setting, we again have the intuition that adding equal noise in all dimensions is non-optimal, and instead, the noise should be given as a monotone function of the standard deviation σ_j in that dimension. In a desire to achieve similar results to that of the Elliptical Gaussian Mechanism, just with normally distributed data, I will use somewhat the same approach to achieve (ε, δ)-DP . They achieve (ε, δ)-DP by finding scalings of each dimension such that all points lie within a unit ball, i.e. a ball with radius 1. There does not exist such a transformation for gaussian data, as there will always be a non-zero probability of observing points outside the unit ball. Instead, I will introduce the constraint that the expected squared distance between any point and the estimated mean $\hat{\boldsymbol{\mu}}$ after the transformation should be 1. I wish to find a scaling of each dimension b_j s.t.

$$\begin{aligned} \tilde{x}_i &:= (x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b} \\ \mathbb{E}\left[\|\tilde{x}\|^2\right] &= 1 \end{aligned} \tag{7}$$

where $\boldsymbol{b} = (b_1, b_2, \ldots, b_d)$, and $\odot$ is the element-wise product. Then $\overline{f}$ can be computed on this transformed dataset $\tilde{X}$, where the probability of clipping is less than $\frac{1}{n}$. As lemma 2.1 shows, (ε, δ)-DP is preserved under post processing, we can therefore add noise to each coordinate drawn from $\mathcal{N}(0, (2C\sigma_{opt})^2)$ in the transformed space to achieve (ε, δ)-DP . The transformation back to the original space is then done by multiplying each dimension with b_j^{-1}, and due to the linearity of transformation this is also done to the noise added. We end up with the following noise vector being added

$$\boldsymbol{\eta} = \left(b_1^{-1}\eta_1, b_2^{-1}\eta_2, \ldots, b_d^{-1}\eta_d\right)$$

in which all $\eta_j \sim \mathcal{N}(0, (2C\sigma_{opt})^2)$, and then by lemma 2.3 we have $b_j^{-1}\eta_j \sim \mathcal{N}(0, (b_j^{-1} \cdot 2C\sigma_{opt})^2)$. The error introduced by the noise is then given by $\|\boldsymbol{\eta}\|^2$ and due to Corollary 2.1 we have that the expectation of this error is

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] = \sum_{j=1}^{d} \left(b_j^{-1} \cdot 2C\sigma_{opt}\right)^2 = (2C\sigma_{opt})^2 \sum_{j=1}^{d} b_j^{-2} \tag{8}$$

The pseudo-code for the mechanism is given in Algorithm 3. At first glance, it seems quite different from Algorithm 2 despite it being very similar. The most notable thing here is that the transformation has to be performed on the dataset, as it is important which points are being clipped, to ensure that the mechanism is (ε, δ)-DP . In Algorithm 2 only the inverse transformation, b_j^{-1}, is being applied on the noise to be added.

Algorithm 3 The Elliptical Gaussian Mechanism for Gaussian data

Input

$\sigma_{opt} \in \mathbb{R}$	Standard deviation as defined by Theorem 2
$\boldsymbol{\sigma} \in \mathbb{R}^d$	Standard deviations of all dimension
$\hat{\boldsymbol{\mu}} \in \mathbb{R}^d$	Differentially private estimate of the mean of all dimensions
$X \in \mathbb{R}^{n \times d}$	Dataset, where $x_{i,j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$
$\boldsymbol{b} \in \mathbb{R}^d$	Scaling vector, s.t. $\mathbb{E}\left[\| \left(x_i - \hat{\boldsymbol{\mu}}\right) \odot \boldsymbol{b}\|^2\right] = 1$
$p_C \in \mathbb{R}$	Probability of clipping points

Output

(ε, δ)-DP estimate of $f(X)$

$\tilde{\boldsymbol{\sigma}} \leftarrow \boldsymbol{\sigma} \odot \boldsymbol{b}$
$C \leftarrow$ Minimal C satisfying $\Pr\left[\|\tilde{x}\| > C\right] \leq p_C$, for $\tilde{x} \sim \mathcal{N}(\mathbf{0}, \tilde{\boldsymbol{\sigma}})$
$T \leftarrow \mathbf{0}$
for $i \in [n]$ **do**
  $\tilde{x}_i \leftarrow (x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}$
  $T \leftarrow T + \min\left(\frac{C}{\|\tilde{x}_i\|}, 1\right) \cdot \tilde{x}_i$
end for
$T \leftarrow (T \odot \boldsymbol{b}^{-1}) + \hat{\boldsymbol{\mu}}$
for $j \in [d]$ **do**
  $\sigma_{\varepsilon,\delta,j} \leftarrow 2C\sigma_{opt} \cdot b_j^{-1}$
  $\eta_j \leftarrow$ sample from $\mathcal{N}(0, \sigma_{\varepsilon,\delta,j}^2)$
end for
return $T + \boldsymbol{\eta}$

If we try to minimize the error introduced by the noise, we are back to a very similar problem to the one solved in [15]. Minimize the error described in equation (8) under the constraint given in equation (7). The constraint can be given as

$$\mathbb{E}\left[\|(x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}\|^2\right] = \sum_{j \in [d]} (b_j \sigma_j)^2 + \|\boldsymbol{b} \odot (\boldsymbol{\mu} - \hat{\boldsymbol{\mu}})\|^2 = 1$$

due to Corollary 2.1. We then relax this constraint by omitting $\|\boldsymbol{b} \odot (\boldsymbol{\mu} - \hat{\boldsymbol{\mu}})\|^2$, both due to ease of analysis and as we assume $\hat{\boldsymbol{\mu}}$ estimates $\boldsymbol{\mu}$ well enough for this term to be negligible. Thus the constraint is

$$\sum_{j \in [d]} (b_j \sigma_j)^2 = 1 \tag{9}$$

This leads to the following Lemma

Lemma 4.1. *Let $X \in \mathbb{R}^{n \times d}$ be a dataset where each dimension is independently gaussianly distributed, i.e. $x_{i,j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$. Assuming that $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$, the expected squared norm of the noise $\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right]$, in Algorithm 3 is minimized when*

$$b_j = \frac{1}{\sqrt{\sigma_j}\sqrt{\sum_{i=1}^d \sigma_i}}$$

Proof

Using lagrangian multipliers we find the local maxima or minima of the function subject to equality constraints. To do so we construct the lagrangian function $\mathcal{L} : \mathbb{R}^{d+1} \to \mathbb{R}$ from the optimization problem given by equation (8) and the constraint given by (9) for ease of notation we define $\sigma_{\varepsilon,\delta} := 2C\sigma_{opt}$. I will also define $b_j \geq 0$ simply to make calculations easier, as b_j is squared there will also be a $-b_j$ which gives the same result.

$$\mathcal{L}(\boldsymbol{b}, \lambda) = \sigma_{\varepsilon,\delta}^2 \sum_{j\in[d]} b_j^{-2} + \lambda \left(\sum_{j=1}^{d} (\sigma_j b_j)^2 - 1 \right)$$

We then find the stationary points of it, by setting the derivative of it to $\mathbf{0}$. The derivative with respect to b_j is

$$\frac{\partial \mathcal{L}}{\partial b_j}(\boldsymbol{b}, \lambda) = \frac{\partial}{\partial b_j} \left(\sigma_{\varepsilon,\delta}^2 \cdot b_j^{-2} + \lambda (\sigma_j b_j)^2 \right) = -2\sigma_{\varepsilon,\delta}^2 b_j^{-3} + 2\lambda \sigma_j^2 b_j$$

I then solve $\frac{\partial \mathcal{L}}{\partial b_j}(\boldsymbol{b}, \lambda) = 0$ for b_j

$$\begin{aligned} &-2\sigma_{\varepsilon,\delta}^2 b_j^{-3} + 2\lambda\sigma_j^2 b_j = 0 \iff \lambda\sigma_j^2 b_j = \sigma_{\varepsilon,\delta}^2 b_i^{-3} \\ &\iff b_j^4 = \frac{\sigma_{\varepsilon,\delta}^2}{\lambda\sigma_j^2} \iff b_j = \frac{\sqrt{\sigma_{\varepsilon,\delta}}}{\lambda^{\frac{1}{4}}\sqrt{\sigma_j}} \end{aligned} \tag{10}$$

I now have the last partial derivative $\frac{\partial \mathcal{L}}{\partial \lambda}(\boldsymbol{b}, \lambda) = 0$ which I solve for λ using the previous expression for b_j.

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \lambda}(\boldsymbol{b}, \lambda) = \sum_{j=1}^{d} (\sigma_j b_j)^2 - 1 \\ &\sum_{j=1}^{d} (\sigma_j b_j)^2 - 1 = 0 \iff \sum_{j=1}^{d} \sigma_j^2 \left(\frac{\sigma_{\varepsilon,\delta}}{\sqrt{\lambda}\sigma_j} \right) = 1 \iff \\ &\frac{\sigma_{\varepsilon,\delta}}{\sqrt{\lambda}} \sum_{j=1}^{d} \frac{\sigma_j^2}{\sigma_j} = 1 \iff \sigma_{\varepsilon,\delta} \sum_{j=1}^{d} \sigma_j = \sqrt{\lambda} \end{aligned} \tag{11}$$

Inserting back into equation (10)

$$b_j = \frac{\sqrt{\sigma_{\varepsilon,\delta}}}{\lambda^{\frac{1}{4}}\sqrt{\sigma_j}} = \frac{\sqrt{\sigma_{\varepsilon,\delta}}}{\sqrt{\sigma_{\varepsilon,\delta} \sum_{i=1}^{d} \sigma_i} \sqrt{\sigma_j}} = \frac{1}{\sqrt{\sigma_j} \sqrt{\sum_{i=1}^{d} \sigma_i}} \tag{12}$$

To show that the value for b_j is a minimum I will use theorem 7.12 from [12]. I must then construct the bordered hessian matrix

$$\overline{H} = \begin{bmatrix} \frac{\partial^2 \mathcal{L}}{\partial \lambda^2} & \frac{\partial^2 \mathcal{L}}{\partial \lambda \partial \boldsymbol{b}} \\ \left(\frac{\partial^2 \mathcal{L}}{\partial \lambda \partial \boldsymbol{b}} \right)^T & \frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{b}^2} \end{bmatrix}$$

In which

$$\frac{\partial^2 \mathcal{L}}{\partial \lambda^2} = 0$$

$$\frac{\partial^2 \mathcal{L}}{\partial \lambda \partial \boldsymbol{b}} = \left(\frac{\partial^2 \mathcal{L}}{\partial \lambda \partial b_1}, \ldots, \frac{\partial^2 \mathcal{L}}{\partial \lambda \partial b_d} \right) = \left(2\sigma_1^2 b_1, \ldots, 2\sigma_d^2 b_d \right)$$

$$\frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{b}^2} = \begin{bmatrix} \frac{\partial^2 \mathcal{L}}{\partial x_1^2} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial x_1 \partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathcal{L}}{\partial x_d \partial x_1} & \cdots & \frac{\partial^2 \mathcal{L}}{\partial x_d^2} \end{bmatrix} = \operatorname{diag}\left(\frac{6\sigma_{\varepsilon,\delta}^2}{b_1^4} + 2\lambda\sigma_1^2, \ldots, \frac{6\sigma_{\varepsilon,\delta}^2}{b_d^4} + 2\lambda\sigma_d^2 \right)$$

The symmetric square matrix $\overline{H}$ of size $d+1$ only has values along the diagonal, the first row and the first column, the rest is 0 as there are no variables which depend on each other. Then theorem 7.12 from [12] states that if the r'th leading principal minor of $\overline{H}$ is negative, evaluated with b_j set by equation (12) and λ set by equation (11) for all $r \in \{2, 3, \ldots, d+1\}$, the values thus give a minimum to the optimization problem. Showing that the r'th leading principal minor is negative is shown as an induction proof in appendix A. So, based on that result, I can conclude that equation (12) gives the value for b_j which minimizes the expected squared norm of the noise in algorithm 3.

■

4.3 Expectation of noise

Now to evaluate the expected noise error of the mechanism, which is given by equation (8), it is highly important to determine the value C. As C should be given by the minimal value which satisfies the inequality

$$\Pr\left[\|(x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}\| > C \right] = \Pr\left[\|(x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}\|^2 > C^2 \right] < \frac{1}{n}$$

again where n denotes the number of points in the dataset. As x_i is a multivariate gaussian distribution, so is $(x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}$ by lemma 2.3, and it is the case that $\|(x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b}\|^2$ is the sum of independent gaussian variables squared. Such a sum is distributed as a generalized Chi-square [5], and neither its PDF nor CDF has a closed form in the general case. These quadratic forms of gaussian vectors have been studied well and for decades [9, 13], and some special cases do have closed forms. Some have also studied giving tail bounds on these probabilities [10], unfortunately, there were none which were of use in this setting. However, there do exist several numerical algorithms for evaluating the cumulative density function with high precision [4]. I would recommend using one of these algorithms in practice and I will do so in the empirical analysis in section 5. For the sake of analysis, I will provide an upper bound on C. There are many ways in which this can be done, I have chosen to do so using one of Bernstein's inequality, which results in a relatively simple expression.

Lemma 4.2. *Let $X_1, X_2, \ldots, X_d$ be d independent random gaussian variables where for $1 \leq j \leq d$ we have that $X_j \sim \mathcal{N}(0, \sigma_j^2)$. Let thereafter the largest standard deviation be denoted as $\sigma_* := \max_{j \in [d]} \sigma_j$, we can then bound the probability for the sum of variables*

squared as follows

$$\Pr\left[\sum_{j\in[d]} X_j^2 \geq t\sqrt{8\sum_{j\in[d]}\sigma_j^4} + \sum_{j\in[d]}\sigma_j^2\right] < e^{-t^2}$$

for

$$0 \leq t \leq \frac{1}{6\sigma_*^2}\sqrt{2\sum_{j\in[d]}\sigma_j^4}$$

Proof
At first we define the random variable $Y_j = X_j^2 - \mathbb{E}\left[X_j^2\right]$ using the j'th gaussian random variable. As $\mathbb{E}[Y_j] = \mathbb{E}\left[X_j^2 - \mathbb{E}\left[X_j^2\right]\right] = \mathbb{E}\left[X_j^2\right] - \mathbb{E}\left[X_j^2\right] = 0$ we have that Y_j is zero centered. We are thus interested in giving bounds on $\Pr\left[\sum_{j\in[d]} Y_j\right]$. We can use Bernstein's inequality, if there exists some $L \in \mathbb{R}$ which satisfies

$$\mathbb{E}\left[|Y_j^k|\right] \leq \frac{1}{2}\mathbb{E}\left[Y_j^2\right] L^{k-2}k! \tag{13}$$

for all $k \in \mathbb{N}$ with $k \geq 2$ and for all $j \in [d]$. [18]

Initially we have that $\mathbb{E}\left[|Y_j^k|\right] = \mathbb{E}\left[|\left(X_j^2 - \mathbb{E}\left[X_j^2\right]\right)^k|\right]$ and since $X_j^2 \geq 0$ and therefore also $\mathbb{E}\left[X_j^2\right] \geq 0$ we can therefore bound it by

$$\mathbb{E}\left[|\left(X_j^2 - \mathbb{E}\left[X_j^2\right]\right)^k|\right] \leq \mathbb{E}\left[|X_j^{2k}|\right] = \mathbb{E}\left[|\left(\sigma_j^2 Z\right)^k|\right] = \sigma_j^{2k}\cdot\mathbb{E}\left[|Z^k|\right]$$

Where $Z \sim \chi_1^2$ is a chi-square random variable with 1 degree of freedom. The moment-generating function of Z is $\mathbb{E}[Z^m] = 1\cdot 3\cdot 5\cdot\ldots\cdot(2m-1)$. Using this we can get the following bound

$$\sigma_j^{2k}\cdot\mathbb{E}\left[|Z^k|\right] = \sigma_j^{2k}\cdot\prod_{c=1}^{k}(2c-1) = \sigma_j^{2k}\cdot 3\cdot\prod_{c=3}^{k}(2c-1)$$
$$\leq \sigma_j^{2k}\cdot 3\cdot\prod_{c=3}^{k}2c = \frac{3}{2}\sigma_j^{2k}\cdot 2^{k-2}\cdot k!$$

Concluding that $\mathbb{E}\left[|Y_j^k|\right] \leq \frac{3}{2}\sigma_j^{2k}\cdot 2^{k-2}\cdot k!$.
Secondly I calculate $\mathbb{E}\left[Y_j^2\right]$ exactly to be used in equation (13)

$$\mathbb{E}\left[Y_j^2\right] = \mathbb{E}\left[\left(X_j^2 - \mathbb{E}\left[X_j^2\right]\right)^2\right] \tag{14}$$
$$= \mathbb{E}\left[X_j^4 + \mathbb{E}\left[X_j^2\right]^2 - 2X_j^2\mathbb{E}\left[X_j^2\right]\right] \tag{15}$$
$$= \mathbb{E}\left[X_j^4\right] + \mathbb{E}\left[X_j^2\right]^2 - 2\mathbb{E}\left[X_j^2\right]^2 \tag{16}$$
$$= \mathbb{E}\left[X_j^4\right] - \mathbb{E}\left[X_j^2\right]^2 = \sigma_j^4\mathbb{E}\left[Z^2\right] - \left(\sigma_j^2\mathbb{E}[Z]\right)^2 \tag{17}$$
$$= 3\sigma_j^4 - \sigma_j^4 = 2\sigma_j^4 \tag{18}$$

Equation (13) is therefore rewritten to

$$\frac{3}{2}\sigma_j^{2k} \cdot 2^{k-2} \cdot k! \leq \frac{1}{2} \cdot 2\sigma_j^4 L^{k-2} k! \iff \frac{3}{2}\sigma_j^{2k-4} \cdot 2^{k-2} \leq L^{k-2}$$

This expression is then split into two cases, when $k = 2$, in which it can be seen from equation (13) that this holds for any $L \in \mathbb{R}$. The other case is $k > 2$, where we have that

$$\left(\frac{3}{2}\right)^{\frac{1}{k-2}} \sigma_j^2 \cdot 2 \leq L$$

As $\lim_{k\to\infty} \left(\frac{3}{2}\right)^{\frac{1}{k-2}} = 1$, and these constraints must hold for all $j \in [d]$, we can define $\sigma_* = \max_{j\in[d]} \sigma_j$ and finally have that

$$L = 3\sigma_*^2$$

We can then use Bernstein's inequality to bound the following

$$\Pr\left[\sum_{j\in[d]} Y_j \geq 2t\sqrt{\sum_{j\in[d]} \mathbb{E}\left[Y_j^2\right]}\right] < e^{-t^2} \tag{19}$$

Which in this case can be rewritten to get a tail bound on $\sum_{j\in[d]} X_j^2$, by reusing results from equations (14)-(18)

$$\Pr\left[\sum_{j\in[d]} Y_j \geq 2t\sqrt{\sum_{j\in[d]} \mathbb{E}\left[Y_j^2\right]}\right] = \Pr\left[\sum_{j\in[d]} \left(X_j^2 - \mathbb{E}\left[X_j^2\right]\right) \geq 2t\sqrt{\sum_{j\in[d]} 2\sigma_j^4}\right]$$

$$= \Pr\left[\sum_{j\in[d]} X_j^2 \geq 2t\sqrt{\sum_{j\in[d]} 2\sigma_j^4} + \sum_{j\in[d]} \mathbb{E}\left[X_j^2\right]\right] = \Pr\left[\sum_{j\in[d]} X_j^2 \geq 2t\sqrt{\sum_{j\in[d]} 2\sigma_j^4} + \sum_{j\in[d]} \sigma_j^2\right]$$

$$= \Pr\left[\sum_{j\in[d]} X_j^2 \geq t\sqrt{8\sum_{j\in[d]} \sigma_j^4} + \sum_{j\in[d]} \sigma_j^2\right]$$

Finally giving us that

$$\Pr\left[\sum_{j\in[d]} X_j^2 \geq t\sqrt{8\sum_{j\in[d]} \sigma_j^4} + \sum_{j\in[d]} \sigma_j^2\right] < e^{-t^2}$$

as long as t lies within the following bounds

$$0 \leq t \leq \frac{1}{2L}\sqrt{\sum_{j\in[d]} \mathbb{E}\left[Y_j^2\right]} = \frac{1}{6\sigma_*^2}\sqrt{2\sum_{j\in[d]} \sigma_j^4}$$

■

Combining lemma 4.2 with lemma 4.1, and assuming $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$. we can conclude the following:

Theorem 4. *Let $X \in \mathbb{R}^{n\times d}$ be a dataset in which all dimensions are independently gaussian random variables, i.e. $x_{i,j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$. Let $\sigma_* := \max_{j\in[d]} \sigma_j$ be the maximal standard deviation of all dimensions. Under the assumption that the estimate $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$, we then have that when $\ln(n) \leq \frac{\sum_{j\in[d]} \sigma_j^2}{18\sigma_*^2}$ the expected squared error of the noise in algorithm 3 can be upper bounded by*

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] \leq 4\sigma_{opt}^2 \cdot \sum_{j\in[d]} \sigma_j \cdot \left(\sqrt{8\ln(n)\sum_{i\in[d]} \sigma_i^2} + \sum_{j\in[d]} \sigma_j\right)$$

when each b_j is chosen according to lemma 4.1.

Proof
By equation (8) the error is given by

$$\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] = (2C\sigma_{opt})^2 \cdot \sum_{i=1}^{d} b_j^{-2} \tag{20}$$

We thus need to determine an upper bound for C, we begin by looking at the transformed data when $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}$.

$$\tilde{x}_i := (x_i - \hat{\boldsymbol{\mu}}) \odot \boldsymbol{b} \text{ , giving } \tilde{x}_{i,j} \sim \mathcal{N}\left(0, (b_j\sigma_j)^2\right)$$
$$\Pr\left[\|\tilde{x}_i\| \geq C\right] = \Pr\left[\|\tilde{x}_i\|^2 \geq C^2\right] = \Pr\left[\sum_{j\in[d]} \tilde{x}_{i,j}^2 \geq C^2\right]$$

This means I can give an upper bound on C^2, whilst having $b_j\sigma_j = \sqrt{\frac{\sigma_j}{\sum_{i\in[d]}\sigma_i}}$ as the standard deviation. Furthermore, I wish the clipping probability to be less than n^{-1}, giving me

$$e^{-t^2} = \frac{1}{n} \implies t = \sqrt{\ln(n)}$$

Combining these results and using the upper bound on C from lemma 4.2 I get the following

$$\begin{aligned} C^2 &\leq \sqrt{\ln(n)}\sqrt{8\sum_{j\in[d]}\left((b_j\sigma_j)^4\right)} + \sum_{j\in[d]}(b_j\sigma_j)^2 \\ &= \sqrt{8\ln(n)\sum_{j\in[d]}\left(\frac{\sigma_j}{\sum_{i\in[d]}\sigma_i}\right)^2} + \sum_{j\in[d]}\frac{\sigma_j}{\sum_{i\in[d]}\sigma_i} \\ &= \frac{\sqrt{8\ln(n)\sum_{j\in[d]}\sigma_j^2}}{\sum_{i\in[d]}\sigma_i} + 1 \end{aligned}$$

Inserting this back into equation (20) we conclude

$$\begin{aligned}
\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right] &= 4\sigma_{opt}^2 \cdot C^2 \cdot \sum_{i=1}^{d} b_j^{-2} \\
&\leq 4\sigma_{opt}^2 \cdot \left(\frac{\sqrt{8\ln(n)\sum_{j\in[d]}\sigma_j^2}}{\sum_{i\in[d]}\sigma_i} + 1 \right) \cdot \sum_{i=1}^{d} \left(\sigma_i \cdot \sum_{j\in[d]} \sigma_j \right) \\
&= 4\sigma_{opt}^2 \cdot \left(\frac{\sqrt{8\ln(n)\sum_{j\in[d]}\sigma_j^2}}{\sum_{i\in[d]}\sigma_i} + 1 \right) \cdot \left(\sum_{i=1}^{d} \sigma_i \right)^2 \\
&= 4\sigma_{opt}^2 \cdot \sum_{i=1}^{d} \sigma_i \cdot \left(\sqrt{8\ln(n)\sum_{j\in[d]}\sigma_j^2} + \sum_{i=1}^{d} \sigma_i \right)
\end{aligned}$$

And the constraint on $t = \sqrt{\ln(n)}$ from lemma 4.2 is

$$\begin{aligned}
&0 \leq t \leq \frac{1}{6(b_j\sigma_*)^2}\sqrt{2\sum_{j\in[d]}(b_j\sigma_j)^4} \iff \\
&\sqrt{\ln(n)} \leq \frac{1}{6}\frac{\sum_{j\in[d]}\sigma_j}{\sigma_*}\sqrt{2\sum_{j\in[d]}\left(\frac{\sigma_j}{\sum_{i\in[d]}\sigma_i}\right)^2} \iff \\
&\sqrt{\ln(n)} \leq \sqrt{\left(\frac{\sum_{j\in[d]}\sigma_j}{\sigma_*}\right)^2 \cdot \frac{2}{36} \cdot \sum_{j\in[d]}\left(\frac{\sigma_j}{\sum_{i\in[d]}\sigma_i}\right)^2} \iff \\
&\ln(n) \leq \frac{1}{18\sigma_*^2} \cdot \sum_{j\in[d]}\sigma_j^2 \iff \\
&\ln(n) \leq \frac{\sum_{j\in[d]}\sigma_j^2}{18\sigma_*^2}
\end{aligned}$$

■

At this point, it would be preferable to compare the error of algorithm 3, to that of doing no transformation, and similarly deciding on a bound C such that the clipping probability was less than n^{-1} on X. However, as I can only give upper bounds on these errors there is little to no value in comparing upper bounds as there is no guarantee on the tightness of these bounds. So even if I were to determine an upper bound for the error without transformation which was higher than that of algorithm 3, it could very well still be that the error was less in reality. As previously mentioned, there exist numerical solutions for evaluating $\Pr\left[\|x_i\| > C^2\right]$ so it is natural to perform an empirically evaluation of these errors and investigating whether the transformation lessens the error. Intuitively it makes sense that allowing noise to be added differently for each dimension, can only produce better results.

5 Emperical Evaluation

For the empirical evaluation, I wanted to investigate the improvement by calculating the expected error on hypothetical datasets. This is in contrast to using real-world datasets, where it is uncertain whether the data is distributed as a multivariate gaussian with independent dimensions. Secondly, I gain the benefit of creating a wide range of hypothetical scenarios and evaluating these. When creating a hypothetical dataset I am merely interested in the standard deviations of each dimension, thus there is no need to truly create a dataset but instead, I chose to synthetically create the parameters of a hypothetical dataset. So I chose to have the standard deviation of each dimension be chosen according to Zipf's law [16]. The distribution is often used to describe the sizes of cities or frequencies of words in texts, but it can also be used to describe discrete distributions with different levels of skewness. This property is what I will be utilizing, as I will choose the standard deviation of dimension $i \in [d]$ as

$$\sigma_i = \frac{\frac{1}{i^\alpha}}{\sum_{j \in [d]} \frac{1}{j^\alpha}}$$

The parameter α thus decides the skewness of how much each dimension contributes to the total amount of variance. Figure 5 shows how much each dimension contributes to the total variance for $\alpha \in \{0.1, 1, 3\}$ and $d = 100$. This means that for high α only a few dimensions have any variance, and for low α it is almost uniformly distributed.

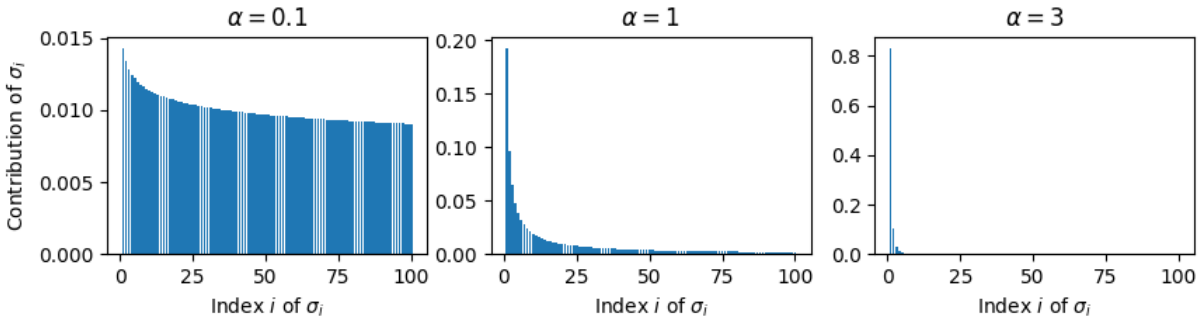

Figure 1: Examples of how standard deviations are distributed according to zipf's law for different skewness levels (α).

So the needed parameters to describe a hypothetical dataset are α, d the number of dimensions, and n the number of points which is needed to decide the clipping probability.

I then calculated the expected error as described by equation (20) denoting this as $\mathcal{E}_t$ and compare that to

$$\mathcal{E}_n = (2C_n\sigma_{opt})^2$$

In which C_n is chosen as the minimal value satisfying $\Pr\left[\|x_i\|^2 > C_n^2\right] \leq \frac{1}{n}$. I calculate this using the python package chi2comb, which is based on the algorithm described in [6].

I then calculate the fraction $\frac{\mathcal{E}_n}{\mathcal{E}_t}$ to compare the expected errors, and as I have σ_{opt}^2 in both the denominator and the numerator I can omit calculating this as doing so is not a

trivial task. Figure 5 shows calculating this fraction for varying $\alpha \in [10^{-2}, 10^2]$, and for $d \in \{10, 100, 1000\}$ and $n \in \{10^2, 10^3, 10^6\}$. It shows that when a few dimensions have much more variance than the rest, performing the transformation improves the expected error, for all configurations of n and d.

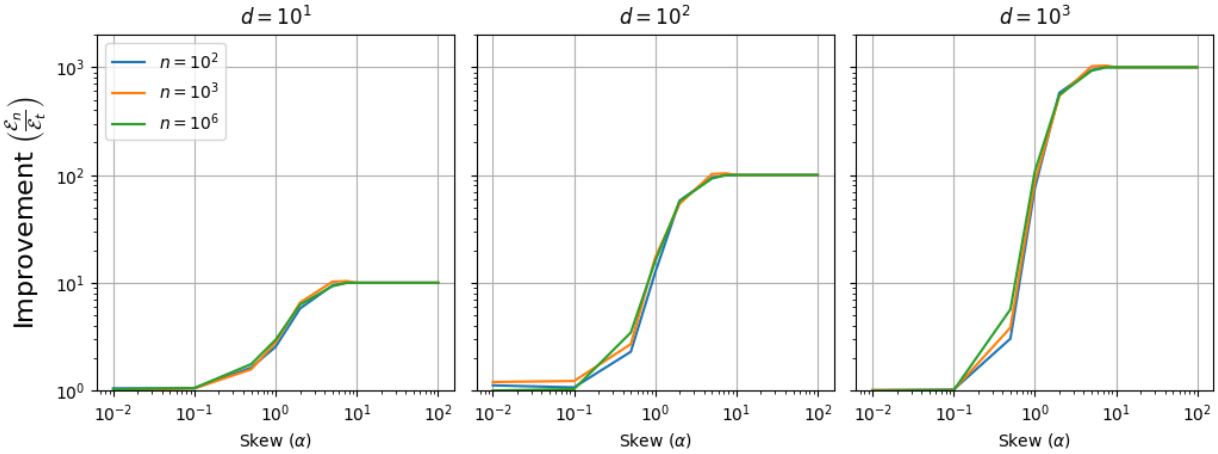

Figure 2: Improvement of applying the transformation. Each subplot shows for a fixed number of dimensions d, how much the improvement is for different skewness parameters α and the number of data points n, which is used to decide the clipping probability.

More generally it seems as though n has no impact on the improvement of the method, which is not trivial as the clipping probability is chosen as $\frac{1}{n}$. It could perhaps have been that the transformation changed the distribution in such a way that the tail was much longer. Interestingly I observe that the method improves by a factor of $[1, d]$ quite like how the Elliptical Gaussian Mechanism improves the Gaussian Mechanism by a factor of $[1, d)$. Perhaps a similar argument could be made in this case, if the distribution of the norms had a closed form or more analysis of it was done. But the empirical evidence is quite strong in favour of this phenomenon being present here as well.

All the code for generating these results and plots can be found on `https://github.com/TSPoulsen/egm`.

6 Discussion & Future work

Earlier when I defined the problem at hand I also defined the constraint that the expected norm after transforming the data should be 1, i.e. the constraint given by equation (9). This constraint is perhaps not optimal in terms of how the problem should be restricted. The idea behind the constraint is to force the data to become more symmetrical in the transformed space, however, it does not necessarily enforce this. An alternative constraint which I initially investigated for this project was to find a transformation, such that the majority of data entries lie within a ball with a diameter of 1, and entries outside would be clipped. In this transformed space it would be sufficient to add noise drawn from $\mathcal{N}(0, \sigma^2_{opt})$. The constraint would thus be the following

$$\Pr\left[\|x_i \odot \boldsymbol{b}\| > \frac{1}{2}\right] \leq \frac{1}{n} \tag{21}$$

To achieve optimal results with this constraint it would be required to know the cumulative density function of $\|x_i \odot \boldsymbol{b}\|$, which as discussed earlier is distributed as a generalized chi-square and does not have a closed form. So without a closed form, there would be no way of finding optimal values for b_j.

Another option would be to have the same constraint, but instead define it in terms of a tail inequality such as the one given by lemma 4.2. However the constraint on t in that lemma can be quite restrictive for some inputs. This means that it is not always possible to set t such that $e^{-t} = \frac{1}{n}$, and the method would not be applicable for all inputs. I also attempted to find optimal values for b_j whilst using Chebyshev's inequality to give closed forms for equation (21). Unfortunately, this lead to a very complicated optimization problem, which I did not attempt to solve. The benefit of using Chebyshev's inequality would be that it is very generalized and would be applicable to any dataset, but it is also quite well known that the bounds given by Chebyshev's inequality are often quite far from the true values. I settled for the constraint where the expected norm is 1, which intuitively seemed quite good and was much easier to solve. The empirical data also support this intuition.

In the empirical evaluation of the method, I chose to compare the method to that which I deemed the most natural thing to do if one were to use the Gaussian Mechanism. Simply clip points with probability $\frac{1}{n}$, and add noise according to this. Both this method, and doing it with a transformation do **not** take into account the bias which comes from clipping the data in the first place. It is not necessarily the case that the ignored clipping errors are equal, it can be that the few points which are clipped in the transformed space introduce much more error than the ones which are clipped in the original space. The argument for ignoring them is that the clipping probability is set very low, thus only very few points are being clipped in the first place. In a situation where the clipping probability is set to account for the bias-variance tradeoff between noise error and clipping error, the clipping error will most likely have a larger impact which could be amplified by the transformation.
Generally Algorithm 3 (doing the transformation) should be compared to state-of-the-art methods, such as the ones in [3, 11]. In such as comparison the clipping bias should most certainly be taken into account as well, and in a more in-depth analysis of the Algorithm 3

it would interesting to investigate the bias-variance tradeoff and find an optimal clipping probability p_C.

6.1 Limitations

One of the biggest limitation of the method proposed in this project is the requirement for an estimate of the mean $\boldsymbol{\mu}$ as this is a problem equivalent to the one which the mechanism proposed here solves. The desire is that incorporating this is possible, as other methods in recent research have achieved something similar. Many parallels can be drawn to [11], as they also clip points according to some sphere in which they privately find a center for the sphere. The main concept which is introduced in this project is that noise is added to each dimension as a function of its variance, which is not done in [11]. My hope is that the analysis presented here is of value in a context where finding the center of the sphere is incorporated into the mechanism itself. An intermediate step in achieving this would be to incorporate this estimate $\hat{\boldsymbol{\mu}}$ into the error analysis in lemma 4.1 and lemma 4.2. Then expressions for b_j and $\mathbb{E}\left[\|\boldsymbol{\eta}\|^2\right]$ could be given based on how well $\hat{\boldsymbol{\mu}}$ estimates $\boldsymbol{\mu}$.

Another major limitation of the proposed method, which is of the same nature, is the requirement for the knowing the variances of all dimensions. This is due to the same reasons as just discussed, that estimating σ_j is in itself a difficult task and would use up a large portion of a given privacy budget [11].

7 Conclusion

While much research has been performed to give good private estimates of sums over a dataset, not many has investigated using the variance of the data to do so. As noise inherently must be added to preserve the privacy of individuals, the intuition is that adding noise in each dimension should depend moreso on the variance in that dimension than the ones in other dimension. This means that more noise is added in dimensions with high variance in order to add less noise in dimensions with little variance. I investigated doing so, but in a very restricted setting. A setting where data was distributed normally and in which strong priori knowledge about the variance and essentially also the mean was available. In this setting I analysed ways in which the added noise could be minimized whilst up-holding the standard of differential privacy, and giving bounds for this noise. I also showed on synthetic dataset that the noise seemed to be somewhere between $[1, d]$ times better when commpared to a naive approach.

Despite major limitation of the proposed method, the project demonstrates the potential of utilizing the variance of individual dimensions and proposes ways in which the noise can be analysed. However, further research is needed to explore the applicability of this approach in more general settings and with different types of data distributions. Additionally, it would be beneficial to test the proposed method on real-world data where certain knowledge about the data distribution is not known.

References

[1] Balle, B., and Wang, Y. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. *CoRR abs/1805.06530* (2018).

[2] Bates, D. *Quadratic Forms of Random Variables.* University of Wisconsin-Madison: STAT 849 lectures, 2010.

[3] Biswas, S., Dong, Y., Kamath, G., and Ullman, J. Coinpress: Practical private mean and covariance estimation. *Advances in Neural Information Processing Systems 33* (2020), 14475–14485.

[4] Chen, T., and Lumley, T. Numerical evaluation of methods approximating the distribution of a large quadratic form in normal variables. *Computational Statistics & Data Analysis 139* (2019), 75–81.

[5] Das, A., and Geisler, W. S. A method to integrate and classify normal distributions, 2020.

[6] Davies, R. B. Algorithm as 155: The distribution of a linear combination of chi-square random variables. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 29*, 3 (1980), 323–333.

[7] Dwork, C., Roth, A., et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science 9*, 3–4 (2014), 211–407.

[8] Dwork, C., Smith, A., Steinke, T., and Ullman, J. Exposed! a survey of attacks on private data. *Annu. Rev. Stat. Appl 4*, 1 (2017), 61–84.

[9] Hallam, A. *Some Theorems on Quadratic Forms and Normal Variables.* Iowa State University: Econ 671 lectures, 2004.

[10] Hanson, D. L., and Wright, F. T. A bound on tail probabilities for quadratic forms in independent random variables. *The Annals of Mathematical Statistics 42*, 3 (1971), 1079–1083.

[11] Huang, Z., Liang, Y., and Yi, K. Instance-optimal mean estimation under differential privacy.

[12] Magnus, J. R. *Matrix differential calculus with applications in statistics and econometrics*, third edition. ed. Wiley Series in Probability and Statistics. Wiley, Hoboken, NJ, 2019.

[13] Mathai, A., and Provost, S. Quadratic forms in random variables.

[14] Narayanan, A., and Shmatikov, V. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)* (2008), IEEE, pp. 111–125.

[15] Pagh, R., and Lebeda, C. Private vector aggregation when coordinates have different sensitivity.

[16] Powers, D. M. W. Applications and explanations of Zipf's law. In *New Methods in Language Processing and Computational Natural Language Learning* (1998).

[17] Sweeney, L. Weaving technology and policy together to maintain confidentiality. *The Journal of Law, Medicine & Ethics 25*, 2-3 (1997), 98–110.

[18] Wainwright, M. J. *Basic tail and concentration bounds*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019, p. 21–57.

Appendix

A Induction proof of the r'th leading principal minor of the Bordered Hessian matrix in lemma 4.1

I will proof that for the matrix

$$\overline{H} = \begin{bmatrix} 0 & 2\sigma_1^2 b_1 & \dots & 2\sigma_d^2 b_d \\ 2\sigma_1^2 b_1 & \frac{6\sigma_{\varepsilon,\delta}^2}{b_1^4} + 2\lambda\sigma_1^2 & & \\ \vdots & & \ddots & 0 \\ 2\sigma_d^2 b_d & 0 & & \frac{6\sigma_{\varepsilon,\delta}^2}{b_d^4} + 2\lambda\sigma_d^2 \end{bmatrix}$$

the r'th leading principal minor is negative for all $r \in \{2, 3, \dots, d+1\}$ when b_j and λ is set as

$$b_j = \frac{1}{\sqrt{\sigma_j}\sqrt{\sum_{i\in[d]} \sigma_i}}$$
$$\lambda = \sigma_{\varepsilon,\delta}^2 \left(\sum_{i\in[d]} \sigma_i \right)^2$$

Proof
Initially I insert these values and get the following matrix

$$\overline{H}_{d+1} = \begin{bmatrix} 0 & \frac{2\sigma_1^{2/3}}{\sqrt{\sum_{i\in[d]} \sigma_i}} & \dots & \frac{2\sigma_d^{2/3}}{\sqrt{\sum_{i\in[d]} \sigma_i}} \\ \frac{2\sigma_1^{2/3}}{\sqrt{\sum_{i\in[d]} \sigma_i}} & 2\sigma_{\varepsilon,\delta}^2 \left(\sum_{i\in[d]} \sigma_i \right)^2 (3\sigma_1^2 + 1) & & \\ \vdots & & \ddots & 0 \\ \frac{2\sigma_1^{2/3}}{\sqrt{\sum_{i\in[d]} \sigma_i}} & 0 & & 2\sigma_{\varepsilon,\delta}^2 \left(\sum_{i\in[d]} \sigma_i \right)^2 (3\sigma_d^2 + 1) \end{bmatrix}$$

I define $\overline{H}_r$ to be the matrix with the first r rows and columns of $\overline{H}_{d+1}$. Secondly I will note that since $\sigma_i > 0$ by definition, we have that all entries $\overline{H}_{d+1,i,j}$ in this matrix are positive. So I will prove that the r'th leading principal minor $\det(\overline{H}_r) = |\overline{H}_r|$ is negative for all $r \in \{2, 3, \dots, d+1\}$ by induction on r. So for the base case I show that it is true for $r = 2$, which is the value

$$|\overline{H_2}| = 0 \cdot H_{2,1,1} - \overline{H}_{2,2,1} \cdot \overline{H}_{2,1,2} < 0$$

Which is true since all entries $\overline{H}_{r,i,j} > 0$. Thus it holds for the base case. Assume now the induction hypothesis, that $|\overline{H}_{r-1}| < 0$, the r'th leading principal minor is then given by

$$|\overline{H}_r| = |\overline{H}_{r-1}| \cdot \overline{H}_{r,r,r} + (-1)^{1+r} \cdot |B_{r-1}| \cdot \overline{H}_{r,1,r} \tag{22}$$

Where B_{r-1} is the matrix where the first row and the last column of $\overline{H}_r$ is removed. Then from equation (22) it can be seen that since $|\overline{H}_{r-1}| < 0$ by the indcution hypothesis then we have that

$$(-1)^{1+r} \cdot |B_{r-1}| < 0 \implies |\overline{H}_r| < 0 \tag{23}$$

which would complete the induction step. So to show that $(-1)^{r+1} \cdot |B_{r-1}| < 0$ I will observe that the last row in B_{r-1}, has only a single value in the first column and the rest are 0.

$$B_{r-1} = \begin{bmatrix} \overline{H}_{r,2,1} & \overline{H}_{r,2,2} & & 0 \\ & & \ddots & \\ \vdots & 0 & & \overline{H}_{r,r-1,r-1} \\ \overline{H}_{r,r,1} & 0 & \dots & 0 \end{bmatrix}$$

Therefore I can calculate the determinant by multiplying $(-1)^{r-1+1} \cdot \overline{H}_{r,r,1}$ with the determinant of the matrix $\tilde{B}_{r-1}$ which is the matrix gotten from removing the last row and the first column of B_{r-1}. $\tilde{B}_{r-1}$ can be observed to be a diagonal matrix, in which all values along the diagonal are positive. Since the determinant of a diagonal matrix is the product over all values in the diagonal, we have that $|\tilde{B}_{r-1}| > 0$. Therefore

$$\begin{aligned} |B_{r-1}| &= (-1)^{r-1+1} \cdot \overline{H}_{r,r,1} \cdot |\tilde{B}_{r-1}| \\ &= (-1)^{r} \cdot \overline{H}_{r,r,1} \cdot |\tilde{B}_{r-1}| \\ \implies (-1)^{1+r} \cdot |B_{r-1}| & \\ &= (-1)^{1+r} \cdot (-1)^{r} \cdot \overline{H}_{r,r,1} \cdot |\tilde{B}_{r-1}| \\ &= (-1)^{2r+1} \cdot \overline{H}_{r,r,1} \cdot |\tilde{B}_{r-1}| \\ &= -1 \cdot \overline{H}_{r,r,1} \cdot |\tilde{B}_{r-1}| \end{aligned}$$

And since both $\overline{H}_{r,r,1} > 0$ and $|\tilde{B}_{r-1}| > 0$, we have that $(-1)^{1+r} \cdot |B_{r-1}| < 0$, completing the induction step by equation (23). ■