MAT 4376 Foundations of Data Privacy

Rafał Kulik

Department of Mathematics and Statistics (University of Ottawa)

Lecture 17-18-19-20

1. Advances in Differential Privacy
   - Differential privacy - definition
   - Laplace noise and differential privacy
   - (ε, δ)-Differential Privacy (Approximate DP) - definition
   - Composition results
   - Evaluating Differential Privacy in complex algorithms
   - Data utility perspective of DP
   - Issues with DP and how to solve them
   - Other mechanisms (beyond Laplace and Gaussian)
   - DP in machine learning
   - Examples of applications

During these lectures we will learn:

- how to use differential privacy in complex mechanisms;
- data utility perspective of DP;
- issues with DP and how to solve them;
- other mechanisms (beyond Laplace and Gaussian);
- DP in machine learning;
- examples of applications.

Some resources

- Webpage support for the textbook:
  - `https://programming-dp.com/ch7.html`;
  - `https://programming-dp.com/ch8.html`;
  - `https://programming-dp.com/ch9.html`;
- PhD thesis of Devyani Biswal.
- Apple technical manual.

Differential privacy - definition

Definition 1 (Differential Privacy, DP)

Let $\mathbf{X}$ be a database, a random element of $\mathcal{D}$. Let Z be a random element with values in a metric space $\mathcal{E}$. Let $\varepsilon > 0$. A randomized mechanism $Q : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^d$ is ε-differentially private if, $\forall\ \mathbf{x}, \mathbf{y} \in \mathcal{D}$, satisfying $d(\mathbf{x}, \mathbf{y}) = 1$, we have

$$\sup_{B \in \mathcal{B}(\mathbb{R}^d)} \frac{\mathbb{P}(Q(\mathbf{X}, Z) \in B \mid \mathbf{X} = \mathbf{x})}{\mathbb{P}(Q(\mathbf{X}, Z) \in B \mid \mathbf{X} = \mathbf{y})} \leq \mathrm{e}^{\varepsilon}.$$

Laplace noise and differential privacy

Once the definition of differential privacy is introduced, we ask the question: *What random mechanism satisfies differential privacy?*

Theorem 2

Let $\varepsilon > 0$. For any $f : \mathcal{D} \to \mathbb{R}$, the randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + Z$$

with the $\mathrm{Laplace}(0, \frac{\Delta \mathrm{f}}{\varepsilon})$-distributed noise Z is ε-differentially private.

Simple example when DP fails

Example 3

Consider database $\mathbf{x} = (x_1, \ldots, X_n)$ with real-valued entries. Let f be the sample mean and consider $O_f(\mathbf{x}) = f(\mathbf{x}) = \bar{\mathbf{x}}$. Note:

Simple example when DP fails

Example 3

Consider database $\mathbf{x} = (x_1, \ldots, X_n)$ with real-valued entries. Let f be the sample mean and consider $O_f(\mathbf{x}) = f(\mathbf{x}) = \bar{\mathbf{x}}$. Note:

- For any set B such that $\bar{\mathbf{x}} \in B$, we have

$$\mathbb{P}(Q(\mathbf{X}) \in B \mid \mathbf{X} = \mathbf{x}) = 1;$$

- For any set B such that $\bar{\mathbf{x}} \notin B$, we have

$$\mathbb{P}(Q(\mathbf{X}) \in B \mid \mathbf{X} = \mathbf{x}) = 0;$$

Thus, the mechanism cannot be differentially private.

Approximate Differential Privacy

The above example indicates that a normal distribution may not satisfy the definition of differential privacy. This leads to a weaker version of Differential Privacy.

Approximate Differential Privacy

The above example indicates that a normal distribution may not satisfy the definition of differential privacy. This leads to a weaker version of Differential Privacy.

Definition 4 ((ε, δ)-Differential Privacy; Approximate Differential Privacy)

Let $\mathbf{X}$ be a database, a random element of $\mathcal{D}$. Let Z be a random element with values in a metric space $\mathcal{E}$. Let $\varepsilon > 0$ and $\delta \in (0, 1)$. A randomized mechanism $Q : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^d$ is (ε, δ)-differentially private if $\forall\ \mathbf{x}, \mathbf{y} \in \mathcal{D}$, satisfying $d(\mathbf{x}, \mathbf{y}) = 1$ and all $B \in \mathcal{B}(\mathbb{R}^d)$, we have

$$\mathbb{P}(Q(\mathbf{X}, Z) \in B \mid \mathbf{X} = \mathbf{x}) \leq \mathrm{e}^{\varepsilon} \mathbb{P}(Q(\mathbf{X}, Z) \in B \mid \mathbf{X} = \mathbf{y}) + \delta\ .$$

Approximate DP and Gaussian noise

The next result shows that the Gaussian mechanism is (ε, δ)-differentially private.

Approximate DP and Gaussian noise

The next result shows that the Gaussian mechanism is (ε, δ)-differentially private.

Theorem 5

Let $\varepsilon \in (0, 1)$ be arbitrary and $c^2 > 2\ln(1.25/\delta)$. For any $f : \mathcal{D} \to \mathbb{R}$, the randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + Z$$

with the centered Gaussian noise with the parameter $\sigma \geq c\Delta f/\varepsilon$ is (ε, δ)-differentially private.

Composition result I (sequential composition)

Let $Q^{(i)} : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^{d^{(i)}}$, $i = 1, \ldots, m$, be response mechanisms. Define $Q : \mathcal{D} \times \mathcal{E}^m \to \mathbb{R}^{\sum_{i=1}^m d^{(i)}}$ by

$$Q(\mathbf{x}, (\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)})) = (Q^{(1)}(\mathbf{x}, \mathbf{z}^{(1)}), \ldots, Q^{(m)}(\mathbf{x}, \mathbf{z}^{(m)})) \, .$$

Composition result I (sequential composition)

Let $Q^{(i)} : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^{d^{(i)}}$, $i = 1, \ldots, m$, be response mechanisms. Define $Q : \mathcal{D} \times \mathcal{E}^m \to \mathbb{R}^{\sum_{i=1}^m d^{(i)}}$ by

$$Q(\mathbf{x}, (\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)})) = (Q^{(1)}(\mathbf{x}, \mathbf{z}^{(1)}), \ldots, Q^{(m)}(\mathbf{x}, \mathbf{z}^{(m)})) \, .$$

Therefore, we apply multiple queries to the **same** database. As such, the privacy deteriorates.

Composition result I

Lemma 6

Let $Z^{(i)}$, $i = 1, \ldots, m$, be independent random elements. If $Q^{(i)}(\cdot, Z^{(i)})$, $i = 1, \ldots, m$, are $\varepsilon^{(i)}$-differentially private, then $Q(\cdot, (Z^{(1)}, \ldots, Z^{(m)}))$ is $\sum_{i=1}^{m} \varepsilon^{(i)}$-differentially private.

Composition result II (parallel composition)

Assume now that the database $\mathbf{x}$ is decomposed into **disjoint** databases $\mathbf{x}_1, \ldots, \mathbf{x}_m$. Let $Q^{(i)} : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^{d^{(i)}}$, $i = 1, \ldots, m$, be response mechanisms. Define $Q : \mathcal{D} \times \mathcal{E}^m \to \mathbb{R}^{\sum_{i=1}^m d^{(i)}}$ by

$$Q(\mathbf{x}, (z^{(1)}, \ldots, z^{(m)})) = (Q^{(1)}(\mathbf{x}_1, z^{(1)}), \ldots, Q^{(m)}(\mathbf{x}_m, z^{(m)})) \,. \qquad (1)$$

Composition result II (parallel composition)

Assume now that the database $\mathbf{x}$ is decomposed into **disjoint** databases $\mathbf{x}_1, \ldots, \mathbf{x}_m$. Let $Q^{(i)} : \mathcal{D} \times \mathcal{E} \to \mathbb{R}^{d^{(i)}}$, $i = 1, \ldots, m$, be response mechanisms. Define $Q : \mathcal{D} \times \mathcal{E}^m \to \mathbb{R}^{\sum_{i=1}^m d^{(i)}}$ by

$$Q(\mathbf{x}, (z^{(1)}, \ldots, z^{(m)})) = (Q^{(1)}(\mathbf{x}_1, z^{(1)}), \ldots, Q^{(m)}(\mathbf{x}_m, z^{(m)})) \,. \quad (1)$$

Therefore, we apply multiple queries to **disjoint** databases. As such, the privacy does not deteriorate.

Composition result II

Lemma 7

Let $Z^{(i)}$, $i = 1, \ldots, m$, be independent random elements. If $Q^{(i)}(\cdot, Z^{(i)})$, $i = 1, \ldots, m$, are $\varepsilon^{(i)}$-differentially private, then the query (1) is $\max_{i=1,\ldots,m} \varepsilon^{(i)}$-differentially private.

Evaluating Differential Privacy in complex algorithms

We use composition results to get privacy budget in complex algorithms.

Evaluating Differential Privacy in complex algorithms

We use composition results to get privacy budget in complex algorithms.

Example 8 (Multiple querries)

Assume that a $\mathbf{x} = (x_1, \ldots, x_n)$ is a database from a population with range $[0, \Lambda]$. Let $m < n$ and define $\mathbf{x}_1 = (x_1, \ldots, x_m)$, $\mathbf{x}_2 = (x_{m+1}, \ldots, x_n)$.

- Calculate $Q_1(\mathbf{x}, Z_1) = f_1(\mathbf{x_1}) + Z_1$, where f_1 is the sample mean and Z_1 is Laplace with the parameter $\Delta f_1/\varepsilon_1$;
- Calculate $Q_2(\mathbf{x}, Z_2) = f_2(\mathbf{x_1}) + Z_2$, where f_2 is the sample variance and Z_2 is Laplace with the parameter $\Delta f_2/\varepsilon_2$;
- Calculate $Q_3(\mathbf{x}, Z_3) = f_3(\mathbf{x_2}) + Z_3$, where f_3 is the sample median and Z_3 is Laplace with the parameter $\Delta f_3/\varepsilon_3$;
- Release $(Q_1(\mathbf{x}, Z_1), Q_2(\mathbf{x}, Z_2), Q_3(\mathbf{x}, Z_3))$.

Evaluating Differential Privacy in complex algorithms

We use composition results to get privacy budget in complex algorithms.

Example 8 (Multiple querries)

Assume that a $\mathbf{x} = (x_1, \ldots, x_n)$ is a database from a population with range $[0, \Lambda]$. Let $m < n$ and define $\mathbf{x}_1 = (x_1, \ldots, x_m)$, $\mathbf{x}_2 = (x_{m+1}, \ldots, x_n)$.

- Calculate $Q_1(\mathbf{x}, Z_1) = f_1(\mathbf{x_1}) + Z_1$, where f_1 is the sample mean and Z_1 is Laplace with the parameter $\Delta f_1/\varepsilon_1$;
- Calculate $Q_2(\mathbf{x}, Z_2) = f_2(\mathbf{x_1}) + Z_2$, where f_2 is the sample variance and Z_2 is Laplace with the parameter $\Delta f_2/\varepsilon_2$;
- Calculate $Q_3(\mathbf{x}, Z_3) = f_3(\mathbf{x_2}) + Z_3$, where f_3 is the sample median and Z_3 is Laplace with the parameter $\Delta f_3/\varepsilon_3$;
- Release $(Q_1(\mathbf{x}, Z_1), Q_2(\mathbf{x}, Z_2), Q_3(\mathbf{x}, Z_3))$.

1. Is the mechanism differentially private?
2. If so, what is the privacy level?

Evaluating Differential Privacy in complex algorithms

Let's start with sensitivities.

- According to Example on slide 13 (Lectures 13-16), $\Delta f_1 = \Lambda/m$.
- We did not calculate the sensitivity for the sample variance, however, it is easy to get from the previous bullet point that $\Delta f_2 = \Lambda^2/m$.
- According to Example on slide 14 (Lectures 13-16), $\Delta f_3 = \Lambda$.

Evaluating Differential Privacy in complex algorithms

Let's start with sensitivities.

- According to Example on slide 13 (Lectures 13-16), $\Delta f_1 = \Lambda/m$.
- We did not calculate the sensitivity for the sample variance, however, it is easy to get from the previous bullet point that $\Delta f_2 = \Lambda^2/m$.
- According to Example on slide 14 (Lectures 13-16), $\Delta f_3 = \Lambda$.

1. According to Composition Result I, $(Q_1(\mathbf{x}, Z_1), Q_2(\mathbf{x}, Z_2))$ is $\varepsilon_1 + \varepsilon_2$-DP.
2. According to Composition Result II, the final release is $\max\{\varepsilon_1 + \varepsilon_2, \varepsilon_3\}$-DP.

Data utility perspective of DP

Consider a database $\mathbf{x} = (x_1, \ldots, x_n)$ derived from a population $\mathcal{P}$. We can consider two distinct scenarios for a database. It may be treated as fixed, in which case we will denote it by $\mathbf{x}$. Alternatively, it may be treated as random, in which case it will be written as $\mathbf{X}$. If the objective is to estimate a population parameter θ, we can use a point estimator $f(\mathbf{x})$ for a suitable function f.

Data utility perspective of DP

Consider a database $\mathbf{x} = (x_1, \ldots, x_n)$ derived from a population $\mathcal{P}$. We can consider two distinct scenarios for a database. It may be treated as fixed, in which case we will denote it by $\mathbf{x}$. Alternatively, it may be treated as random, in which case it will be written as $\mathbf{X}$. If the objective is to estimate a population parameter θ, we can use a point estimator $f(\mathbf{x})$ for a suitable function f. For example, if θ represents the population mean, then $f(\mathbf{x})$ is the sample mean, $\bar{\mathbf{x}}$. In the context of privacy, we consider the randomized output perturbation mechanism, defined as follows:

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + Z \ ,$$

where Z is a random variable (or a random vector if f has values in $\mathbb{R}^d$) **independent of the database**. In other words, the estimator of θ is **perturbed by a random noise**. The additional noise Z contributes to a decline in the utility of the data.

Data utility perspective of DP

For example, if $Z \sim \text{Lap}(\Delta f/\varepsilon)$, we can compare the **mean square error (MSE)** of both the original and privatized estimators. The MSE of the estimator θ is given by:

$$\text{MSE}(f(\mathbf{X})) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] \,,$$

$$\text{MSE}(O_f(\mathbf{X}, Z)) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] + 2(\Delta f/\varepsilon)^2 \,.$$

The term $2(\Delta f/\varepsilon)^2$ is the additional contribution from the privatization.

Data utility perspective of DP

For example, if $Z \sim \text{Lap}(\Delta f/\varepsilon)$, we can compare the **mean square error (MSE)** of both the original and privatized estimators. The MSE of the estimator θ is given by:

$$\text{MSE}(f(\mathbf{X})) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] \,,$$

$$\text{MSE}(O_f(\mathbf{X}, Z)) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] + 2(\Delta f/\varepsilon)^2 \,.$$

The term $2(\Delta f/\varepsilon)^2$ is the additional contribution from the privatization. If the population has a range of $[0, \Lambda]$, then the sensitivity is given by $\Delta f = \frac{\Lambda}{n}$ Consequently, the variance of the noise is given by:

$$\text{Var}(Z) = 2\left(\frac{\Lambda^2}{n^2\varepsilon^2}\right) \,. \tag{2}$$

If Λ is large, a considerable amount of noise is introduced. Additionally, the range may be either unknown or infinite.

Data utility perspective of DP

For example, if $Z \sim \text{Lap}(\Delta f/\varepsilon)$, we can compare the **mean square error (MSE)** of both the original and privatized estimators. The MSE of the estimator θ is given by:

$$\text{MSE}(f(\mathbf{X})) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] \,,$$

$$\text{MSE}(O_f(\mathbf{X}, Z)) = \mathbb{E}[(f(\mathbf{X}) - \theta)^2] + 2(\Delta f/\varepsilon)^2 \,.$$

The term $2(\Delta f/\varepsilon)^2$ is the additional contribution from the privatization. If the population has a range of $[0, \Lambda]$, then the sensitivity is given by $\Delta f = \frac{\Lambda}{n}$ Consequently, the variance of the noise is given by:

$$\text{Var}(Z) = 2\left(\frac{\Lambda^2}{n^2\varepsilon^2}\right) \,. \tag{2}$$

If Λ is large, a considerable amount of noise is introduced. Additionally, the range may be either unknown or infinite. Consequently, one of the most important practical issues is to **reduce the amount of noise added while maintaining the same level of privacy.**

Issues with DP and how to solve them

We will present several issues and indicate some solutions. For each propose solution, we need:

- state properly the proposed algorithm;
- indicate if this algorithm is DP;
- indicate its data utility;
- compare to the classical DP.

Issue 1: too much noise added

The classical definition of differential privacy only allows for the use of Laplace noise. From the statistical inference point of view, it may be desirable to use a normal noise. The Gaussian output perturbation mechanism fulfills the (ε, δ)-differential privacy whenever the variance $\sigma^2 > 2\Delta f \ln(1.25/\delta)/\varepsilon$. Now, the parameter δ has to be small. This implies that the variance must be substantial, which, in turn, necessitates the addition of a considerable amount of noise.

Addressing Issue 1 - Mixed Noise Mechanism

Definition 9

Let $f : \mathcal{D} \to \mathbb{R}_+$. Let $t_0 > 0$ and $\sigma_{MNM} > 0$. We call

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + \mathcal{N}(0, \sigma^2_{MNM})\mathbb{1}\{\Delta f \leq t_0\} + \text{Lap}(\Delta f/\varepsilon)\mathbb{1}\{\Delta f > t_0\}\ ,$$

the *Mixed Noise Mechanism*.

Addressing Issue 1 - Mixed Noise Mechanism

Definition 9

Let $f : \mathcal{D} \to \mathbb{R}_+$. Let $t_0 > 0$ and $\sigma_{MNM} > 0$. We call

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + \mathcal{N}(0, \sigma^2_{MNM})\mathbb{1}\{\Delta f \leq t_0\} + \text{Lap}(\Delta f/\varepsilon)\mathbb{1}\{\Delta f > t_0\}\ ,$$

the *Mixed Noise Mechanism*.

In other words, when the sensitivity Δf is large, we choose to add a Laplace noise, while when Δf is small, we choose to add a Gaussian noise.

Addressing Issue 1 - Mixed Noise Mechanism

Definition 9

Let $f : \mathcal{D} \to \mathbb{R}_+$. Let $t_0 > 0$ and $\sigma_{MNM} > 0$. We call

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + \mathcal{N}(0, \sigma^2_{MNM})\mathbb{1}\{\Delta f \leq t_0\} + \text{Lap}(\Delta f/\varepsilon)\mathbb{1}\{\Delta f > t_0\}\ ,$$

the *Mixed Noise Mechanism*.

In other words, when the sensitivity Δf is large, we choose to add a Laplace noise, while when Δf is small, we choose to add a Gaussian noise. We need to answer: 1) Is the algorithm DP? 2) What are the parameters t_0 and σ_{MNM}?

Addressing Issue 1 - Mixed Noise Mechanism

Theorem 10

Let $\varepsilon > 0$, $\delta \in (0,1)$. The Mixed Noise Mechanism

$$O_f(\mathbf{x}) = f(\mathbf{x}) + \mathcal{N}(0, \sigma^2_{MNM})\mathbb{1}\{\Delta f \leq t_0\} + Lap(\Delta f/\varepsilon)\mathbb{1}\{\Delta f > t_0\}\ ,$$

with the threshold $t_0 = \sqrt{\pi}\delta\epsilon/2$ and $\sigma^2_{MNM} = 2\left(\frac{\Delta f}{\varepsilon}\right)^2$ is (ε, δ)-differentially private.

Addressing Issue 1 - Mixed Noise Mechanism

- The variance of the Gaussian is the same as the Laplace one. It is much smaller than for the Gaussian mechanism.
- Thus, since

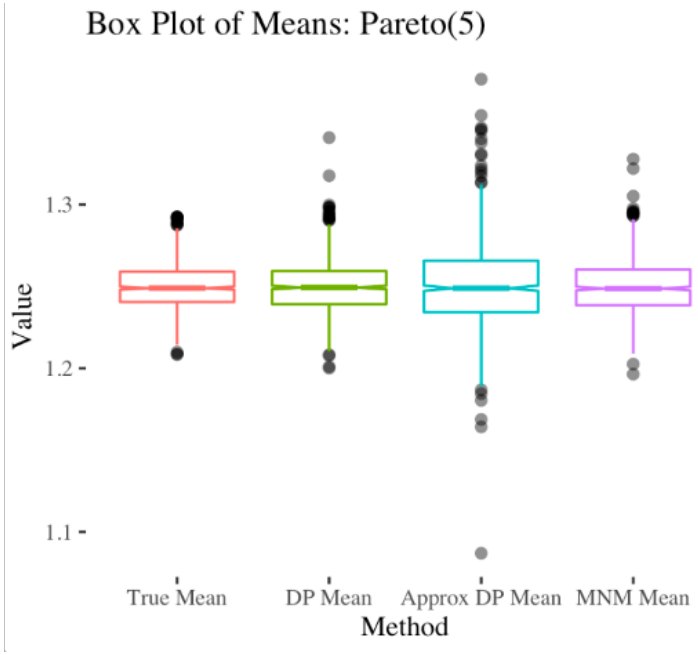
$$\mathrm{Var}\left(\mathcal{N}(0,\sigma^2_{MNM})\mathbb{1}\{\Delta f \leq t_0\} + \mathrm{Lap}(\Delta f/\varepsilon)\mathbb{1}\{\Delta f > t_0\}\right) = 2\left(\frac{\Delta f}{\varepsilon}\right)^2,$$

the data utility of the MNM is the same as of Laplace mechanism.

Addressing Issue 1 - Mixed Noise Mechanism

Example 11

In this example we are going to simulate data from the Pareto distribution with the density $\alpha x^{-\alpha-1}$, $x > 1$. We calculate sample mean and noisy sample mean using different mechanisms.



Issue 2: using local sensitivity

As previously stated, utilizing the global sensitivity may not be a practical option. Conversely, the local sensitivity may be employed, which is readily computable for each query and each data set. Nevertheless, the utilization of the local sensitivity may violate differential privacy.

Issue 2: using local sensitivity

As previously stated, utilizing the global sensitivity may not be a practical option. Conversely, the local sensitivity may be employed, which is readily computable for each query and each data set. Nevertheless, the utilization of the local sensitivity may violate differential privacy.

Example 12 (The local sensitivity is not differentially private)

Assume that the database is real-valued and the entries x_i come from a distribution with the support $[0, \Lambda]$, $0 < \Lambda < \infty$. Assume that n is odd. Let $f(\mathbf{x}) = \text{median}(x_1, \ldots, x_n)$. Let $m = \frac{n+1}{2}$ be the rank of the median element. The global sensitivity of the median is then given by $\Delta f = \Lambda$. The addition of noise in accordance with this sensitivity will result in the destruction of data utility.

Issue 2: using local sensitivity

As previously stated, utilizing the global sensitivity may not be a practical option. Conversely, the local sensitivity may be employed, which is readily computable for each query and each data set. Nevertheless, the utilization of the local sensitivity may violate differential privacy.

Example 12 (The local sensitivity is not differentially private)

Assume that the database is real-valued and the entries x_i come from a distribution with the support $[0, \Lambda]$, $0 < \Lambda < \infty$. Assume that n is odd. Let $f(\mathbf{x}) = \text{median}(x_1, \ldots, x_n)$. Let $m = \frac{n+1}{2}$ be the rank of the median element. The global sensitivity of the median is then given by $\Delta f = \Lambda$. The addition of noise in accordance with this sensitivity will result in the destruction of data utility.
By contrast, the local sensitivity is given by $\Delta^{(\text{local})} f(\mathbf{x}) = \max(x_m - x_{m-1}, x_{m+1} - x_m)$. This approach yields less noise, but may result in a violation of $(\varepsilon, \delta)-$differentially privacy.

Issue 2: using local sensitivity

Indeed, consider two data sets.

- Dataset 1: $x_1 = \cdots = x_m = 0$ and $x_{m+1} = \cdots = x_n = \Lambda$;
- Dataset 2: $x_1 = \cdots = x_{m+1} = 0$ and $x_{m+2} = \cdots = x_n = \Lambda$.

Issue 2: using local sensitivity

Indeed, consider two data sets.

- Dataset 1: $x_1 = \cdots = x_m = 0$ and $x_{m+1} = \cdots = x_n = \Lambda$;
- Dataset 2: $x_1 = \cdots = x_{m+1} = 0$ and $x_{m+2} = \cdots = x_n = \Lambda$.

It can be noted that, in both datasets, the median is equal to zero. In the first data set, the local sensitivity is also equal to Λ, whereas in the second dataset, the local sensitivity is zero. Additionally, the Hamming distance between these two datasets is 1. Thus, if the mechanism $f(\mathbf{x}) + Z = 0 + Z$, with Z representing a Laplace noise with a parameter proportional to the local sensitivity, then in the second scenario, no noise will be added.

Issue 2: using local sensitivity

Indeed, consider two data sets.

- Dataset 1: $x_1 = \cdots = x_m = 0$ and $x_{m+1} = \cdots = x_n = \Lambda$;
- Dataset 2: $x_1 = \cdots = x_{m+1} = 0$ and $x_{m+2} = \cdots = x_n = \Lambda$.

It can be noted that, in both datasets, the median is equal to zero. In the first data set, the local sensitivity is also equal to Λ, whereas in the second dataset, the local sensitivity is zero. Additionally, the Hamming distance between these two datasets is 1. Thus, if the mechanism $f(\mathbf{x}) + Z = 0 + Z$, with Z representing a Laplace noise with a parameter proportional to the local sensitivity, then in the second scenario, no noise will be added. Therefore, the probability of receiving a non-zero response to the randomized query is zero for the second data set and non-zero for the first data set. As a result, the protocol does not satisfy the requirements to be considered $(\varepsilon, \delta)-$ differentially private.

Addressing Issue 2 - solving sensitivity issue

There are no satisfactory solutions to this problem (that is, the problem of using the appropriate sensitivity).

- One potential solution is obtained via the smooth sensitivity.
- Another approach is through the *general sensitivity*. Refer to PhD thesis of Devyani Biswal.

Smooth sensitivity

The concept of **smooth sensitivity** lies between the local and the global sensitivity. As in the case of local sensitivity, it can be computed for the given database and the query. As in the case of the global sensitivity, it yields an appropriate version of differential privacy.

Smooth sensitivity

The concept of **smooth sensitivity** lies between the local and the global sensitivity. As in the case of local sensitivity, it can be computed for the given database and the query. As in the case of the global sensitivity, it yields an appropriate version of differential privacy. Recall that a database $\mathbf{x}$ has values in $\mathcal{D}$. Recall the notation $\Delta^{(\mathrm{local})} f(\mathbf{x})$ for the local sensitivity of a query f. Recall also that the global sensitivity is $\Delta f = \max_{\mathbf{x} \in \mathcal{D}} \Delta^{(\mathrm{local})} f(\mathbf{x})$. We always have $\Delta^{(\mathrm{local})} f(\mathbf{x}) \leq \Delta f$.

Smooth sensitivity

Definition 13 (A Smooth Bound on Local Sensitivity)

Let $\beta > 0$. A function $S : \mathcal{D} \to \mathbb{R}^+$ is a β-smooth upper bound on the local sensitivity of f if it satisfies the following requirements:

$$\forall \mathbf{x} \in \mathcal{D}: \quad S(\mathbf{x}) \geq \Delta^{(\text{local})} f(\mathbf{x});$$

$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}, d(\mathbf{x}, \mathbf{y}) = 1: \quad S(\mathbf{x}) \leq e^{\beta} S(\mathbf{y}).$$

Smooth sensitivity

Definition 13 (A Smooth Bound on Local Sensitivity)

Let $\beta > 0$. A function $S : \mathcal{D} \to \mathbb{R}^+$ is a β-smooth upper bound on the local sensitivity of f if it satisfies the following requirements:

$$\forall \mathbf{x} \in \mathcal{D}: \quad S(\mathbf{x}) \geq \Delta^{(\text{local})} f(\mathbf{x});$$
$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}, d(\mathbf{x}, \mathbf{y}) = 1: \quad S(\mathbf{x}) \leq e^{\beta} S(\mathbf{y}).$$

The last property yields the following result for any two neighbouring databases:

$$e^{-\beta} \leq \frac{S(\mathbf{x})}{S(\mathbf{y})} \leq e^{\beta} .$$

Smooth sensitivity

Definition 13 (A Smooth Bound on Local Sensitivity)

Let $\beta > 0$. A function $S : \mathcal{D} \to \mathbb{R}^+$ is a β-smooth upper bound on the local sensitivity of f if it satisfies the following requirements:

$$\forall \mathbf{x} \in \mathcal{D}: \quad S(\mathbf{x}) \geq \Delta^{(\text{local})} f(\mathbf{x});$$
$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}, d(\mathbf{x}, \mathbf{y}) = 1: \quad S(\mathbf{x}) \leq e^{\beta} S(\mathbf{y}).$$

The last property yields the following result for any two neighbouring databases:

$$e^{-\beta} \leq \frac{S(\mathbf{x})}{S(\mathbf{y})} \leq e^{\beta} .$$

The constant function $S(\mathbf{x}) = \Delta f$ (the global sensitivity), represents the 0-smooth upper bound. On the other hand, when $\beta > 0$, the function S serves as a highly conservative upper bound on the local sensitivity of f.

Smooth sensitivity

Definition 14 (Smooth Sensitivity)

For $\beta > 0$, the β-smooth sensitivity of f is

$$S^*_{f,\beta}(\mathbf{x}) = \max_{\mathbf{y}\in\mathcal{D}} \left(\Delta^{(local)} f(\mathbf{y}) \cdot e^{-\beta d(\mathbf{x},\mathbf{y})} \right) . \tag{3}$$

Smooth sensitivity

The following lemma demonstrates that the function $S^*_{f,\beta}$ fulfills the conditions in 13. It can be shown that this function represents the optimal β-smooth upper bound.

Lemma 15

$S^*_{f,\beta}$ is a β-smooth upper bound on the local sensitivity. Additionally, $S^*_{f,\beta}(\mathbf{x}) \leq S(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{D}$ for every β-smooth upper bound S on the local sensitivity $\Delta^{(\text{local})} f(\mathbf{x})$.

Smooth sensitivity

Does smooth sensitivity lead to Differential Privacy?

Smooth sensitivity

Does smooth sensitivity lead to Differential Privacy?

Proposition 16

Let $\varepsilon > 0$ and $\delta' \in (0,1)$. Let $f : \mathcal{D} \to \mathbb{R}_+$ and let $S : \mathcal{D} \to \mathbb{R}_+$ be a β-smooth upper bound on the local sensitivity of f. If $\beta \leq \frac{\varepsilon}{2\ln(2/\delta')}$ and $\delta' \in (0,1)$, the randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + Z$$

with the $Laplace(0, 2\frac{S(\mathbf{x})}{\varepsilon})$-distributed noise Z. Then, for $\mathbf{x} \sim \mathbf{y}$ and all Borel sets B,

$$\mathbb{P}(O_f(\mathbf{x}, Z) \in B) \leq e^{\varepsilon}\mathbb{P}(O_f(\mathbf{y}, Z) \in B) + \frac{\delta'}{2}(e^{\varepsilon/2} + 1) .$$

Smooth sensitivity

Corollary 17

Let $\delta \in [0, 1]$. Under the previous conditions, the algorithm is (ε, δ)-differentially private whenever

$$\frac{\delta'}{2}(e^{\varepsilon/2} + 1) \leq \delta < 1 \, .$$

Smooth sensitivity

Corollary 17

Let $\delta \in [0, 1]$. Under the previous conditions, the algorithm is (ε, δ)-differentially private whenever

$$\frac{\delta'}{2}(e^{\varepsilon/2} + 1) \leq \delta < 1 \, .$$

Thus, the addition of Laplace noise with smooth sensitivity yields approximate differential privacy instead of ε-differential privacy. Two questions arise here.

- *Do distributions exist that achieve ε-differential privacy with the smooth sensitivity?*
- *Can normal noise with the smooth sensitivity be used?*

Smooth sensitivity

Both questions can be answered affirmatively. The following observations are presented without a formal proof. Let S be a β-smooth sensitivity.

- Let $\varepsilon > 0$. Let $\beta \leq \frac{\varepsilon}{2(\gamma+1)}$ and $\gamma > 1$. Assume that Z is sampled from the density $h(z) \propto \frac{1}{1+|z|^\gamma}$. Let $c_\varepsilon(\mathbf{x}) = \frac{2(\gamma+1)S(\mathbf{x})}{\varepsilon}$. The randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + c_\varepsilon(\mathbf{x})Z$$

  is ε- differentially private.

Smooth sensitivity

- Let $\varepsilon > 0$ and $\delta \in (0, 1)$. Take $\beta = \frac{\varepsilon}{2\ln(2/\delta)}$. Let $d_\varepsilon(\mathbf{x}) = 5\sqrt{2\ln(2/\delta)}\frac{S(\mathbf{x})}{\varepsilon}$. Assume that Z is standard normal. The randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + d_\varepsilon(\mathbf{x})Z$$

is (ε, δ)-differentially private.

Smooth sensitivity

- Let $\varepsilon > 0$ and $\delta \in (0,1)$. Take $\beta = \frac{\varepsilon}{2\ln(2/\delta)}$. Let $d_\varepsilon(\mathbf{x}) = 5\sqrt{2\ln(2/\delta)}\frac{S(\mathbf{x})}{\varepsilon}$. Assume that Z is standard normal. The randomized output perturbation mechanism

$$O_f(\mathbf{x}, Z) = f(\mathbf{x}) + d_\varepsilon(\mathbf{x})Z$$

is (ε, δ)-differentially private. The last item should be compared to 5, where we add the noise of the form $c_\varepsilon Z$, where $c_\varepsilon > \sqrt{2\ln(1.25/\delta)}\Delta f/\varepsilon$. Since the smooth sensitivity will be close to the global sensitivity, Δf, we observe that for small δ, it is necessary to add more noise when using the smooth sensitivity than when using the global sensitivity.

Smooth sensitivity - computation

As defined in Definition 14, the smooth sensitivity is not computable, since it uses all of the possible databases **y**. For some specific queries f, the smooth sensitivity can be computed approximately.

Definition 18

Let **x** be a database. The local sensitivity of f at **distance k** is

$$A_f^{(k)}(\mathbf{x}) = \max_{\mathbf{y}\in\mathcal{D}:d(\mathbf{x},\mathbf{y})\leq k} \Delta^{(\text{local})} f(\mathbf{y}) \, .$$

We would like to point out that to compute $A_f^{(k)}(\mathbf{x})$ we still need to know all the possible databases **y**. Note that

$$A_f^{(0)}(\mathbf{x}) = \Delta^{(\text{local})} f(\mathbf{x}) \, .$$

Smooth sensitivity - computation

Now the smooth sensitivity can be approximated in terms of $A_f^{(k)}$:

$$\widetilde{S}^*_{f,\beta}(\mathbf{x}) = \max_{k=0,1,\ldots,n} e^{-k\beta} \left(\max_{\mathbf{y}:d(\mathbf{x},\mathbf{y})=k} \Delta^{(\text{local})} f(\mathbf{y}) \right)$$

$$= \max_{k=0,1,\ldots,n} e^{-k\beta} A_f^{(k)}(\mathbf{x}) = \max \left(\Delta^{(\text{local})} f(\mathbf{x}), \max_{k=1,\ldots,n} e^{-k\beta} A_f^{(k)}(\mathbf{x}) \right) .$$

Smooth sensitivity - computation

Example 19

For $f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$, the local sensitivity is $\frac{1}{n}\max|x_i|$. If $d(\mathbf{x},\mathbf{y}) = 1$,

$$A^{(1)}(\mathbf{x}) = \max\left|\frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{n}\sum_{i=1, i\neq j}^{n} x_i - \frac{1}{n}y_j\right| = \frac{1}{n}\max|x_j - y_j|\ ,$$

where the maximum is understood to be taking the maximum over all rows j in the database and all possible entries y_j coming from the original population.

Smooth sensitivity - computation

Example 19

For $f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i$, the local sensitivity is $\frac{1}{n}\max|x_i|$. If $d(\mathbf{x},\mathbf{y}) = 1$,

$$A^{(1)}(\mathbf{x}) = \max\left|\frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{n}\sum_{i=1, i\neq j}^{n} x_i - \frac{1}{n}y_j\right| = \frac{1}{n}\max|x_j - y_j|\ ,$$

where the maximum is understood to be taking the maximum over all rows j in the database and all possible entries y_j coming from the original population. If the population is $[0,\Lambda]$, then $A^{(1)}(\mathbf{x}) = \max_j\{\max\{x_j, |x_j-\Lambda|\}\}/n$. Next,

$$A^{(k)}(\mathbf{x}) = \max\left|\frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{n}\sum_{i=1, i\neq J_k}^{n} x_i - \frac{1}{n}\sum_{i\in J_k} y_i\right|\ ,$$

where the max is taken again over all k-tuples $J_k = \{(j_1,\ldots,j_k)\in\{1,\ldots,n\}^k\}$ and all the entries y_j.

Smooth sensitivity - computation

Thus, the smooth sensitivity can be calculated when sampling from a bounded population. However, it is not possible to compute the smooth sensitivity in an unbounded case. Therefore, the smooth sensitivity is not a particularly useful measure in the context of the mean query.

Smooth sensitivity - computation

Thus, the smooth sensitivity can be calculated when sampling from a bounded population. However, it is not possible to compute the smooth sensitivity in an unbounded case. Therefore, the smooth sensitivity is not a particularly useful measure in the context of the mean query.
The next example illustrates the utility of the smooth sensitivity in addressing median queries.

Example 20

For the function $f(\boldsymbol{x}) = \text{median}(x_1, \ldots, x_n)$ (assuming that n is odd and $m = \frac{n+1}{2}$), the local and global sensitivity are, respectively:

$$\Delta^{(\text{local})} f(\boldsymbol{x}) = \max(x_{m+1} - x_m, x_m - x_{m-1}) \ , \ \Delta f = \Lambda_2 + \Lambda_1 \ .$$

Then

$$\widetilde{S}^*_{f,\beta}(\boldsymbol{x}) = \max_{k=0,\ldots,n} \left(e^{-k\beta} \cdot \max_{t=0,\ldots,k+1} (x_{m+t} - x_{m+t-k-1}) \right) \ . \qquad (4)$$

Smooth sensitivity - computation

Example 21

Consider again the following example:

- Dataset 1: $x_1 = \cdots = x_m = 0$ and $x_{m+1} = \cdots = x_n = \Lambda$;
- Dataset 2: $x_1 = \cdots = x_{m+1} = 0$ and $x_{m+2} = \cdots = x_n = \Lambda$.

In light of Example 12, we want to check if the smooth sensitivity is zero for one dataset and non-zero for another data. It turns out that for both databases, the values within the brackets of equation (4) alternate between 0 and Λ, and thus the issue previously encountered in Example 12 is no longer present.

Smooth sensitivity - summary

- Smooth sensitivity lies between the local and the global one (hence, less noise to be added as in the global case);
- It gives DP as opposed to the local sensitivity (there is a price to pay, possible decline in privacy).
- A smooth sensitivity can be calculated based on the database, whereas the global sensitivity cannot be.
- There are still problems with unbounded population.

Issue 3 - unrealistic output

Example 22

Consider a hypothetical scenario in which a census dataset is being queried with the objective of determining the number of individuals born on Mars. The addition of noise from a Laplace mechanism with variance $2/\varepsilon^2$ will satisfy differential privacy. Although the actual number of people born on Mars is zero (at least for the time being), it is necessary to add noise to ensure the privacy of future human martians. Successive outputs from the Laplace mechanism could be: $-1.71, 2.31, -1.20, 0.652$. However bizarre the query, negative outputs are patently illogical and inconsistent. By the symmetry of the Laplace distribution, on average 50% of the outputs will be negative.

Issue 3 - unrealistic output

Example 22

Consider a hypothetical scenario in which a census dataset is being queried with the objective of determining the number of individuals born on Mars. The addition of noise from a Laplace mechanism with variance $2/\varepsilon^2$ will satisfy differential privacy. Although the actual number of people born on Mars is zero (at least for the time being), it is necessary to add noise to ensure the privacy of future human martians. Successive outputs from the Laplace mechanism could be: $-1.71, 2.31, -1.20, 0.652$. However bizarre the query, negative outputs are patently illogical and inconsistent. By the symmetry of the Laplace distribution, on average 50% of the outputs will be negative.
The example seems to be strange, but it was exactly the problem with US census.

Issue 3 - unrealistic output

Example 23

Let us suppose that our interest lies in the noisy standard deviation. Two methods exist for achieving this result, and a bizarre occurrence will be presented as an example. The noisy sample variance (nSV) and the noisy standard deviation (nSD) can be expressed as follows:

$$\text{nSV} = S^2 + \text{Lap}(\Lambda^2/n\varepsilon)\ ,\ \ \text{nSD} = \sqrt{S^2 + \text{Lap}(\Lambda^2/n\varepsilon)}\ .$$

If we are interested in the nSD, it is possible that the nSV may assume a negative value. In such an instance, it is not feasible to calculate the nSD. If we consider the standard deviation as the direct mechanism, we can achieve the noisy SD through the following mechanism:

$$\text{nSD} = S + \text{Lap}(\Lambda/n\varepsilon)\ .$$

Addressing Issue 3 - Bounded Laplace mechanism

We are interested in queries $Q : \mathcal{D} \times \mathcal{E} \to \mathrm{Dom}$ on datasets $\mathbf{x} \in \mathcal{D}$ mapping to a finite domain $\mathrm{Dom} = [l, u] \subset \mathbb{R}$ ($l < u$, both finite). We are concerned only with output perturbation mechanisms:

$$O_f(\mathbf{x}, z) = f(\mathbf{x}) + z\ , \quad \mathbf{x} \in \mathcal{D}, z \in \mathbb{R}\ .$$

When the noise is $\mathrm{Laplace}(0, \Delta f/\varepsilon)$, pure differential privacy holds.

Addressing Issue 3 - Bounded Laplace mechanism

We are interested in queries $Q : \mathcal{D} \times \mathcal{E} \to \mathrm{Dom}$ on datasets $\mathbf{x} \in \mathcal{D}$ mapping to a finite domain $\mathrm{Dom} = [l, u] \subset \mathbb{R}$ ($l < u$, both finite). We are concerned only with output perturbation mechanisms:

$$O_f(\mathbf{x}, z) = f(\mathbf{x}) + z \ , \quad \mathbf{x} \in \mathcal{D}, z \in \mathbb{R} \ .$$

When the noise is $\mathrm{Laplace}(0, \Delta f/\varepsilon)$, pure differential privacy holds. Set $Q_{f(\mathbf{x})} := Q(\mathbf{x}, Z)$, then the random variable $Q_{f(\mathbf{x})}$ has the density

$$g_{Q_{f(\mathbf{x})}}(v) = \frac{1}{2b} \exp\left(-\frac{|v - f(\mathbf{x})|}{b}\right) \ , \quad b = \frac{\Delta f}{\varepsilon} \ , \quad v \in \mathbb{R}.$$

The idea of the bounded Laplace mechanism is to restrict the domain of the density above to $v \in \mathrm{Dom}$. This will correspond to the randomized query $Q(\mathbf{x}, Z) = f(\mathbf{x}) + Z$ being restricted to the domain Dom. This is important to note that the restriction will be on $f(\mathbf{x}) + Z$, not on Z. As such, the restriction will depend on the database $\mathbf{x}$. This creates some complications.

Addressing Issue 3 - Bounded Laplace mechanism

Definition 24 (Bounded Laplace Mechanism)

Given $b > 0$ and $\mathrm{Dom} \subset \mathbb{R}$, the bounded Laplace mechanism $Q_{f(\mathbf{x})}$ is given by its probability density function:

$$g_{Q_f(\mathbf{x})}(v) = \begin{cases} 0, & \text{if } v \notin D \\ \frac{1}{C_{f(\mathbf{x})}} \frac{1}{2b} e^{-\frac{|v-f(\mathbf{x})|}{b}}, & \text{if } v \in D, \end{cases}$$

where $C_{f(\mathbf{x})} = \int_D \frac{1}{2b} e^{-\frac{|v-f(\mathbf{x})|}{b}} dv$ is a normalization constant.

Addressing Issue 3 - Bounded Laplace mechanism

Example 25

Assume that $\boldsymbol{x}$ describes age. It is reasonable to assume that age is between 0 and 110. Let $f(\boldsymbol{x}) = \min_j x_j$. Assume, the query returns a response of 10. Adding an unbounded Laplace noise (with the real domain) may lead to an unrealistic negative age. Instead, the bounded Laplace mechanism will add Laplace noise with the domain $[-10, 100]$. The resulting output will belong to the prescribed range $[0, 110]$.

Addressing Issue 3 - Bounded Laplace mechanism

In classical differential privacy, we always add $\mathrm{Laplace}(0, \Delta f/\varepsilon)$ noise, regardless of the output of the query, $f(\mathbf{x})$. Here, the added Laplace noise will depend on the query and hence on the database. As such, the bounded Laplace mechanism does not consistently satisfy differential privacy when utilizing parameters derived from the pure Laplace mechanism. This is due to the fact that the output is contingent upon the original database (in comparison to the scenario of the Laplace mechanism with local sensitivity).

Addressing Issue 3 - Bounded Laplace mechanism

Nevertheless, we can provide an answer when we preserve approximate differential privacy. To this end, it is necessary to note that the data-dependent constant $C_{f(\mathbf{x})}$ has the form:

$$C_q := 1 - \frac{1}{2}\left(\exp(-(q-l)/b) + \exp(-(u-q))/b)\right) \ ,$$

where $q = f(\mathbf{x})$. Define

$$\Delta C = \frac{C_{l+\Delta f}}{C_l} \ .$$

Unlike the constant $C_{f(\mathbf{x})}$, the new constant ΔC does not depend on the database (but it depends on b).

Theorem 26

The bounded Laplace mechanism is (ε, δ)- differentially private whenever

$$b \geq \frac{\Delta f}{\varepsilon - \log \Delta C - \log(1-\delta)}.$$

Exponential mechanism

Assume what we have a dataset $\mathbf{x} = (x_1, \ldots, x_n)$. Let $\mathcal{Q}$ be a set of objects. Let $f : \mathcal{D} \times \mathcal{Q} \to \mathbb{R}$. We want to provide an output q that maximizes $f(\mathbf{x}, q)$:

$$\operatorname{argmax}_{q \in \mathcal{Q}} \{f(\mathbf{x}, q)\}.$$

Goal: Privately select an object with the highest score.

Exponential mechanism

Assume what we have a dataset $\mathbf{x} = (x_1, \ldots, x_n)$. Let $\mathcal{Q}$ be a set of objects. Let $f : \mathcal{D} \times \mathcal{Q} \to \mathbb{R}$. We want to provide an output q that maximizes $f(\mathbf{x}, q)$:

$$\operatorname{argmax}_{q \in \mathcal{Q}} \{f(\mathbf{x}, q)\}.$$

Goal: Privately select an object with the highest score.
Algorithm: Let Q be a random variable defined on $\mathcal{Q}$ such that $\mathbb{P}(Q = q)$ is proportional to $\exp\left(\frac{\varepsilon}{2\Delta} f(\mathbf{x}, q)\right)$.

Exponential mechanism

Assume what we have a dataset $\mathbf{x} = (x_1, \ldots, x_n)$. Let $\mathcal{Q}$ be a set of objects. Let $f : \mathcal{D} \times \mathcal{Q} \to \mathbb{R}$. We want to provide an output q that maximizes $f(\mathbf{x}, q)$:

$$\operatorname{argmax}_{q \in \mathcal{Q}} \{f(\mathbf{x}, q)\}.$$

Goal: Privately select an object with the highest score.
Algorithm: Let Q be a random variable defined on $\mathcal{Q}$ such that $\mathbb{P}(Q = q)$ is proportional to $\exp\left(\frac{\varepsilon}{2\Delta} f(\mathbf{x}, q)\right)$.
Sensitivity of the exponential mechanism is

$$\Delta := \sup_{q \in \mathcal{Q}} \sup_{\mathbf{x}, \mathbf{y} : d(\mathbf{x}, \mathbf{y}) = 1} |f(\mathbf{x}, q) - f(\mathbf{y}, q)|.$$

Theorem 27

The exponential mechanism is ε-differentially private.

Exponential mechanism - data utility

Important: The output Q is random, but belongs to the set $\mathcal{Q}$. In general, it is hard to provide formulas for data utility. Let $f^*(\mathbf{x}) = \sup_{q \in \mathcal{Q}} f(\mathbf{x}, q)$.

Lemma 28

Assume that $\mathcal{Q}$ is finite of size d. Then for every $t > 0$,

$$\mathbb{P}\left(f(\mathbf{x}, Q) < f^*(\mathbf{x}) - \frac{2\Delta(\ln(d) + t)}{\varepsilon} \right) \leq e^{-t},$$

$$\mathbb{E}[f(\mathbf{x}, Q)] \geq f^*(\mathbf{x}) - \frac{2\Delta(\ln(d) + t)}{\varepsilon}.$$

Exponential mechanism - data utility

Proof: To make the proof more readable, we'll drop the $\mathbf{x}$ symbol in the score function f. Thus, we will write $f(q)$ instead of $f(\mathbf{x}, q)$. Let $\psi_t = \frac{2\Delta(\ln(d)+t)}{\varepsilon}$. Consider "bad outputs"

$$\mathcal{B}_t = \{q \in \mathcal{Q} : f(q) \leq f^* - \psi_t\}, \quad \mathcal{G}_t = \mathcal{B}_t^c.$$

We want to prove $\mathbb{P}(B_t) \leq e^{-t}$.

Exponential mechanism - data utility

Proof: To make the proof more readable, we'll drop the $\mathbf{x}$ symbol in the score function f. Thus, we will write $f(q)$ instead of $f(\mathbf{x}, q)$. Let $\psi_t = \frac{2\Delta(\ln(d)+t)}{\varepsilon}$. Consider "bad outputs"

$$\mathcal{B}_t = \{q \in \mathcal{Q} : f(q) \leq f^* - \psi_t\}, \quad \mathcal{G}_t = \mathcal{B}_t^c.$$

We want to prove $\mathbb{P}(B_t) \leq e^{-t}$. Write $\mathbb{P}(Q = q) = Ce^{\frac{\varepsilon}{2\Delta} f(q)}$. Let q^* be an output with score f^*. Then

$$\mathbb{P}(B_t) = \sum_{q \in B_t} \mathbb{P}(Q = q) \leq \frac{\sum_{q \in B_t} \mathbb{P}(Q = q)}{\mathbb{P}(Q = q^*)} = \frac{\sum_{q \in B_t} e^{\frac{\varepsilon}{2\Delta} f(q)}}{e^{\frac{\varepsilon}{2\Delta} f^*}}.$$

Exponential mechanism - data utility

Proof: To make the proof more readable, we'll drop the $\mathbf{x}$ symbol in the score function f. Thus, we will write $f(q)$ instead of $f(\mathbf{x}, q)$. Let $\psi_t = \frac{2\Delta(\ln(d)+t)}{\varepsilon}$. Consider "bad outputs"

$$\mathcal{B}_t = \{q \in \mathcal{Q} : f(q) \leq f^* - \psi_t\}, \quad \mathcal{G}_t = \mathcal{B}_t^c.$$

We want to prove $\mathbb{P}(B_t) \leq e^{-t}$. Write $\mathbb{P}(Q = q) = Ce^{\frac{\varepsilon}{2\Delta} f(q)}$. Let q^* be an output with score f^*. Then

$$\mathbb{P}(B_t) = \sum_{q \in B_t} \mathbb{P}(Q = q) \leq \frac{\sum_{q \in B_t} \mathbb{P}(Q = q)}{\mathbb{P}(Q = q^*)} = \frac{\sum_{q \in B_t} e^{\frac{\varepsilon}{2\Delta} f(q)}}{e^{\frac{\varepsilon}{2\Delta} f^*}}.$$

Since the bad q satisfy $f(q) \leq f^* - \psi_t$, the numerator is bounded by

$$|B_t| \exp\left(\frac{\varepsilon}{2\Delta} f^* - (\ln(d) + t)\right).$$

Hence, since $|B_t| \leq d$,

$$\mathbb{P}(B_t) \leq |B_t| e^{-\ln(d)-t} \leq e^{-t}.$$

Exponential mechanism

Example 29 (Running an election)

- Set of objects: election candidates - $\mathcal{Q} = \{1, 2, 3\}$.
- Sensitive dataset: votes.
- Score function: number of votes for each candidate, e.g. $f(\mathbf{x}, 1) = 15$, $f(\mathbf{x}, 2) = 20$, $f(\mathbf{x}, 3) = 25$.
- Non-private: pick the candidate with the highest score (3).
- Private release: Choose e.g. $\varepsilon = 1$, Then, the first candidate has the chance of winning equal to

$$\frac{e^{\frac{1}{2}*15}}{e^{\frac{1}{2}*15} + e^{\frac{1}{2}*20} + e^{\frac{1}{2}*25}}.$$

Report noisy-max

The set-up is the same as for the exponential mechanism. Now, the release is

$$\operatorname{argmax}_{q_i \in \mathcal{Q}} \{f(\mathbf{x}, q_i) + Z_i\},$$

where Z_i are i.i.d. exponential with the rate $\frac{\varepsilon}{2\Delta}$.

Report noisy-max

The set-up is the same as for the exponential mechanism. Now, the release is

$$\operatorname{argmax}_{q_i \in \mathcal{Q}} \{f(\mathbf{x}, q_i) + Z_i\},$$

where Z_i are i.i.d. exponential with the rate $\frac{\varepsilon}{2\Delta}$.

Example 30

Assume the original scores are $(15, 20, 25)$ giving $q^* = 3$. After adding noise, we may have $(21, 16, 20)$ giving $q^* = 1$.

The noisy-max algorithm is ε-DP and achieves the same bounds as the exponential algorithm.

DP in ML - Gradient Descent

Let $f : \mathbb{R}^p \to \mathbb{R}$ be convex and differentiable. The goal is to find

$$\beta_* = \operatorname{argmin} f(\beta).$$

A necessary and sufficient condition for optimality of β_* is provided by the zero-gradient condition $\nabla f(\beta_*) = 0$. Gradient descent is an iterative algorithm for solving this fixed point equation

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \nabla f(\beta^{(h)}) \,, \quad h = 0, 1, 2, \ldots, N-1 \,, \tag{5}$$

where $\gamma_h > 0$ is a stepsize parameter. In Machine Learning this parameter is called *learning rate*. Above, N is the number of steps in the algorithm.

DP in ML - Gradient Descent

Let $f : \mathbb{R}^p \to \mathbb{R}$ be convex and differentiable. The goal is to find

$$\beta_* = \operatorname{argmin} f(\beta).$$

A necessary and sufficient condition for optimality of β_* is provided by the zero-gradient condition $\nabla f(\beta_*) = 0$. Gradient descent is an iterative algorithm for solving this fixed point equation

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \nabla f(\beta^{(h)}) , \quad h = 0, 1, 2, \ldots, N-1 , \tag{5}$$

where $\gamma_h > 0$ is a stepsize parameter. In Machine Learning this parameter is called *learning rate*. Above, N is the number of steps in the algorithm. This update has a natural geometric interpretation: by computing the gradient, we determine the direction of steepest descent $-\nabla f(\beta^{(h)})$, and then walk in this direction for a certain amount determined by the stepsize γ_h.

Gradient Descent

Example 31

Let $f(x) = 0.5x^2$. Choose $\beta^{(0)} = 1$. Then the update rule becomes

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \beta^{(h)} .$$

Gradient Descent

Example 31

Let $f(x) = 0.5x^2$. Choose $\beta^{(0)} = 1$. Then the update rule becomes

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \beta^{(h)} .$$

Example 32

Let $f(\beta) = \frac{1}{n}\sum_{i=1}^{n} L_i(\beta)$, were L_i is a loss-function that depends on the data. For example, in the one-dimensional OLS we have $L_i(\beta) = (Y_i - \beta X_i)^2$. Then the gradient descent step is

$$\beta^{(h+1)} = \beta^{(h)} - \frac{\gamma_h}{n} \sum_{i=1}^{n} \nabla L_i(\beta^{(h)}) ,$$

with

$$\nabla L_i(\beta) = -X_i(Y_i - \beta X_i).$$

Gradient Descent - convergence

The basic rate of convergence of the gradient descent is as follows:

$$|f(\bar{\beta}^{(N)}) - f^*| \leq \frac{\|\beta^{(0)} - \beta_*\|_2^2}{\sum_{j=0}^{N-1} \gamma_j} \tag{6}$$

where

$$\bar{\beta}^{(N)} = \frac{\sum_{j=0}^{N-1} \gamma_j \beta^{(j)}}{\sum_{j=0}^{N-1} \gamma_j} \, .$$

Gradient Descent - convergence

The basic rate of convergence of the gradient descent is as follows:

$$|f(\bar{\beta}^{(N)}) - f^*| \leq \frac{\|\beta^{(0)} - \beta_*\|_2^2}{\sum_{j=0}^{N-1} \gamma_j} \tag{6}$$

where

$$\bar{\beta}^{(N)} = \frac{\sum_{j=0}^{N-1} \gamma_j \beta^{(j)}}{\sum_{j=0}^{N-1} \gamma_j} \ .$$

If $\gamma_j = \gamma$, then the bound is of the order $1/N$. That is, to get a precision of order 0.001, we need $N = 1000$ iteration steps.

Gradient Descent - convergence

The basic rate of convergence of the gradient descent is as follows:

$$|f(\bar{\beta}^{(N)}) - f^*| \leq \frac{\|\beta^{(0)} - \beta_*\|_2^2}{\sum_{j=0}^{N-1} \gamma_j} \tag{6}$$

where

$$\bar{\beta}^{(N)} = \frac{\sum_{j=0}^{N-1} \gamma_j \beta^{(j)}}{\sum_{j=0}^{N-1} \gamma_j} \, .$$

If $\gamma_j = \gamma$, then the bound is of the order $1/N$. That is, to get a precision of order 0.001, we need $N = 1000$ iteration steps. The above bound is valid under some assumptions on f: convexity, differentiability and some conditions on the second order derivatives. We will not discuss the precise conditions.

Stochastic optimization problem

Let $L : \mathbb{R}^p \to \mathbb{R}$ be a measurable function. Let x_i, $i = 1, \ldots, n$ be a sample from X, where X is a random vector in $\mathbb{R}^p$ with a distribution $F = F_X$. Set $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times p}$ and $\mathbf{X} = (X_1, \ldots, X_n) \in \mathbb{R}^{n \times p}$, where X_i have the same distribution as X. That is, in the thesis terminology, $\mathbf{x}$ is a database, and we will treat the database as fixed.

Stochastic optimization problem

Let $L : \mathbb{R}^p \to \mathbb{R}$ be a measurable function. Let x_i, $i = 1, \ldots, n$ be a sample from X, where X is a random vector in $\mathbb{R}^p$ with a distribution $F = F_X$. Set $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times p}$ and $\mathbf{X} = (X_1, \ldots, X_n) \in \mathbb{R}^{n \times p}$, where X_i have the same distribution as X. That is, in the thesis terminology, $\mathbf{x}$ is a database, and we will treat the database as fixed. Functions L_i, $i = 1, \ldots, n$, will depend on data. Formally, we will let $L_i(\cdot) = L(\cdot; x_i)$. For $\beta \in \mathbb{R}^p$ we define

$$f(\beta; \boldsymbol{x}) := \frac{1}{n} \sum_{i=1}^{n} L_i(\beta) = \frac{1}{n} \sum_{i=1}^{n} L(\beta; x_i) \, .$$

We will usually drop the dependence on the data writing

$$f(\beta) = \frac{1}{n} \sum_{i=1}^{n} L_i(\beta) \, .$$

Stochastic optimization problem

Then, $f(\beta)$ is an empirical estimator of

$$\tilde{f}(\beta) := \mathbb{E}[L(\beta; X)] \,. \tag{7}$$

Above, $\mathbb{E}$ is the expectation with respect to the distribution of X (recall that X_i come from the same distribution as X).
Define further

$$\widehat{\beta}_* := \operatorname{argmin}_{\beta \in \mathbb{R}^p} f(\beta)$$

and

$$\widehat{f}^* = f(\widehat{\beta}_*) \,.$$

Note that

- $\widehat{\beta}_*$ depends on data $\boldsymbol{x} = (x_1, \ldots, x_n)$.
- $\widehat{f}^*$ depends on data $\boldsymbol{x} = (x_1, \ldots, x_n)$.

Stochastic optimization problem

In some elementary situations, the minimization problem can be solved explicitly by taking the gradient and solving $\nabla f(\beta) = 0$.

Example 33 (Sample mean)

Assume that X_i are random variables and have the same distribution as X. Here, the observations are $x_i \in \mathbb{R}$. Let

$$\tilde{f}(\beta) = \mathbb{E}[(X - \beta)^2] \, .$$

Then

$$L_i(\beta) = (x_i - \beta)^2, \quad f(\beta) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \beta)^2 \, .$$

Solving $\nabla f(\beta) = 0$ yields

$$\widehat{\beta}_* = \frac{1}{n} \sum_{i=1}^{n} x_i = \bar{\boldsymbol{x}} \, .$$

Stochastic optimization problem

Example 34 (Linear regression and RIDGE)

Consider a linear regression problem $U_i = \beta_0 + \beta_1 V_i + \varepsilon_i$, where $X_i := (U_i, V_i)$ have the same distribution as $X = (U, V)$. Here, the observations are $x_i = (u_i, v_i) \in \mathbb{R}^2$. Let $\beta = (\beta_0, \beta_1)$ and

$$\tilde{f}(\beta) = \frac{1}{2}\mathbb{E}[(U - \beta_0 - \beta_1 V)^2] + \frac{1}{2}\lambda\beta_1^2 .$$

Then

$$L_i(\beta) = \frac{1}{2}(u_i - \beta_0 - \beta_1 v_i)^2 + \frac{\lambda}{2}\beta_1^2,$$

and

$$f(\beta) = \frac{1}{2n}\sum_{i=1}^{n}(u_i - \beta_0 - \beta_1 v_i)^2 + \frac{\lambda}{2}\beta_1^2 .$$

Stochastic optimization problem

Hence,

$$\nabla f(\beta) = \left(-\frac{1}{n}\sum_{i=1}^{n}(u_i - \beta_0 - \beta_1 v_i), -\frac{1}{n}\sum_{i=1}^{n} v_i(u_i - \beta_0 - \beta_1 v_i) + \lambda\beta_1 \right)^T .$$

Solving $\nabla f(\beta) = 0$ yields $\widehat{\beta}_* = (\widehat{\beta}_{0,*}, \widehat{\beta}_{1,*})$ as the ridge estimator:

$$\widehat{\beta}_{1,*} = \frac{\frac{1}{n}\sum_{i=1}^{n}(v_i - \bar{\mathbf{v}})(u_i - \bar{\mathbf{u}})}{\frac{1}{n}\sum_{i=1}^{n}(v_i - \bar{\mathbf{v}})^2 + \lambda} , \quad \widehat{\beta}_{0,*} = \bar{\mathbf{u}} - \widehat{\beta}_{1,*}\bar{\mathbf{v}} . \tag{8}$$

Stochastic Gradient Descent

One-pass (online) **Stochastic Gradient Descent** is performed as follows:

- Divide the dataset of size n into disjoint batches of size b.
- For $h = 1, \ldots, N = n/b$ update

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \widehat{\nabla} f_{h+1}(\beta^{(h)}) \,,$$

where the gradient is only evaluated based on the data from the batch. That is, instead of

$$\nabla f(\beta^{(h)}) = \frac{1}{n} \sum_{i=1}^{n} \nabla L(\beta^{(h)}, x_i),$$

we calculate

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \frac{1}{b} \sum_{i \in I_{h+1}} \nabla L(\beta^{(h)}, x_i),$$

where

$$I_{h+1} = \{hb + 1, \ldots, (h+1)b\}$$

is the set of indices of size b.

Stochastic Gradient Descent

One-pass (online) **Stochastic Gradient Descent** is performed as follows:

- Divide the dataset of size n into disjoint batches of size b.
- For $h = 1, \ldots, N = n/b$ update

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \widehat{\nabla} f_{h+1}(\beta^{(h)}) \,,$$

where the gradient is only evaluated based on the data from the batch. That is, instead of

$$\nabla f(\beta^{(h)}) = \frac{1}{n} \sum_{i=1}^{n} \nabla L(\beta^{(h)}, x_i),$$

we calculate

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \frac{1}{b} \sum_{i \in I_{h+1}} \nabla L(\beta^{(h)}, x_i),$$

where

$$I_{h+1} = \{hb + 1, \ldots, (h+1)b\}$$

is the set of indices of size b.

Important: each data point X_i is used only once in the algorithm.

Stochastic Gradient Descent

Multi-pass (offline) **Stochastic Gradient Descent** is performed as follows:

- Assume we have a dataset of size n.
- For $h = 1, \ldots, N$, choose at random b data points.
- For $h = 1, \ldots, N$ update

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \widehat{\nabla} f_{h+1}(\beta^{(h)}) \,,$$

where the gradient is only evaluated based on the data from the batch:

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \frac{1}{b} \sum_{i \in \Omega_{h+1}} \nabla L(\beta^{(h)}, x_i) = \frac{1}{b} \sum_{i \in \Omega_{h+1}} \nabla L_i(\beta^{(h)}),$$

where Ω_{h+1} is the set of b indices chosen at random from $\{1, \ldots, n\}$.

Stochastic Gradient Descent

Multi-pass (offline) **Stochastic Gradient Descent** is performed as follows:

- Assume we have a dataset of size n.
- For $h = 1, \ldots, N$, choose at random b data points.
- For $h = 1, \ldots, N$ update

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \widehat{\nabla} f_{h+1}(\beta^{(h)}) \,,$$

where the gradient is only evaluated based on the data from the batch:

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \frac{1}{b} \sum_{i \in \Omega_{h+1}} \nabla L(\beta^{(h)}, x_i) = \frac{1}{b} \sum_{i \in \Omega_{h+1}} \nabla L_i(\beta^{(h)}),$$

where Ω_{h+1} is the set of b indices chosen at random from $\{1, \ldots, n\}$.

Important: each data point X_i is used only once in the given iteration, but may be used multiple times in the algorithm.

Stochastic Gradient Descent

In case of stochastic gradient descent we need to choose learning rates very carefully.

Theorem 35

Consider the offline SGD algorithm with the batch size $b = 1$. Under the appropriate assumptions

$$\mathbb{E}_{\boldsymbol{x}}[f(\bar{\beta}^{(N)}) - f^*] \leq \frac{\|\beta^{(0)} - \widehat{\beta}_*\|_2^2}{\sum_{j=0}^{N-1} \gamma_j} + C \frac{\sum_{j=0}^{N-1} \gamma_j^2}{\sum_{j=0}^{N-1} \gamma_j} \tag{9}$$

where

$$\bar{\beta}^{(N)} = \frac{\sum_{j=0}^{N-1} \gamma_j \beta^{(j)}}{\sum_{j=0}^{N-1} \gamma_j} .$$

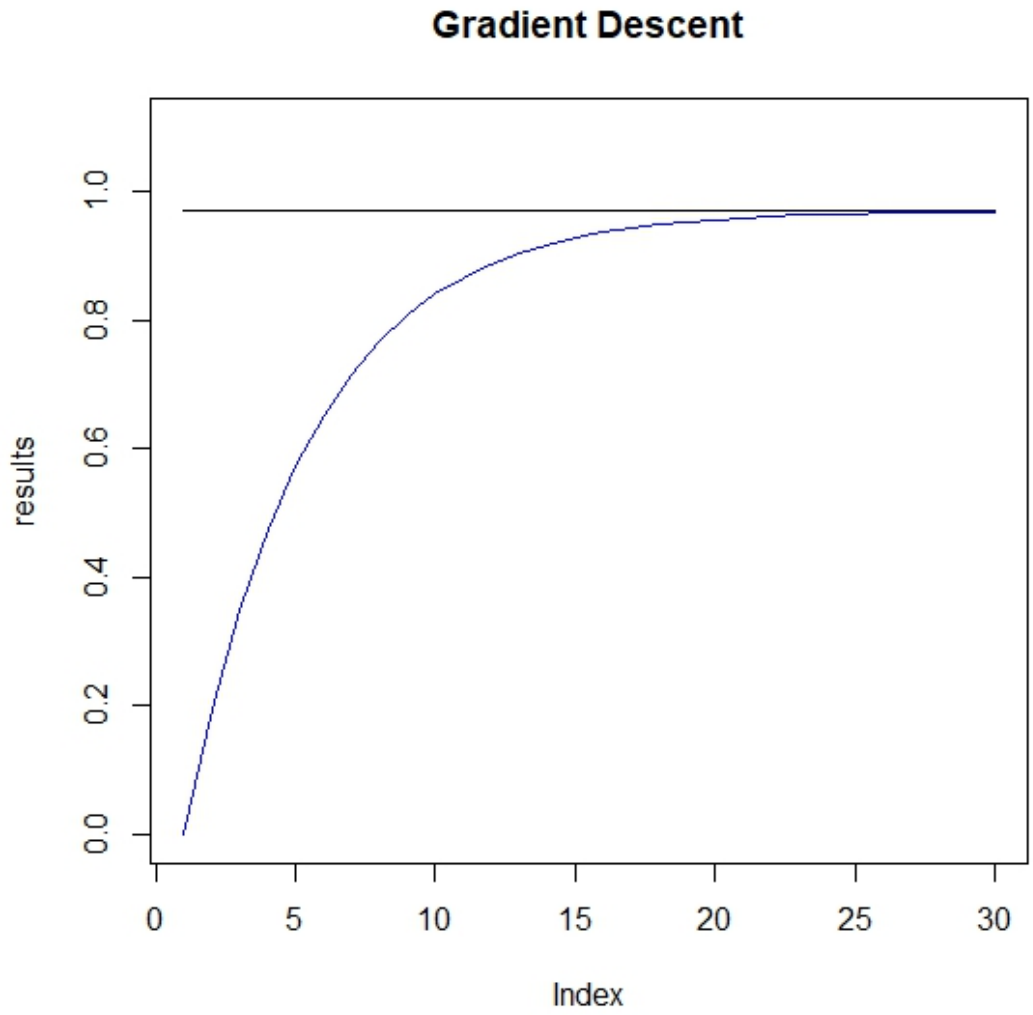
Stochastic Gradient Descent - convergence

- Above, the expectation is conditional on the data. As such, the constant depends on the data.
- If $\gamma_h = \gamma$ is constant, then the first term on the right hand side of (9) is of order $1/N$, as in the standard case of gradient descent.
- If $\gamma_h = \gamma$ is constant, then the second term in (9) *does not vanish* as $N \to \infty$. As such, **the stochastic gradient descent with constant learning rates does not converge.** At the last iteration $\beta^{(N)}$ equals $\widehat{\beta}_* + \text{noise}$. We need rates of the form $\gamma_h = h^{-\rho}$, $\rho \in (0, 1)$.

Illustrations - Example 33

```
n=1000; x=rexp(n);
beta_star=mean(x); # True minimizer
# Gradient Descent
beta0=0; gamma=0.1;
N=30; Index=c(1:N);
results=c(beta0)
for(i in 1:(N-1))
 {
 beta.old=results[i];
 beta.new=beta.old+gamma*2*mean(x-beta.old);
 results[i+1]=beta.new
 }
plot(Index,results,type="l",col="blue",ylim=c(0,1.1))
lines(Index,rep(beta_star,N))
```

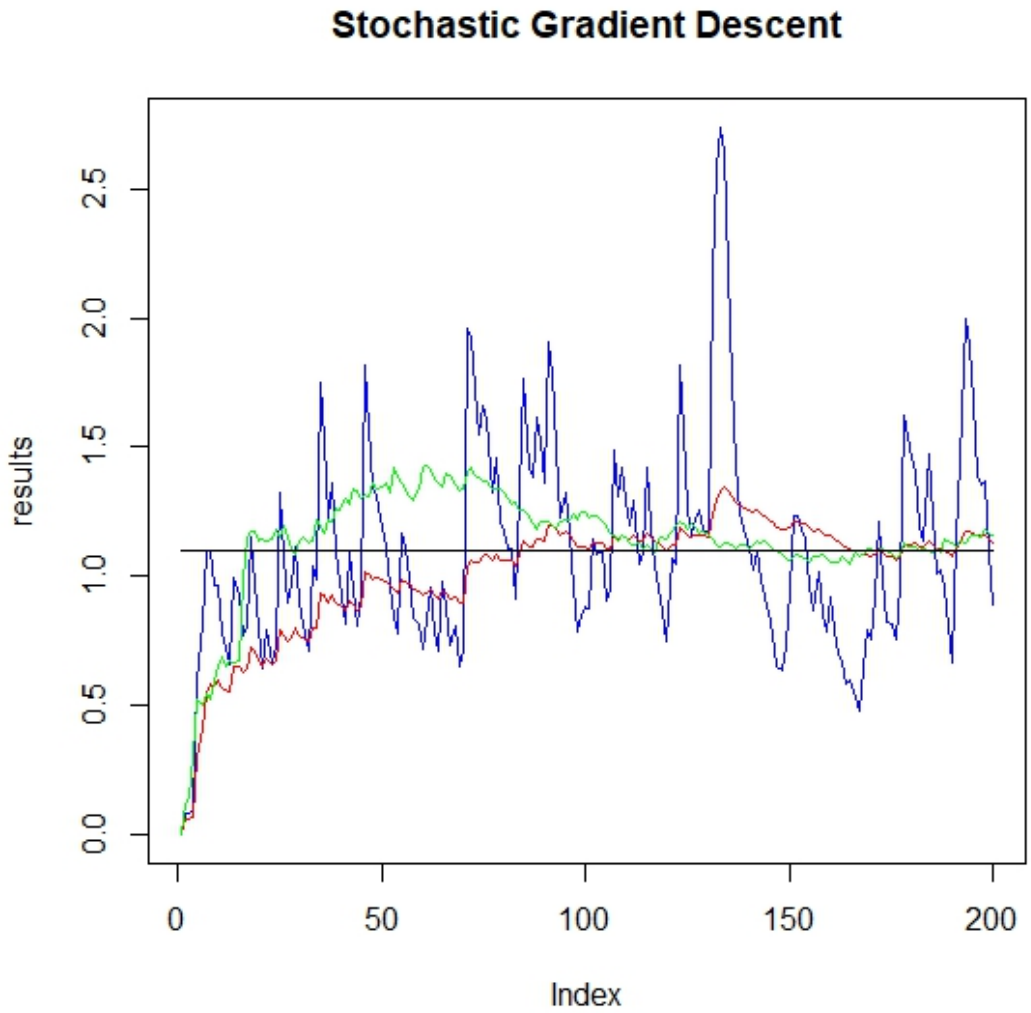
Illustrations - Example 33



Illustrations - Example 33

```
n=1000; x=rexp(n); beta_star=mean(x);
# Stochastic Gradient Descent
beta0=0; gamma=0.1; N=200; Index=c(1:N);
res=c(beta0); res.h=c(beta0); res.r=c(beta0)
for(i in 1:(N-1))
 {J=sample(1:n,1);
 b.old=res[i];b.old.h=res.h[i];b.old.r=res.r[i];
 b.new=b.old+gamma*2*(x[i]-b.old);
 b.new.h=b.old.h+gamma*2*(x[i]-b.old.h)/sqrt(1+i);
 b.new.r=b.old.r+gamma*2*(x[J]-b.old.r)/sqrt(1+i);
 res[i+1]=b.new;res.h[i+1]=b.new.h;res.r[i+1]=b.new.r}
M=max(res,res.h)
plot(Index,res,type="l",col="blue",ylim=c(0,M),main="SGD")
lines(Index,res.h,type="l",col="red");
lines(Index,res.r,type="l",col="green")
lines(Index,rep(beta_star,N))
```

Illustrations - Example 33



Illustrations - Example 33

- GD converges after 30 iterations;
- SGD with the constant learning rate does not converge;
- Both SGDs with the decreasing learning rates converge;
- For GD, we need to compute $N * n = 30,000$ gradients;
- For SGD, we need to compute $N * 1 = 200$ gradients;

Privacy of SGD

- Recall the standard DP: Release $f(\mathbf{x}) + Z$, where e.g. $f(\mathbf{x}) = \bar{\mathbf{x}}$. Thus, we release the sample mean + noise.
- In the SGD context, we release $\beta^{(N)}$. Recall: if we use constant learning rates, then at the last iteration $\beta^{(N)}$ equals $\widehat{\beta}_* + \text{noise}$. Hence, the algorithm itself provides privacy protection. But, at the same time, the algorithm does not converge.
- **The idea:** make an algorithm in such the way that we have privacy, but the algorithm converges.

Differentially Private SGD

Consider offline SGD with batch size $b = 1$. Write

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \nabla L_{\Omega_{h+1}}(\beta^{(h)}) = \nabla L(\beta^{(h)}, x_{\Omega_{h+1}}),$$

where now Ω_{h+1}, $h \geq 0$, is a random number selected from $\{1, \ldots, n\}$.

Differentially Private SGD

Consider offline SGD with batch size $b = 1$. Write

$$\widehat{\nabla} f_{h+1}(\beta^{(h)}) = \nabla L_{\Omega_{h+1}}(\beta^{(h)}) = \nabla L(\beta^{(h)}, x_{\Omega_{h+1}}),$$

where now Ω_{h+1}, $h \geq 0$, is a random number selected from $\{1, \ldots, n\}$. We consider a version of the output perturbation mechanism. Instead of

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \nabla L_{\Omega_{h+1}}(\beta^{(h)})$$

we consider

$$\beta^{(h+1)} = \beta^{(h)} - \gamma_h \left(\nabla L_{\Omega_{h+1}}(\beta^{(h)}) + \theta_h Z_h \right) , \tag{10}$$

where Z_h is a sequence of independent, identically distributed p-dimensional random vectors with the standard Laplace distribution and θ_h is a nonnegative sequence to be introduced below.

Differentially Private SGD

Let $\mathbf{x}$ and $\mathbf{y}$ be two neighbouring databases. The functions L_i and hence the sequence $\beta^{(h)}$ depend on the underlying database. We will need to make this dependence explicit. Furthermore, the sequence $\beta^{(h)}$ is random. Hence, we will write

$$\beta^{(h+1)}(\mathbf{x}) = \beta^{(h)}(\mathbf{x}) - \gamma_h \left(\nabla L_{\Omega_{h+1}}(\beta^{(h)}(\mathbf{x}); \mathbf{x}) + \theta_h Z_t \right) ,$$

and

$$\beta^{(h+1)}(\mathbf{y}) = \beta^{(h)}(\mathbf{y}) - \gamma_h \left(\nabla L_{\Omega_{h+1}}(\beta^{(h)}(\mathbf{y}); \mathbf{y}) + \theta_h Z_t \right) ,$$

Differentially Private SGD

Let $\mathbf{x}$ and $\mathbf{y}$ be two neighbouring databases. The functions L_i and hence the sequence $\beta^{(h)}$ depend on the underlying database. We will need to make this dependence explicit. Furthermore, the sequence $\beta^{(h)}$ is random. Hence, we will write

$$\beta^{(h+1)}(\mathbf{x}) = \beta^{(h)}(\mathbf{x}) - \gamma_h \left(\nabla L_{\Omega_{h+1}}(\beta^{(h)}(\mathbf{x}); \mathbf{x}) + \theta_h Z_t \right) ,$$

and

$$\beta^{(h+1)}(\mathbf{y}) = \beta^{(h)}(\mathbf{y}) - \gamma_h \left(\nabla L_{\Omega_{h+1}}(\beta^{(h)}(\mathbf{y}); \mathbf{y}) + \theta_h Z_t \right) ,$$

This way we define the output-perturbation mechanism sequences

$$\mathcal{A}_h(\mathbf{x}) = \beta^{(h)}(\mathbf{x}) , \quad h = 1, 2, 3, \ldots ,$$

and

$$\mathcal{A}_h(\mathbf{y}) = \beta^{(h)}(\mathbf{y}) , \quad h = 1, 2, 3, \ldots .$$

Differentially Private SGD

The following theorem is the main result on DP-property of the Stochastic Gradient Descent.

Theorem 36

Let $\Delta(\nabla L)$ be the global sensitivity of the gradient ∇L. Assume that:

- *Z_h, $h = 1, 2 \ldots$, is the sequence of independent, identically distributed random vectors with the standard Laplace distribution;*
- *The constants θ_h are given by*

$$\theta_1 = \Delta(\nabla L)/\varepsilon\ ,\ \ \theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}} + 1\right\}\Delta(\nabla L)/\varepsilon\ ,\ \ h = 2, 3, \ldots\ ,$$

where a_h is an arbitrary sequence of nonnegative numbers.

Then, the sequence $\mathcal{A}_h$, $h = 1, 2, \ldots$ is εa_h-DP.

Privacy of the Differentially Private SGD

Proof: Let $\beta^{(0)}$ be fixed. We run the SGD algorithm for two neighbouring databases $\mathbf{x}$ and $\mathbf{y}$. We must keep in mind that we have two sources of randomness, the random index Ω_{h+1} and the private noise Z_h.

Privacy of the Differentially Private SGD

Proof: Let $\beta^{(0)}$ be fixed. We run the SGD algorithm for two neighbouring databases $\mathbf{x}$ and $\mathbf{y}$. We must keep in mind that we have two sources of randomness, the random index Ω_{h+1} and the private noise Z_h.

- Thus, we will consider conditioning $\mathbb{E}_{\Omega_{h+1}}[\cdot] = \mathbb{E}[\cdot \mid \Omega_{h+1}]$ on Ω_{h+1}.
- For the subsequent steps we need to condition not only on Ω_{h+1}, but also on the current value $\beta^{(h)}$. We then use the notation $\mathbb{E}_h[\cdot] = \mathbb{E}[\cdot \mid \Omega_{h+1}, \beta^{(h)}]$.

Privacy of the Differentially Private SGD

Proof: Let $\beta^{(0)}$ be fixed. We run the SGD algorithm for two neighbouring databases $\mathbf{x}$ and $\mathbf{y}$. We must keep in mind that we have two sources of randomness, the random index Ω_{h+1} and the private noise Z_h.

- Thus, we will consider conditioning $\mathbb{E}_{\Omega_{h+1}}[\cdot] = \mathbb{E}[\cdot \mid \Omega_{h+1}]$ on Ω_{h+1}.
- For the subsequent steps we need to condition not only on Ω_{h+1}, but also on the current value $\beta^{(h)}$. We then use the notation $\mathbb{E}_h[\cdot] = \mathbb{E}[\cdot \mid \Omega_{h+1}, \beta^{(h)}]$.

In what follows, B is a Borel set in $\mathbb{R}^p$. Then, we write $cB + a$, $a \in \mathbb{R}^p$, $c \in \mathbb{R}$, for $cB + a = \{cb + a : b \in B\}$.

Differentially Private SGD

Step $h = 1$ (we will do one step only!) At the first iteration $h = 1$ we have

$$\begin{aligned}
\mathbb{P}(\mathcal{A}_1(\mathbf{x}) \in B) &= \mathbb{P}\left(\beta^{(0)} - \gamma_1 \left(\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x}) + \theta_1 Z_1\right) \in B\right) \\
&= \mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(\theta_1 Z_1 \in \frac{\beta^{(0)} - B}{\gamma_0} - \nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x})\right)\right] \\
&= \mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(Z_1 \in \frac{\beta^{(0)} - B}{\theta_1 \gamma_0} - \frac{1}{\theta_1}\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x})\right)\right] .
\end{aligned}$$

Differentially Private SGD

Step $h = 1$ (we will do one step only!) At the first iteration $h = 1$ we have

$$\begin{aligned}\mathbb{P}(\mathcal{A}_1(\mathbf{x}) \in B) &= \mathbb{P}\left(\beta^{(0)} - \gamma_1\left(\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x}) + \theta_1 Z_1\right) \in B\right) \\ &= \mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(\theta_1 Z_1 \in \frac{\beta^{(0)} - B}{\gamma_0} - \nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x})\right)\right] \\ &= \mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(Z_1 \in \frac{\beta^{(0)} - B}{\theta_1\gamma_0} - \frac{1}{\theta_1}\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x})\right)\right] .\end{aligned}$$

Goal: Replace $\mathbf{x}$ with $\mathbf{y}$.
Let

$$B_1 = \frac{\beta^{(0)} - B}{\theta_1\gamma_0} - \frac{1}{\theta_1}\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x})$$

and set

$$q = \frac{1}{\theta_1}\left\{\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{x}) - \nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{y})\right\} .$$

Differentially Private SGD

We note that with the choice $\theta_1 = \Delta(\nabla L)/\varepsilon$ we have

$$|q| = \frac{1}{\theta_1}\left\{\nabla L(\beta^{(0)}, x_{\Omega_1}) - \nabla L(\beta^{(0)}, y_{\Omega_1})\right\} \leq \frac{1}{\theta_1}\Delta(\nabla L) \leq \varepsilon\ .$$

Thus, using the property of the Laplace distribution (as in the classical case),

$$\begin{aligned}
&\mathbb{P}(\mathcal{A}_1(\mathbf{x}) \in B) = \mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(Z_1 \in B_1\right)\right] \\
&\leq e^{\varepsilon}\mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(Z_1 \in B_1 + q\right)\right] \\
&= e^{\varepsilon}\mathbb{E}_{\Omega_1}\left[\mathbb{P}\left(Z_1 \in \frac{\beta^{(0)} - B}{\theta_1\gamma_0} - \frac{1}{\theta_1}\nabla L_{\Omega_1}(\beta^{(0)}; \mathbf{y})\right)\right] \\
&= e^{\varepsilon}\mathbb{P}(\mathcal{A}_1(\mathbf{y}) \in B)\ .
\end{aligned}$$

In conclusion, the first iteration in the SGD algorithm is ε-DP.

Data utility of the Differentially Private SGD

Theorem 37

Consider the offline DP-SGD algorithm with the batch size $b = 1$. Under the appropriate assumptions

$$\mathbb{E}_{\boldsymbol{x}}[f(\bar{\beta}^{(N)}) - f^*] \leq \underbrace{\frac{\|\beta^{(0)} - \widehat{\beta}_*\|_2^2}{\sum_{j=0}^{N-1} \gamma_j}}_{=:A_1} + \underbrace{C\frac{\sum_{j=0}^{N-1} \gamma_j^2}{\sum_{j=0}^{N-1} \gamma_j}}_{=:A_2} + \underbrace{C\frac{\sum_{j=0}^{N-1} \gamma_j^2 \theta_j^2}{\sum_{j=0}^{N-1} \gamma_j}}_{=:A_3} \tag{11}$$

where

$$\bar{\beta}^{(N)} = \frac{\sum_{j=0}^{N-1} \gamma_j \beta^{(j)}}{\sum_{j=0}^{N-1} \gamma_j}.$$

Data utility of the Differentially Private SGD

- The third term in (11) is the contribution due to privacy.
- Can we make it vanishing as $N \to \infty$?

Data utility of the Differentially Private SGD I

Consider a decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$ (a typical choice in the SGD literature). Assume again that $a_h = 1$, that is, we want to preserve the same ε-DP at each iteration.

Data utility of the Differentially Private SGD I

Consider a decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$ (a typical choice in the SGD literature). Assume again that $a_h = 1$, that is, we want to preserve the same ε-DP at each iteration. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx \left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h\frac{\Delta(\nabla L)}{\varepsilon},$$

Data utility of the Differentially Private SGD I

Consider a decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$ (a typical choice in the SGD literature). Assume again that $a_h = 1$, that is, we want to preserve the same ε-DP at each iteration. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx \left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h\frac{\Delta(\nabla L)}{\varepsilon},$$

$$A_1 = \frac{1}{\sum_{j=0}^{N-1}\gamma_j} = \frac{1}{\sum_{j=0}^{N-1}(j+1)^{-\rho}} \approx N^{\rho-1} \to 0 \quad \text{as } N \to \infty,$$

$$A_2 = \frac{\sum_{j=0}^{N-1}\gamma_j^2}{\sum_{j=0}^{N-1}\gamma_j} = \frac{\sum_{j=0}^{N-1}(j+1)^{-2\rho}}{\sum_{j=0}^{N-1}(j+1)^{-\rho}} \approx N^{-\rho} \to 0 \quad \text{as } N \to \infty,$$

$$A_3 = \frac{\sum_{j=0}^{N-1} j^{-2\rho} j^2}{\sum_{j=0}^{N-1} j^{-\rho}} \approx \frac{N^{3-2\rho}}{N^{1-\rho}} = N^{2-\rho} \to \infty \quad \text{as } N \to \infty.$$

Data utility of the Differentially Private SGD I

Consider a decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$ (a typical choice in the SGD literature). Assume again that $a_h = 1$, that is, we want to preserve the same ε-DP at each iteration. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx \left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}} + 1\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h\frac{\Delta(\nabla L)}{\varepsilon},$$

$$A_1 = \frac{1}{\sum_{j=0}^{N-1}\gamma_j} = \frac{1}{\sum_{j=0}^{N-1}(j+1)^{-\rho}} \approx N^{\rho-1} \to 0 \quad \text{as } N \to \infty,$$

$$A_2 = \frac{\sum_{j=0}^{N-1}\gamma_j^2}{\sum_{j=0}^{N-1}\gamma_j} = \frac{\sum_{j=0}^{N-1}(j+1)^{-2\rho}}{\sum_{j=0}^{N-1}(j+1)^{-\rho}} \approx N^{-\rho} \to 0 \quad \text{as } N \to \infty,$$

$$A_3 = \frac{\sum_{j=0}^{N-1} j^{-2\rho} j^2}{\sum_{j=0}^{N-1} j^{-\rho}} \approx \frac{N^{3-2\rho}}{N^{1-\rho}} = N^{2-\rho} \to \infty \quad \text{as } N \to \infty.$$

Thus, the DP-SGD with the decreasing learning rate and the constant level of privacy **does not converge**.

Data utility of the Differentially Private SGD II

Consider again the decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$. Now, we assume that $a_h = h^{\kappa}$, $\kappa > 0$, that is, the privacy decreases at each iteration. We want to choose κ as small as possible.

Data utility of the Differentially Private SGD II

Consider again the decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$. Now, we assume that $a_h = h^{\kappa}$, $\kappa > 0$, that is, the privacy decreases at each iteration. We want to choose κ as small as possible. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{-\kappa}\left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{1-\kappa}\frac{\Delta(\nabla L)}{\varepsilon}\ .$$

Data utility of the Differentially Private SGD II

Consider again the decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$. Now, we assume that $a_h = h^\kappa$, $\kappa > 0$, that is, the privacy decreases at each iteration. We want to choose κ as small as possible. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{-\kappa}\left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{1-\kappa}\frac{\Delta(\nabla L)}{\varepsilon}\ .$$

Next, A_1 and A_2 are the same as in the preceding example. But,

$$A_3 := \frac{\sum_{j=0}^{N-1} j^{-2\rho} j^{2-2\kappa}}{\sum_{j=0}^{N-1} j^{-\rho}} \approx \frac{N^{3-2\rho-2\kappa}}{N^{1-\rho}} = N^{2-\rho-2\kappa}\ .$$

The last term vanishes whenever $2\kappa + \rho > 2$.
In application, one often chooses $\rho = 1/2$. Then the optimal choice is $\kappa > 3/4$.

Data utility of the Differentially Private SGD II

Consider again the decreasing learning rate, $\gamma_h = (1+h)^{-\rho}$, $\rho \in (0,1)$. Now, we assume that $a_h = h^\kappa$, $\kappa > 0$, that is, the privacy decreases at each iteration. We want to choose κ as small as possible. Then

$$\theta_h = \frac{1}{a_h}\left\{\frac{\sum_{j=0}^{h-2}\gamma_j}{\gamma_{h-1}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{-\kappa}\left\{\frac{(h-1)^{-\rho+1}}{h^{-\rho}}\right\}\frac{\Delta(\nabla L)}{\varepsilon} \approx h^{1-\kappa}\frac{\Delta(\nabla L)}{\varepsilon}\ .$$

Next, A_1 and A_2 are the same as in the preceding example. But,

$$A_3 := \frac{\sum_{j=0}^{N-1} j^{-2\rho} j^{2-2\kappa}}{\sum_{j=0}^{N-1} j^{-\rho}} \approx \frac{N^{3-2\rho-2\kappa}}{N^{1-\rho}} = N^{2-\rho-2\kappa}\ .$$

The last term vanishes whenever $2\kappa + \rho > 2$.
In application, one often chooses $\rho = 1/2$. Then the optimal choice is $\kappa > 3/4$.
The algorithm is DP-private AND converges.

DP in practice

- Apple technical documentation.
- US census 2020: `https://www.census.gov/programs-surveys/decennial-census/decade/2020/planning-management/process/disclosure-avoidance/differential-privacy.html`
  `https://hdsr.mitpress.mit.edu/pub/3vj5j6i0/release/3`