sciendo Proceedings on Privacy Enhancing Technologies ; 2020 (2):175–208

Peeter Laud*, Alisa Pankova, and Martin Pettai

A Framework of Metrics for Differential Privacy from Local Sensitivity

Abstract: The meaning of differential privacy (DP) is tightly bound with the notion of distance on databases, typically defined as the number of changed rows. Considering the semantics of data, this metric may be not the most suitable one, particularly when a distance comes out as larger than the data owner desired (which would undermine privacy). In this paper, we give a mechanism to specify continuous metrics that depend on the locations and amounts of changes in a much more nuanced manner. Our metrics turn the set of databases into a *Banach space*. In order to construct DP information release mechanisms based on our metrics, we introduce *derivative sensitivity*, an analogue to local sensitivity for continuous functions. We use this notion in an analysis that determines the amount of noise to be added to the result of a database query in order to obtain a certain level of differential privacy, and demonstrate that derivative sensitivity allows us to employ powerful mechanisms from calculus to perform the analysis for a variety of queries. We have implemented the analyzer and evaluated its efficiency and precision.

Keywords: differential privacy, sensitivity

DOI 10.2478/popets-2020-0023
Received 2019-08-31; revised 2019-12-15; accepted 2019-12-16.

1 Introduction

Differential privacy [8] (DP) is one of the most prominent ways to quantitatively define privacy losses from releasing derived information about data collections, and to rigorously argue about the accumulation of these losses in information processing systems. Differentially private information release mechanisms invariably employ the addition of noise somewhere during the processing, hence reducing the utility of the result. For specific information processing tasks, or families of tasks, there exist carefully designed methods to achieve DP with only a little loss in utility [16]; for some methods, this loss asymptotically approaches zero as the size of the processed dataset grows [24]. But a general method for making an information release mechanism differentially private is to add noise of appropriate magnitude to the output of this mechanism. This magnitude depends on the *sensitivity* of the mechanism — the amount of change of its output when its input is changed by a unit amount. This paper is devoted to the study of computing (or safely approximating) the sensitivity of information release mechanisms given by their source code.

The definition of the ratio of changes of the outputs and inputs of the information release mechanism requires metrics on both of them. A common approach is requiring the outputs to be numeric. In fact, the output of the mechanism is a single real number, and the distance between outputs is their difference. This approach can be generalized to mechanisms that output histograms, i.e. tuples of numbers.

The definition of the metric on inputs is a richer question, and really reflects how the data owner wants to quantify its privacy. DP can be defined with respect to any metric on the set of possible inputs [5]. For database tables and databases, a common metric has been the number of different rows [23]. But the data owner may consider different changes in one row of some database table as different, particularly if it involves some numeric attributes. A small change in such attribute in a single row may be considered as "small", and the DP mechanism should strive to hide such change. A large change in the same attribute in a single row may be allowed to be more visible in the output. A change in geographic coordinates may be defined differently from a change in the length or the quantity of something. Finally, when several attributes change in a single row, then there are many reasonable ways of combining their changes (including summing them up, or taking their maximum) in defining how much the row has changed. A variety of definitions for the combination of changes is also possible if several rows change in a table, or several tables change in a database. We note that for inputs to the information release mechanism, the overestimation of their distance leads to the underestimation of the

***Corresponding Author: Peeter Laud:** Cybernetica AS, E-mail: peeter@cyber.ee
Alisa Pankova: Cybernetica AS, E-mail: alisa@cyber.ee
Martin Pettai: Cybernetica AS, E-mail: martin@cyber.ee

amount of noise added to achieve a certain level of DP. Hence it is crucial to handle a wide range of metrics, from which the data owner may select the one that he considers to best reflect his privacy expectations.

With new definitions of DP come the questions of achieving it. DP w.r.t. different metrics has been considered before [5], but only with *global* sensitivity of the query being used to scale the noise necessary for achieving it. In this paper, we generalize the notion of *local sensitivity* [24] to real-valued metrics and at the same time simplify it, tying it with the fundamental notions of functional analysis and expanding the kinds of input data to which it may be applied. We give a general framework for defining different metrics on databases, and computing the sensitivities of queries with respect to these input metrics. It has the following scope:

– Being based on local sensitivity, we require the actual content of the database in order to compute the distribution of added noise for achieving DP.
– Our metrics for rows, tables, and databases are combinations and compositions of ℓ_p-metrics with $1 \leq p \leq \infty$. Certain rows or columns may be left out from the computation of the metric, effectively declaring them as public. The columns that remain should be numeric (or encoded as numeric). The number of table rows defines dimensionality of the metric space, and is considered public.
– Computation of sensitivity of a function is based on its derivative, so the framework primarily covers continuous functions. Non-continuous constructions (like filtering) are approximated by continuous functions, which itself introduces some noise.
– As queries we support a significant subset of SQL with projections, filters, joins and also certain types of subqueries. The final output is numeric, i.e. an aggregation or a couple of aggregations. We do not support DISTINCT queries, and GROUP BY is limited to public and discrete attributes.

We start from presenting the metric-defining framework (Sec. 4.1). We follow it up with the definition of sensitivity that makes use of the continuity of the metrics; this sensitivity is applicable for mappings from the defined metric spaces to real numbers (Sec. 4.2). We show that this sensitivity can be used to measure the amount of noise to be added to the outputs of these maps in order to make them differentially private. We describe how to compute this sensitivity and its smooth upper bounds from the description of mappings. We show how these results can be applied to SQL queries (Sec. 5).

The multitude of different metrics presents a novel problem. The computations of derivative sensitivity (our name for the analogue of local sensitivity, defined in this paper) for various functions out of metric spaces can be done only for certain metrics, depending on the particular function. At the same time, the data owner has defined a particular metric for the databases, and wants the queries against his database to be differentially private with respect to this metric. In Sec. 5.3 we show how to approximate and upper-bound sensitivities of the same function with respect to different metrics, and thereby provide a solution to this problem.

We evaluate a concrete implementation of our theory in Sec. 7, demonstrating its benefits on particular examples. Our analyzer takes as inputs the description of the database and the metric on it, the query to be performed, as well as the actual contents of the database (as required by local sensitivity based approaches), and returns the value of smooth derivative sensitivity of this query at this database. The returned value can be used to scale the added noise in order to obtain a certain level of DP. The analyzer is integrated with the database management system in order to perform the computations specific to the contents of the database.

2 Related Work

Differential privacy was introduced by Dwork [8]. PINQ [23] is a worked-out example of using it for providing privacy-preserving replies to database queries. In an implementation, our analyzer of SQL queries would occupy the same place as the PINQ wrapper which analyses LINQ queries and maintains the privacy budget.

It has been recognized that any metric on the set of possible inputs gives us a definition of DP with respect to this metric [5, 13], but there has been no investigation of a systematic construction of such metrics that are also usable in constructing DP mechanisms. A particular application of DP w.r.t. a particular metric has been the privacy-preserving processing of location data [6]. The personalized differential privacy [12] can also be seen as an instance of using an arbitrary metric, albeit with a more complex set of distances. In Blowfish [17, 20], the metric rises from the distance on a graph where vertices are the possible database instances.

We use *norms* to state the privacy requirements on input data. Normed vector spaces have appeared in the DP literature in the context of K-normed mechanisms [3, 15], which extend the one-dimensional and generalize the many-dimensional Laplace mechanism. These mechanisms are rather different from our techniques and they do not explore the use of completeness of norms and differentiability of information release mechanism to find their sensitivity.

Nissim et al. [24] introduce *local sensitivity* and its smooth upper bounds, and use them to give differentially private approximations for certain statistical functions. The local sensitivity of a function is similar to its derivative. This has been noticed [19], but we are not aware of this similarity being thoroughly exploited, except perhaps for devising DP machine learning methods [28]. In this paper, this similarity will play a central role.

A couple of different static approaches for determining sensitivities of SQL queries or their upper bounds have been proposed. Palamidessi and Stronati [25] apply abstract interpretation to an SQL query, following its abstract syntax tree, combining the sensitivities of relational algebra operations (projection and filter have sensitivity at most 1, set operations have sensitivity at most 2, etc.) similarly to [11]. Additionally, they track the diameters of the domains of the values in the outcome of the query; the sensitivity cannot be larger than the diameter. Pierce et al. [14, 26] use linear and/or dependent types to derive bounds on the sensitivities of programs.

The computability of the precise sensitivity of the queries is studied by Arapinis et al. [2]. They identify a subclass of queries (*Conjunctive queries* with restricted `WHERE`-clauses) for which the sensitivity can be precisely determined. However, they also show that the problem is uncomputable in general. In addition, they show how functional dependencies and cardinality constraints may be used to upper-bound sensitivities of join-queries.

Cardinality constraints are also used by Johnson et al. [18] in their abstract interpretation based approach. For a database, they consider the maximum frequency of a value of an attribute in one of the tables, maximized over all databases at most at some distance to the original database, thereby building on the notion of (smooth) local sensitivity.

3 Preliminaries

3.1 Sensitivity and Differential Privacy

Let X be the set of possible databases. We assume that there is a metric $d_X(x, x')$ for $x, x' \in X$, quantifying the difference between two databases. For example, we could define $d_X(x, x') = n$ for two datatables whose tables differ in exactly n rows in total.

For a set Y, let $\mathcal{D}(Y)$ denote the set of all probability distributions over Y (seen as mappings from Y to $[0, 1]$). For $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, \ldots, n\}$. Let $[1, \infty]$ denote the set $\{x \in \mathbb{R} \mid x \geq 1\} \cup \{\infty\}$.

Suppose that someone wants to make a query to the database. If the data is (partially) private, the query output may leak some sensitive information about the data. Noise can be added to the output to reduce privacy leakage. One possible definition of privacy is that, from the output one should not be able to learn whether an individual row is present in the table or not. More generally, we may consider two *neighbouring* databases $x, x' \in X$, i.e. such that $d_X(x, x') = 1$.

Definition 1 (differential privacy, [9])**.** Let X be a metric space and $f : X \to \mathcal{D}(Y)$. The mapping f is (ϵ, δ)-*differentially private* if for all (measurable) $Y' \subseteq Y$, and for all x, x', where $d_X(x, x') = 1$, the following inequality holds:

$$Pr[f(x) \in Y'] \leq e^{\epsilon} Pr[f(x') \in Y'] + \delta \ . \qquad (1)$$

The mapping f is ϵ-*differentially private* if it is $(\epsilon, 0)$-differentially private.

The noise magnitude depends on the difference between the outputs of f. The more different the outputs are, the more noise we need to add to make them indistinguishable from each other. This is quantified by the global sensitivity of f.

Definition 2 (global sensitivity)**.** For $f : X \to Y$, the *global sensitivity* of f is $GS_f = \max_{x,x' \in X} \frac{d_Y(f(x), f(x'))}{d_X(x, x')}$.

Sensitivity is the main tool in arguing the differential privacy of various information release mechanisms. For mechanisms that add noise to the query output, this value serves as a parameter for the noise distribution. The noise is proportional to GS_f. One suitable noise distribution is $Lap(\frac{GS_f}{\epsilon})$, where $Lap(\lambda)(z) \propto e^{-|z|/\lambda}$. The sampled noise is sufficient to make the output of f differentially private, regardless of the input of f.

Differential privacy itself can also be seen as an instance of sensitivity. Define the following distance d_{dp} over $\mathcal{D}(Y)$:

$$d_{\mathrm{dp}}(\chi, \chi') = \inf\{\epsilon \in \mathbb{R}^+ \mid \forall y \in Y : |\ln(\chi(y)/\chi'(y))| \leq \epsilon\}.$$

Then a mechanism $\mathcal{M}$ from X to Y is ϵ-DP iff it is ϵ-sensitive with respect to the distances d_X on X and d_{dp} on $\mathcal{D}(Y)$.

3.2 Local and Smooth Sensitivity

Our work extends the results of [24], which makes use of instance-based additive noise. Since noise is always added to the output of a query that is applied to a particular state of the database, and some state may require less noise than the other, the noise magnitude may depend on the data to which the function is applied.

Definition 3 (local sensitivity)**.** For $f : X \to Y$, an integer-valued metric $d_X : X \times X \to \mathbb{N}$, and $x \in X$, the *local sensitivity* of f at x is $LS_f(x) = \max_{x' \in X : d_X(x,x')=1} d_Y(f(x), f(x'))$.

The use of local sensitivity may allow the use of less noise, particularly when the global sensitivity of f is unbounded. However, $LS_f(x)$ may not be directly used to determine the magnitude of the noise, because this magnitude may itself leak something about x. To solve this problem, Nissim et al. [24] use a *smooth upper bound* on $LS_f(x)$. It turns out that such an upper bound is sufficient to achieve differential privacy for f; potentially with less noise than determined by GS_f.

Definition 4 (smooth bound)**.** For $\beta > 0$, a function $S : X \to \mathbb{R}^+$ is a β*-smooth upper bound* on f if it satisfies the following requirements:
- $\forall x \in X : S(x) \geq f(x)$;
- $\forall x, x' \in X : S(x) \leq e^{\beta \cdot d_X(x,x')} S(x')$.

Nissim et al. [24] showed how to add noise based on the smooth bound on LS_f. The statement that we present in Theorem 1 is based on combination of Lemma 2.5 and Example 2 of [24].

Definition 5 (generalized Cauchy distribution)**.** For a parameter $\gamma \in \mathbb{R}_+$, $\gamma > 1$, the *generalized Cauchy distribution* $GenCauchy(\gamma) \in \mathcal{D}(\mathbb{R})$ is given by the proportionality

$$GenCauchy(\gamma)(x) \propto 1/(1 + |x|^\gamma) \ .$$

Theorem 1 (local sensitivity noise [24])**.** *Let η be a fresh random variable sampled according to $GenCauchy(\gamma)$. Let $\alpha = \frac{\epsilon}{4\gamma}$ and $\beta = \frac{\epsilon}{\gamma}$. For a function $f : X \to \mathbb{R}$, let $S : X \to \mathbb{R}$ be a β-smooth upper bound on the local sensitivity of f. Then the information release mechanism $f(x) + \frac{S(x)}{\alpha} \cdot \eta$ is ϵ-differentially private.*

3.3 Norms and Banach Spaces

The local sensitivity in Def. 3 is defined for integer-valued metrics on inputs. Let us try to generalize it to real-valued metrics. We may choose a small value $\delta > 0$ and round d_X up to the nearest higher multiple of δ:

$$\tilde{d}_X^\delta(x, x') := \delta \left\lceil \frac{d_X(x, x')}{\delta} \right\rceil$$

It is easy to see that $\tilde{d}_X^\delta$ is still a metric. Then we can define local sensitivity as

$$LS_f^\delta(x) = \frac{1}{\delta} \max_{x' \in X : \tilde{d}_X^\delta(x,x')=\delta} d_Y(f(x), f(x')) \ .$$

This is very similar to rescaled Def. 3. Finally, we define

$$\begin{aligned} LS_f(x) &= \lim_{\delta \to 0} LS_f^\delta(x) \\ &= \lim_{\delta \to 0} \frac{1}{\delta} \Big(\max_{x' \in X : 0 < d_X(x,x') \leq \delta} d_Y(f(x), f(x')) \Big) \end{aligned}$$

This looks quite similar to the definition of derivative (except for the use of max). If $X = Y = \mathbb{R}$ and d_X and d_Y are the absolute-value metrics then this will be equal to the absolute value of the derivative f' of f at x if $f'(x)$ exists. In this case we get $LS_f(x) = |f'(x)|$. Because it is based on derivative, we call it *derivative sensitivity* and write $\mathrm{DS}[f](x) = |f'(x)|$.

We would like to extend derivative sensitivity to metrics other than absolute value and to functions with more than one variable. One such extension of derivative is the *Fréchet derivative* in Banach spaces.

First, we recall some basics of Banach space theory. Throughout this paper, we denote vectors by $\mathbf{x}$, and norms by $\|\cdot\|_N$, where N specifies the particular norm. Banach spaces do not allow arbitrary metrics and instead require norms, but many useful metrics can also be viewed as norms.

Definition 6 (norm and seminorm)**.** A *seminorm* is a function $\|\cdot\| : V \to \mathbb{R}$ from a vector space V, satisfying the following axioms for all $\mathbf{x}, \mathbf{y} \in V$:
- $\|\mathbf{x}\| \geq 0$;
- $\|\alpha\mathbf{x}\| = |\alpha| \cdot \|\mathbf{x}\|$ (implying that $\|\mathbf{0}\| = 0$);
- $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (triangle inequality).

Additionally, if $\|\mathbf{x}\| = 0$ holds only if $\mathbf{x} = \mathbf{0}$, then $\|\cdot\|$ is a *norm*.

Some of the most useful norms in practice are ℓ_1 (i.e., sum), ℓ_2 (geographical distance), and ℓ_∞ (maximum). These are instances of ℓ_p-norms.

Definition 7 (ℓ_p-norm)**.** Let $X_i \subseteq \mathbb{R}$, $p \in [1, \infty]$. The ℓ_p-norm of $\mathbf{x} \in X_1 \times \cdots \times X_n$, denoted $\|\mathbf{x}\|_p$ is defined as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} .$$

For $p = \infty$, ℓ_∞ is defined as

$$\|\mathbf{x}\|_\infty = \lim_{p\to\infty} \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p} = \max_{i=1}^{n} |x_i| .$$

Definition 8 (Banach space)**.** A *Banach space* is a vector space with a norm that is *complete* (i.e. each converging sequence has a limit).

Banach spaces combine vector spaces with distances, which are necessary for defining differential privacy. The completeness property allows us to define derivatives. Using the norm of a Banach space, we may generalize the notion of continuous function from real numbers to Banach spaces.

Definition 9 (Continuous function in Banach space)**.** Let V and W be Banach spaces, and $U \subset V$ an open subset of V. A function $f : U \to W$ is called *continuous* if

$$\forall \epsilon > 0 : \exists \delta > 0 : \left\|x - x'\right\|_V \le \delta \Rightarrow \left\|f(x) - f(x')\right\|_W \le \epsilon.$$

The notion of the derivative of a function can be also extended to Banach spaces.

Definition 10 (Fréchet derivative)**.** Let V and W be Banach spaces, and $U \subset V$ an open subset of V. A function $f : U \to W$ is called *Fréchet differentiable at* $x \in U$ if there exists a bounded linear operator $df_x : V \to W$ such that $\lim_{h\to 0} \frac{\|f(x+h)-f(x)-df_x(h)\|_W}{\|h\|_V} = 0$. Such operator df_x is called Fréchet derivative of f at the point x.

The mean value theorem can be generalized to Banach spaces (to a certain extent).

Theorem 2 (Mean value theorem ([4], Chapter XII))**.** *Let V and W be Banach spaces, and $U \subset V$ an open subset of V. Let $f : U \to W$, and let $x, x' \in U$. Assume that f is defined and is continuous at each point $(1-t)x + tx'$ for $0 \le t \le 1$, and differentiable for $0 < t < 1$. Then there exists $t^* \in (0,1)$ such that*

$$\left\|f(x) - f(x')\right\|_W \le \|df_z\|_{V\to W} \left\|x - x'\right\|_V$$

for $z = (1-t^*)x + t^*x'$, where $\|\cdot\|_{V\to W}$ denotes the norm of operator that maps from V to W.

4 Metrics and Derivative Sensitivity

4.1 Banach Spaces of Databases

Let us have a database with a number of tables. The schema for each table is fixed. We see that database as a point in some Banach space (thus the distance between databases is the norm of their difference), where each cell in each table corresponds to a dimension of that space. As dimensions of Banach spaces are fixed (indeed, they are vector spaces $\mathbb{R}^n$ with some extra structure), the number of rows in each table is also fixed, and each row can be seen to have a public *identity* (similarly to [17]). In this paper, we consider the following *composite* norms and seminorms as possible norms for databases:

Definition 11 (composite seminorm)**.** Let $\|\cdot\|_N$ be a seminorm of the vector space $\mathbb{R}^n$. It is a *composite seminorm* if one of the following holds for all $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$:
- There exists $i \in [n]$, such that $\|\mathbf{x}\|_N = |x_i|$. Such seminorm *uses the variable x_i*.
- There exists a composite seminorm $\|\cdot\|_M$ and $a \in \mathbb{R}^+$, such that $\|\mathbf{x}\|_N = a \cdot \|\mathbf{x}\|_M$. The seminorm $\|\cdot\|_N$ *uses the same variables* as $\|\cdot\|_M$.
- There exist composite seminorms $\|\cdot\|_{M_1}, \ldots, \|\cdot\|_{M_k}$ and $p \in [1, \infty]$, s.t. $\|\mathbf{x}\|_N = \left\|\|\mathbf{x}\|_{M_1}, \ldots, \|\mathbf{x}\|_{M_k}\right\|_p$. The seminorm $\|\cdot\|_N$ uses the union of the variables used by all $\|\cdot\|_{M_i}$.

Let $\mathsf{vars}(N)$ be the set of variables used by $\|\cdot\|_N$.

We normally define the norms for rows of each table using the constructions allowed in Def. 11. We then state that the norm of the table is some ℓ_p-norm of the vector of the norms of its rows (last item of Def. 11) and the norm of the database is some ℓ_p-norm of the vector of the norms of its tables. Note that the notion of *seminorm* is used only for building blocks of composite ℓ_p-norms, and a seminorm constructed as in Def. 11 becomes a norm if $\mathsf{vars}(N) = \{x_1, \ldots, x_n\}$.

4.2 From Derivative Sensitivity to DP

Having defined a database as a point in some Banach space, we propose constructions that allow us to achieve differential privacy for functions defined over Banach spaces.

Definition 12. Let X be (an open convex subset of) a Banach space. Let $f : X \to \mathbb{R}$. Let f be Fréchet differentiable at each point of X. The *derivative sensitivity* of f is the following mapping from X to $\mathbb{R}_+$:

$$\mathrm{DS}[f](\mathbf{x}) = \|df_{\mathbf{x}}\| \ .$$

where $df_{\mathbf{x}}$ is the Fréchet derivative of f at $\mathbf{x}$ and $\|df_{\mathbf{x}}\|$ is the operator norm of $df_{\mathbf{x}}$.

Similarly to the local sensitivity of [24], we will need to find smooth upper bounds on derivative sensitivity to compute the noise. We extend the definition of smoothness (Def. 4) to the case where X is any Banach space:

Definition 13. Let $p : X \to \mathbb{R}$ and $\beta \in \mathbb{R}$. The mapping p is *β-smooth*, if $p(\mathbf{x}) \leq e^{\beta \cdot \|\mathbf{x}' - \mathbf{x}\|} \cdot p(\mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}' \in X$.

The next theorem shows how to compute the magnitude of noise to be added to $f(\mathbf{x})$ in order to obtain a differentially private information release mechanism. A smooth upper bound to $\mathrm{DS}[f]$ plays the central role here. We consider the same noise distributions as in [24].

Theorem 3. *Let $\gamma, b, \beta \in \mathbb{R}_+$, $\gamma > 1$. Let $\epsilon = (\gamma + 1)(b + \beta)$. Let η be a random variable distributed according to GenCauchy(γ). Let c be a β-smooth upper bound on $\mathrm{DS}[f]$ for a function $f : X \to \mathbb{R}$. Then $g(\mathbf{x}) = f(\mathbf{x}) + \frac{c(\mathbf{x})}{b} \cdot \eta$ is ϵ-differentially private.*

Theorem 3 can be proven similarly to Theorem 1, which has been done in [24]. The main difference is that, since we are using derivative sensitivity instead of local sensitivity, we apply the mean value theorem to find an upper bound on $\|f(x) - f(y)\|$ for two neighbouring databases x and y. The detailed proof can be found in App. C.2.

Cauchy noise has heavy tails. In its stead, we may use Laplace noise to achieve DP, but this requires a more complex proof and only gives us (ϵ, δ)-DP. The strength of our result depends on a technical condition on the β-smooth upper bound on $\mathrm{DS}[f]$, for which the key notion is the following.

Definition 14. A *path* in a Banach space X is a continuous function $h : [0, 1] \to X$. The path h is *shortest*, if for all $x_1, x_2, x_3 \in [0, 1]$, $x_1 \leq x_2 \leq x_3$, the equality $d_X(h(x_1), h(x_3)) = d_X(h(x_1), h(x_2)) + d_X(h(x_2), h(x_3))$ holds.

Theorem 4. *Let $b, \beta, \epsilon \in \mathbb{R}_+$, $b > 0$, $b + \beta \leq \epsilon$. Define $k = 1 + (\epsilon - b)/\beta$. Let $\delta = e^{-k}$. Let η be a random variable distributed according to $Lap(1)$. Let c be a β-smooth upper bound on $\mathrm{DS}[f]$ for a function $f : X \to \mathbb{R}$, where X is Banach space and d_X is the distance corresponding to the norm of X. Define $g(\mathbf{x}) := f(\mathbf{x}) + \frac{c(\mathbf{x})}{b} \cdot \eta$. Then g is $(\epsilon, 2e^{\epsilon}\delta)$-differentially private. If, additionally for any two points $\mathbf{x}_1, \mathbf{x}_2 \in X$ there exists a shortest path between them, such that c is monotonic along that path, then the factor "2" may be removed.*

This theorem is proved in App. C.3.

4.3 Computing Derivative Sensitivity and Smooth Upper Bounds

Given a description of a function $f : \mathbb{R}^n \to \mathbb{R}$, how do we determine (a β-smooth upper bound) on its derivative sensitivity? This question is ill-posed, because we have not stated the norm on $\mathbb{R}^n$; Def. 11 gives us many possibilities to define it. If two norms are related, then the derivative sensitivity with respect to one of them should tell us something about the other, too.

A composite seminorm in $\mathbb{R}^n$ can be seen as the semantics of a formal expression over the variables $x_1, \ldots, x_n$, where the term constructors are $\|\ldots\|_p$ of any arity. We write $N \preceq M$ for two seminorms N and M, if $\|\mathbf{x}\|_N \leq \|\mathbf{x}\|_M$ for all $\mathbf{x} \in \mathbb{R}^n$. The following results about $\preceq$ are proved in App. C.4.

Lemma 1. *Let N be a composite norm over $\mathbf{x} = (x_1, \ldots, x_n)$. Let composite seminorms N' and $V_1, \ldots, V_m$ be such, that $N = N'(V_1, \ldots, V_m)$, and for all $i \in [n]$ let W_i be a seminorm such that $V_i \preceq W_i$. Then, $N'(V_1, \ldots, V_m) \preceq N'(W_1, \ldots, W_m)$.*

Lemma 2. *Let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$. Let N be a composite norm, defined over variables $\mathbf{x}$. There exist $0 \leq \alpha_i, \beta_i \in \mathbb{R}$ for $i \in [n]$, such that $\|\alpha_1 x_1, \ldots, \alpha_n x_n\|_p \leq \|x_1, \ldots, x_n\|_N \leq \|\beta_1 x_1, \ldots, \beta_n x_n\|_q$, where:*
- *p is the largest ℓ_p-norm constructor in N;*
- *q is the smallest ℓ_p-norm constructor in N.*

The following three lemmas give us the basic combinators for statements about derivative sensitivity, reducing the task of finding the derivative sensitivity of a particular function with respect to a particular norm on its domain, to a series of tasks from basic calculus. We prove these lemmas in App. C.5.

Lemma 3. *Let $f : \mathbb{R}^n \to \mathbb{R}$, and let $\mathbb{R}^n$ be equipped with the norm ℓ_p. Then $\|df_{\mathbf{x}}\|$ is the ℓ_q-norm of the gradient* *vector* $\nabla f(\mathbf{x})$*, where* $q = \frac{p}{p-1}$ *(if* $p = 1$ *then* $q = \infty$ *and* vice versa*).*

The ℓ_q-norm is the *dual norm* of the ℓ_p-norm; we denote q by $\mathsf{dual}(p)$.

Lemma 4. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *have the derivative sensitivity* g *with respect to the norm* N*. Let* $a \cdot N$ *denote the scaling of the output of the norm* N *by* $a \in \mathbb{R}_+$*. Then* f *has the derivative sensitivity* g/a *with respect to* $a \cdot N$*.*

Lemma 5. *(a) Let* $(V_1, \|\cdot\|_{V_1})$ *and* $(V_2, \|\cdot\|_{V_2})$ *be Banach spaces. Let* $V = V_1 \times V_2$*. Let for all* $(v_1, v_2) \in V$*,*

$$\|(v_1, v_2)\|_V = \|(\|v_1\|_{V_1}, \|v_2\|_{V_2})\|_p$$

Then $(V, \|\cdot\|_V)$ *is a Banach space.*
(b) Suppose furthermore that a function $f : V \to \mathbb{R}$ *is differentiable at each point of* V*. Fix a point* $v = (v_1, v_2) \in V$*. Let* $g : V_1 \to \mathbb{R}$ *be such that* $g(x_1) = f(x_1, v_2)$ *and* $h : V_2 \to \mathbb{R}$ *be such that* $h(x_2) = f(v_1, x_2)$*. Let* $c_1 = \mathrm{DS}[g](v_1)$ *and* $c_2 = \mathrm{DS}[h](v_2)$*. Then* $\mathrm{DS}[f](v) = \|(c_1, c_2)\|_{\mathsf{dual}(p)}$*.*

These lemmas reduce the computation of the derivative sensitivity of a function to the computation of certain partial derivatives. We note that while the computation of the local sensitivity of a mapping may have a high computational complexity [24], the continuity of our functions ensures that the computation of the derivatives falls into the polynomial complexity class under some mild conditions [21].

Example. Consider computing differentially privately the time that a ship takes to reach the port. This time can be expressed as

$$f(x, y, v) = \frac{\|x, y\|_2}{v}, \quad f : \mathbb{R}^3 \to \mathbb{R}$$

where (x, y) are the coordinates of the ship (with the port at $(0, 0)$) and v is the speed of the ship. When thinking about sensitivity of the information, and expressing it in terms of the change to the variables x, y, v, we consider the distance between geographic locations to be the Euclidean distance, and the distance between speeds to be their arithmetic difference. We are interested in hiding both the location and the speed, and would like to combine the distances by adding them up, i.e. a change in both the location and speed is considered a greater change than only one of these two changes. As the change in location and change in speed are measured in different units, and may have different significance, we add scalings to these two norms. Hence we want to compute the sensitivity of f with respect to the norm $\|a \cdot \|x, y\|_2, b \cdot \|v\|_1\|_1$. Here $a, b \in \mathbb{R}_+$ are scalings.

Let $f^v(x, y) = f^{x,v}(y) = f^{y,v}(x) = f^{x,y}(v) = f(x, y, v)$, where the f-s with superscripts are considered functions with less arguments; the superscripts are interpreted as parameters instead.

The partial derivatives of f (as well as f-s with superscripts) are

$$\begin{aligned} \frac{\partial f}{\partial x} = \frac{\partial f^v}{\partial x} = \frac{df^{y,v}}{dx} &= \frac{x}{v\sqrt{x^2 + y^2}} \\ \frac{\partial f}{\partial y} = \frac{\partial f^v}{\partial y} = \frac{df^{x,v}}{dy} &= \frac{y}{v\sqrt{x^2 + y^2}} \\ \frac{\partial f}{\partial v} = \frac{df^{x,y}}{dv} &= -\frac{\sqrt{x^2 + y^2}}{v^2} \end{aligned} \quad (2)$$

By Lemma. 3, the absolute values of these derivatives are the derivative sensitivities of $f^{y,v}$, $f^{x,v}$ and $f^{x,y}$, respectively. Also by Lemma. 3, the derivative sensitivity of f^v with respect to the ℓ_2-norm $\|x, y\|_2$ is

$$\mathrm{DS}[f^v](x, y) = \|(\mathrm{DS}[f^{y,v}](x), \mathrm{DS}[f^{x,v}](y))\|_2 = \frac{\sqrt{2}}{|v|} \ . \quad (3)$$

By Lemma. 4, we have to divide (3) and (2) by respectively a and b, in order to obtain the derivative sensitivities of f^v and $f^{x,y}$ with respect to the scaled norms $a \cdot \|x, y\|_2$ and $b \cdot \|v\|_1$. Finally, we use Lemma 5 to obtain the derivative sensitivity of f:

$$\begin{aligned} \mathrm{DS}[f](x, y, v) = \|(\mathrm{DS}[f^v](x, y), \mathrm{DS}[f^{x,y}](v))\|_\infty = \\ \max\{\sqrt{2}/(a|v|), \sqrt{x^2 + y^2}/(bv^2)\} \ . \quad \blacksquare \end{aligned} \quad (4)$$

In this example, we have used the ℓ_1 norm to compare speeds. This is perhaps not the most useful measure, because it states that the difference between speeds “1” and “2” is the same as between the speeds “101” and “102”, although the first change would affect the arrival time greatly, and the second, not so much. A more reasonable approach is to apply the norm not to the speed v itself, but to something derived from it. One reasonable choice seems to be $1/v$. In this way, a multiplicative change in speed would correspond to a multiplicative change in the arrival time. An even better way to define the norm on speeds is to apply the ℓ_1 norm to $\ln v$. We explore this variant below.

To achieve differential privacy, we need to find a smooth upper bound on the derivative sensitivity. We have combining lemmas for deriving the smoothness properties of functions from Banach spaces, and their derivative sensitivities. The following lemma, proved in App. C.6, gives an alternative definition of β-smoothness. It is easier to use, and implies Def. 13.

Lemma 6. *Let X be a Banach space. If $\mathrm{DS}[f]$ exists then $f : X \to \mathbb{R}$ is β-smooth if $\frac{\mathrm{DS}[f](x)}{|f(x)|} \leq \beta$ for all $x \in X$.*

As a particular instance of Lemma 6, a differentiable function $f : \mathbb{R} \to \mathbb{R}$ is β-smooth if $\left|\frac{f'(x)}{f(x)}\right| \leq \beta$.

The following three lemmas describe the different possibilities for computing β-smooth upper bounds for composite functions. They are proved in App. C.7.

Lemma 7. *Let $f(x) : \mathbb{R} \to \mathbb{R}$ be β_f-smooth, and let $g(x) : \mathbb{R} \to \mathbb{R}$ be β_g-smooth.*
1. *If $f(x), g(x) > 0$, then $f(x) + g(x)$ is $\max(\beta_f, \beta_g)$-smooth;*
2. *$f(x) \cdot g(x)$ is $\beta_f + \beta_g$-smooth;*
3. *$f(x) \,/\, g(x)$ is $\beta_f + \beta_g$-smooth.*

Lemma 8. *Let X_i for $i \in \{1, \ldots, n\}$ be Banach spaces, and let $X = \prod_{i=1}^{n} X_i$. Let $f_i : X_i \to \mathbb{R}$ be β_i-smooth. Then, $f(x_1, \ldots, x_n) = \|f_1(x_1), \ldots, f_n(x_n)\|_p$ is $\|(\beta_i)_{i=1}^n\|_p$-smooth as well as $\max_{i=1}^n(\beta_i)$-smooth, where the norm of X is the $\ell_{\mathsf{dual}(p)}$-combination of the norms of all X_i.*

In general, the smoothness is worse if the variables of different f_i are not disjoint, as shown in the next lemma.

Lemma 9. *Let X_i for $i \in \{1, \ldots, n\}$ be Banach spaces, $X = \prod_{i=1}^{n} X_i$. Let $f_i : X \to \mathbb{R}$ be β_i^j-smooth for X_j. Let $x = (x_1, \ldots, x_n)$. Then, $f(x) = \|f_1(x), \ldots, f_n(x)\|_p$ is $\left\|(\max_j \beta_i^j)_{i=1}^n\right\|_p$-smooth.*

Example. Extending the previous example, let us find a smooth upper bound for the derivative sensitivity of the function expressing the time it takes for some ship (out of n) to reach the port at the coordinates $(0, 0)$. This $3n$-variable function is

$$f(x_1, y_1, v_1, \ldots, x_n, y_n, v_n) = \min_{i=1}^{n} \frac{\|x_i, y_i\|_2}{v_i} \quad .$$

Note that v_i is in the power -1 and we do not know how to find the smooth derivative sensitivity of the function $f_{v,i}(v_i) = v_i^{-1}$ (we only know how to do it for power functions with exponent ≥ 1). Let us define $w_i = \zeta \ln v_i$. The coefficient ζ is used to control the distance by which the whole input vector changes if $\ln v_i$ is changed by 1. Similarly, we add a coefficient to the geographical coordinates: $s_i = \alpha x_i, t_i = \alpha y_i$. The ratio between ζ and α shows how much the privacy of the speed is valued compared to privacy of the geographic coordinates, *relative to each other*. The smaller ζ/α is, the *more* we value the privacy of the speed.

We consider $(s_1, t_1, w_1, \ldots, s_n, t_n, w_n)$ as an element of the Banach space $(\mathbb{R}^{3n}, \|\ \|)$ where

$$\|(s_1, t_1, w_1, \ldots, s_n, t_n, w_n)\| = \|(\|(\|(s_1, t_1)\|_2, w_1)\|_1, \ldots, \|(\|(s_n, t_n)\|_2, w_n)\|_1)\|_p$$

Then $v_i = e^{w_i/\zeta}, x_i = \frac{s_i}{\alpha}, y_i = \frac{t_i}{\alpha}$ and

$$g(s_1, t_1, w_1, \ldots, s_n, t_n, w_n) = \frac{1}{\alpha} \min_{i=1}^{n} \frac{\|s_i, t_i\|_2}{e^{w_i/\zeta}}$$

Now the derivative sensitivity of $g_{w,i}(w_i) = e^{-w_i/\zeta}$ is

$$\mathrm{DS}[g_{w,i}](w_i) = \frac{1}{\zeta} e^{-w_i/\zeta}$$

which is $\frac{1}{\zeta}$-smooth. The function $g_{w,i}$ itself is also $\frac{1}{\zeta}$-smooth. The derivative sensitivity of $g_{st,i}(s_i, t_i) = \|s_i, t_i\|_2$ in $(\mathbb{R}^2, \ell_2)$ is

$$\mathrm{DS}[g_{st,i}](s_i, t_i) = 1$$

which is β-smooth for all β. The function $g_{st,i}$ is $\frac{1}{\zeta}$-smooth if $\frac{1}{\|s_i, t_i\|_2} \leq \frac{1}{\zeta}$, i.e. if $\|s_i, t_i\|_2 \geq \zeta$. A $\frac{1}{\zeta}$-smooth upper bound on $g_{st,i}$ is

$$\hat{g}_{st,i}(s_i, t_i) = \begin{cases} \|s_i, t_i\|_2 & \text{if } \|s_i, t_i\|_2 \geq \zeta \\ \zeta e^{\frac{\|s_i, t_i\|_2}{\zeta} - 1} & \text{otherwise} \end{cases}$$

An upper bound on the derivative sensitivity of $g_i(s_i, t_i, w_i) = \frac{\|s_i, t_i\|_2}{e^{w_i/\zeta}}$ is $c_{g_i}(s_i, t_i, w_i) =$

$$\begin{aligned} &\|(\mathrm{DS}[g_{st,i}](s_i, t_i) \cdot g_{w,i}(w_i), \mathrm{DS}[g_{w,i}](w_i) \cdot \hat{g}_{st,i}(s_i, t_i))\|_\infty \\ &= \left\| \left(1 \cdot e^{-w_i/\zeta}, \frac{1}{\zeta} e^{-w_i/\zeta} \cdot \hat{g}_{st,i}(s_i, t_i)\right) \right\|_\infty \\ &= \frac{\max\left(1, \frac{\hat{g}_{st,i}(s_i, t_i)}{\zeta}\right)}{e^{w_i/\zeta}} \quad , \end{aligned}$$

and it is $\frac{1}{\zeta}$-smooth because $\mathrm{DS}[g_{w,i}](w_i)$, $\mathrm{DS}[g_{st,i}]$, $g_{w,i}(w_i)$, and $\hat{g}_{st,i}(s_i, t_i))$ are $\frac{1}{\zeta}$-smooth.

A $\frac{1}{\zeta}$-smooth upper bound on $\mathrm{DS}[g]$ is

$$c(u) = \frac{\max_i c_{g_i}(s_i, t_i, w_i)}{\alpha} = \frac{1}{\alpha} \max_i \frac{\max\left(1, \frac{\hat{g}_{st,i}(s_i, t_i))}{\zeta}\right)}{e^{w_i/\zeta}}$$

where $u = (s_1, t_1, w_1, \ldots, s_n, t_n, w_n)$.

Now we can use Theorem 3 to compute an ϵ-differentially private version of g:

$$h(u) = g(u) + \frac{c(u)}{b} \cdot \eta$$

$$\epsilon = (\gamma + 1)(b + \frac{1}{\zeta})$$

$$\gamma > 1, b > 0, \eta \sim \mathit{GenCauchy}(\gamma)$$

To compute an ϵ-differentially private version of f, we first transform $(x_1, y_1, v_1, \ldots, x_n, y_n, v_n)$ into u and then compute $h(u)$. ■

Continuous function	Approximated condition	name
$\dfrac{e^{\alpha x}}{e^{\alpha x}+1}$	$x \geq 0$	sigmoid
$\dfrac{2}{e^{-\alpha x}+e^{\alpha x}}$	$x = 0$	tauoid

Table 1. Continuous approximations of step functions

5 Application to SQL Queries

In this section, we describe how the theory of Sec. 4 can be applied to SQL queries, computing the smooth upper bounds of their derivative sensitivities, such that an appropriate amount of noise may be added to turn them differentially private. Our theory deals with functions that return a numeric value, so the query should return a single output. We consider queries of the form

SELECT *aggr expr* FROM t1 AS s1,...,tn AS sn
WHERE *condition* ,

where:
- *expr* is an expression over table columns, computed as a continuous function.
- *condition* is a boolean expression over predicates $P(x) \in \{x < 0, x = 0\}$, where x is an expression of the same form as *expr*. Since all functions have to be continuous, these predicates are computed using continuous approximations to the step functions, listed in Table. 1.
- *aggr* is one of the operations SUM, COUNT, MIN, MAX.

GROUP BY queries can be simulated by generating for each group a separate query, with a filter selecting that particular group. Hence, we can group either by a public or a discrete attribute to get a finite number of groups. We do not support DISTINCT queries, as we do not know how to efficiently continuously approximate a function that removes repeating elements from a list of arguments.

Let our database have n tables. Let R_i be the Banach space of the potential values of rows for the i-th table, and n_i the number of rows in this table. Let $T_i = R_i^{n_i}$ and $D = T_1 \times \cdots \times T_n$. Starting from the norms on R_i, and applying ℓ_p-norms, define a norm for D.

5.1 Derivative Sensitivity of Queries with Respect to a Component

Our ultimate goal is to enforce differential privacy w.r.t. a certain *component* of a database. A component is a rectangular subset of cells in a table, given by the subsets of columns and rows, into which they belong. The set of all (sensitive) entries is a vector over $\mathbb{R}^m$. The possible vectors of sensitive entries together with the norm forms a Banach space. As next we describe, how the SQL query is converted into a function from $\mathbb{R}^m$ to $\mathbb{R}$. In Sec. 5.2 and 5.3 we explain, how we compute a smooth upper bound of the derivative sensitivity of this function.

5.1.1 Query without a Filter

To make a query on the database, we want to join those n tables. Consider an input $(t_1, \ldots, t_n) \in D$. Let us first consider the cross product $t = t_1 \times \cdots \times t_n$, i.e. joining without any filters. First, let us assume that the n joined tables are distinct, i.e. no table is used more than once. Each row of the cross product is an element of $R = R_1 \times \cdots \times R_n$, thus t is an element of $T = R^{n_1 \cdots n_n}$.

The query contains an aggregating function $f : T \to \mathbb{R}$. All non-sensitive entries of the data tables are treated as constants. Suppose we want to compute the sensitivity of f w.r.t. a subset s of rows of t_i. Then the subset of rows of the cross product, filtered through s, is $t_s = t_1 \times \cdots \times t_{i-1} \times s \times t_{i+1} \times \cdots \times t_n$. Each row in t_s is depends from exactly one row in s.

Let $s = r_1, \ldots, r_k$ and let $t_s = \bigcup_{j=1}^{k} t_{r_j}$ where t_{r_j} is the subset of rows that depend from the row r_j. The sets of rows t_{r_j} are disjoint. Let $t_{r_j} = \{u_{j1}, \ldots, u_{jm_j}\}$. For each row u_{jk}, we select the same (semi)norm as the norm for the row r_j, with the additional columns not contributing to the norm. The norm for t_{r_j} is computed by combining the norms for $u_{j1}, \ldots, u_{jm_j}$ using ℓ_∞-norm. The norm for t_s is then computed by combining the norms for t_{r_j} using the norm that combined the norms of the rows of t_i, i.e. ℓ_{p_i}.

5.1.2 Query with a Filter

A filter that *does not depend* on sensitive data can be applied directly to the cross product of the input tables, and we may then proceed with the query without a filter. A filter that *does depend* on sensitive data is treated as a part of the query. We treat this filter as a continuous function, applied in such a way that the discarded rows would be ignored by the aggregating function. We combine sigmoids and tauoids to obtain the approximated value of the indicator $\sigma(x_i) \in \{0, 1\}$, denoting whether the row x_i satisfies the filter.

Hence, if the filter depends on sensitive data, we have the following set-up:
- There is a set of rows $\{x_1, \ldots, x_m\}$.
- There is a function f_i applied to the row x_i, returning a real number. For different rows, this function may be different, e.g. it may be determined by the public cells of the row.
- There is a filtering function σ_i applied to the row x_i. It returns a real number. It approximates a boolean condition, i.e. its values are mostly near 0 and 1.
- There is an aggregation function applied to a subset of the values $f_1(x_1), \ldots, f_m(x_m)$. Only such $i \in \{1, \ldots, m\}$ are selected, where the condition holds.

To convert the SQL query into a continuous function, the functions f_i and σ_i are combined as follows, depending on the aggregation function:
- SUM. The values 0 do not affect the sum, hence we compute the result as $\sum_{i=1}^{m} f_i(x_i) \cdot \sigma_i(x_i)$.
- COUNT: The values of f_i do not affect the result. We compute the result as $\sum_{i=1}^{m} \sigma(x_i)$, counting all entries for which $\sigma_i(x_i) = 1$. The sensitivity of such query only depends on the sensitivity of σ.
- MIN, MAX: If $\sigma_i(x_i)$ is 0, then we need to replace the actual value $f_i(x_i)$ with some large [resp. small] value that would not affect the result of MIN [resp. MAX]. Our conversion of the SQL query proceeds by first defining $\Delta := \mathrm{MAX}(f(x_1), \ldots, f(x_n)) - \mathrm{MIN}(f(x_1), \ldots, f(x_n))$, and then computing the result by applying MIN to the values $f_i(x_i) + (1 - \sigma_i(x_i)) \cdot \Delta$. MAX is computed similarly, changing the first "+" into a "−".

If we know that the compared values are integers and hence $d(x, x') \geq 1$ for $x \neq x'$, we can do better than using sigmoids or tauoids from Table 1, defining precise functions:
- $x > y \iff \min(1, \max(0, x - y))$.
- $x = y \iff 1 - \min(1, \max(0, |x - y|))$.

An advantage of these functions is that they do not lose precision due to addition and multiplication.

For real numbers, we may bound the precision and assume e.g. that $d(x, x') \geq 1/k$ for some $k \geq 1$, which allows to use similar functions. The sensitivity of such comparisons will be k times larger than for integers.

5.2 Inferring Derivative Sensitivities

Previously we saw, how to convert an SQL query into a continuous function from $\mathbb{R}^m$ to $\mathbb{R}$, where each variable refers to the contents of a particular cell in a table of the database. Using the lemmas in Sec. 4.3, we can compute smooth upper bounds to their derivative sensitivities. The results given there can be specified to the concrete computation steps arising in the conversions of SQL queries. These specifications are given below; they are used to convert a SQL query to another query that computes the smooth upper bound of the derivative sensitivity of the original query. Examples of the whole conversion workflow can be seen in App. B, where the database schema is described in App. B.1, the privacy-sensitive parts of the database in App. B.2, the considered SQL queries in App. B.3, and, finally, the conversion results in App. B.5. These conversion results include both the conversion of the original query to a query with smooth semantics, and the query that computes an upper bound on the derivative sensitivity.

Let the write-up $\mathrm{DS}[f] \leq'_N h \sim \beta$ mean that h is a β-smooth upper bound on the derivative sensitivity of f, according to the norm $\|\cdot\|_N$ on the domain of f. For compositions, we also need the upper bounds for the (absolute values of) functions f themselves; let $f \sim_N \beta$ denote that f is β-smooth, and $f \leq g \sim_N \beta$ denote that g is a β-smooth upper bound of $|f|$, again according to the norm $\|\cdot\|_N$ on the domain of f. Table 2 lists the smooth upper bounds of some simple uni- and multivariate functions and their derivative sensitivities (using absolute value as the norm on $\mathbb{R}$). These upper bounds are proved in App. C.8. For composite functions, the rules for computing the β-smooth upper bounds are given in Fig. 1. This summarizes the statements of lemmas of Sec. 4.3, which are proven in App. C.9.

5.3 Query Norm vs Database Norm

The facts and rules in Table 2 and Fig. 1 are in principle sufficient to compute the smooth upper bounds of derivative sensitivity of functions resulting from SQL queries with respect to *all* composite norms. Still, when we naïvely apply them, we end up finding the sensitivity for a particular norm that is somehow "natural" for the function. In practice, it may happen that we actually need sensitivity w.r.t. some different norm, because the data owner specified so. For example, we know how to compute the sensitivity w.r.t. the norm $\|x_1, x_2\|_1$, but are interested in differential privacy w.r.t. $\|x_1, x_2\|_2$.

$$\frac{\forall i : \mathrm{DS}[f_i] \preceq'_N h_i \sim \beta}{\mathrm{DS}[\sum_i c_i f_i] \preceq'_N \sum_i c_i h_i \sim \beta}(+_D)$$

$$\frac{\forall i : f_i \le g_i \sim_N \beta}{\sum_i c_i f_i \le \sum_i c_i g_i \sim_N \beta}(+_S)$$

$$\frac{f_i \le g_i \sim_N \beta_i}{f_1 \cdot f_2 \le g_1 \cdot g_2 \sim_N \beta_1 + \beta_2}(*_S)$$

$$\frac{f_i \le g_i \sim_{N_i} \beta \quad \mathsf{vars}(f_1) \perp \mathsf{vars}(f_2)}{f_1 \cdot f_2 \le g_1 \cdot g_2 \sim_{N_1+N_2} \beta}(*_S^{\perp})$$

$$\frac{\mathrm{DS}[f] \preceq'_N g \sim \beta \quad N \preceq M}{\mathrm{DS}[f] \preceq'_M g \sim \beta}(\preceq_D)$$

$$\frac{f \sim_N \beta}{f \sim_{a \cdot N} \beta / a}(N_S)$$

$$\frac{f_2 \sim_{|\cdot|} \beta \quad \forall \mathbf{x} : \mathrm{DS}[f_1](\mathbf{x}) \le B}{f_2 \circ f_1 \sim_N \beta B}(\circ_S)$$

$$\frac{\mathrm{DS}[f_i] \preceq'_{N_i} h_i \sim \beta_i \quad \forall i, j : \mathsf{vars}(N_i) \perp \mathsf{vars}(N_j) \wedge f_i \cdot f_j \ge 0}{\mathrm{DS}[\sum_{i=1}^k f_i] \preceq'_{\ell_p(N_1,\dots,N_k)} \|h_1,\dots,h_k\|_{\mathsf{dual}(p)} \sim \|\beta_1,\dots,\beta_k\|_{\mathsf{dual}(p)}}(+_D^{\perp})$$

$$\frac{f_i \le g_i \sim_N \beta_i \quad \mathrm{DS}[f_i] \preceq'_N h_i \sim \beta'_i}{\mathrm{DS}[f_1 \cdot f_2] \preceq'_N g_1 \cdot h_2 + g_2 \cdot h_1 \sim \max(\beta_1 + \beta'_2, \beta'_1 + \beta_2)}(*_D)$$

$$\frac{f_i \le g_i \sim_{N_i} \beta \quad \mathrm{DS}[f_i] \preceq'_{N_i} h_i \sim \beta \quad \mathsf{vars}(N_1) \perp \mathsf{vars}(N_2)}{\mathrm{DS}[f_1 \cdot f_2] \preceq'_{N_1+N_2} g_1 \cdot h_2 + g_2 \cdot h_1 \sim \beta}(*_D^{\perp})$$

$$\frac{\mathrm{DS}[f] \preceq'_N g \sim \beta \quad N \preceq M}{\mathrm{DS}[f] \preceq'_M g \sim \beta}(\preceq_D)$$

$$\frac{\mathrm{DS}[f] \preceq'_N g \sim \beta \quad \forall \mathbf{x} : \bar{g}(\mathbf{x}) = g(\mathbf{x})/a}{\mathrm{DS}[f] \preceq'_{a \cdot N} \bar{g} \sim \beta/a}(N_D)$$

$$\frac{\mathrm{DS}[f_1] \preceq'_N h_1 \sim \beta_1 \quad \forall \mathbf{x} : h_1(\mathbf{x}) \le B \quad f'_2 \sim_{|\cdot|} \beta_2}{\mathrm{DS}[f_2 \circ f_1] \preceq'_N |f'_2 \circ f_1| \cdot h_1 \sim \beta_2 B + \beta_1}(\circ_D)$$

$$\frac{\mathrm{DS}[f_i] \preceq'_{N_i} h_i \sim \beta \quad \forall i, j : \mathsf{vars}(N_i) \perp \mathsf{vars}(N_j)}{\mathrm{DS}[\min\{f_i,\dots,f_k\}] \preceq'_{\ell_p(N_1,\dots,N_k)} \max\{h_1,\dots,h_k\} \sim \beta}(\min_D^{\perp})$$

$$\frac{f_i \le g_i \sim_{N_i} \beta \quad \forall i, j : \mathsf{vars}(N_i) \perp \mathsf{vars}(N_j)}{\min\{f_1,\dots,f_k\} \le \min\{g_1,\dots,g_k\} \sim_{\ell_p(N_1,\dots,N_k)} \beta}(\min_S^{\perp})$$

Fig. 1. Upper bounds for composite functions

$f(x)$	**cond.s**	g, **s.t.** $f \le g \sim_{\lvert\cdot\rvert} \beta$	h, **s.t.** $\mathrm{DS}[f] \preceq'_{\lvert\cdot\rvert} h \sim \beta$
x^r	$r \ge 1$, $x > 0$	$\begin{cases} x^r & \text{if } x \ge \frac{r}{\beta} \\ \mathsf{pw}^r_\beta(x) & \text{oth.} \end{cases}$	$\begin{cases} r x^{r-1} & \text{if } x \ge \frac{r-1}{\beta} \\ r \cdot \mathsf{pw}^{r-1}_\beta(x) & \text{oth.} \end{cases}$

$f(x)$	**cond.s**	g, **s.t.** $f \le g \sim_{\lvert\cdot\rvert} \beta$	h, **s.t.** $\mathrm{DS}[f] \preceq'_{\lvert\cdot\rvert} h \sim \beta$
e^{rx}	$\lvert r\rvert \le \beta$	e^{rx}	$\lvert r\rvert e^{rx}$
c	$c \in \mathbb{R}$	c	0
$\frac{e^{\alpha x}}{e^{\alpha x}+1}$	$\beta \ge \alpha$	$\frac{e^{\alpha x}}{e^{\alpha x}+1}$	$\frac{\alpha e^{\alpha x}}{(e^{\alpha x}+1)^2}$
$\frac{e^{\alpha x}}{e^{\alpha x}+1}$	$\beta < \alpha$	1	$\frac{\alpha e^{\beta x}}{(e^{\beta x}+1)^2}$
$\frac{2e^{\alpha x}}{1+e^{2\alpha x}}$	$\beta \ge \alpha$	$\frac{2e^{\alpha x}}{1+e^{2\alpha x}}$	$\frac{2\lvert\alpha\rvert e^{\alpha x}}{1+e^{2\alpha x}}$

$f(\mathbf{x})$	g, **s.t.** $f \le g \sim_{\ell_p} \beta$	h, **s.t.** $\mathrm{DS}[f] \preceq'_{\ell_p} h \sim \beta$
$\lVert\mathbf{x}\rVert_p$	$\begin{cases} \lVert\mathbf{x}\rVert_p & \text{if } \lVert\mathbf{x}\rVert_p \ge \frac{1}{\beta} \\ \mathsf{pw}^1_\beta(\lVert\mathbf{x}\rVert_p) & \text{oth.} \end{cases}$	1

Here $\mathsf{pw}^r_\beta(x) = \left(\frac{r}{\beta}\right)^r \cdot e^{\beta x - r}$

Table 2. Upper bounds for uni- and multivariate functions

Let the *query norm* (denoted N_{qr}) be the norm for which we *can* compute derivative sensitivity. Let the *database norm* (denoted N_{db}) be the norm for which we *want to* compute derivative sensitivity.

If $N_{qr} \preceq N_{db}$, then the rule $(\preceq_D)$ allows us to use the computed $\mathrm{DS}[f]$ for N_{qr} also with the norm N_{db}. But if $N_{qr} \not\preceq N_{db}$, then we cannot directly use the sensitivity w.r.t. N_{qr}.

According to rule (N_D), if a the upper bound to the derivative sensitivity of the function f is β-smooth according to N_{qr}, then, its $\frac{1}{\alpha}$-times scaled version is $\frac{\beta}{\alpha}$-smooth in according to the norm $\alpha \cdot N_{qr}$ for any $\alpha > 0$. We compute sensitivity w.r.t. such norm $\alpha \cdot N_{qr}$, that $\alpha \cdot N_{qr} \preceq N_{db}$. The sensitivity becomes $\frac{\beta}{\alpha}$-smooth instead of β-smooth, which affects the amount of noise required to achieve differential privacy.

We show that a suitable α always exists for a composite norm (Def. 11) if N_{db} uses all the variables $x_1, \dots, x_n$. This assumption is reasonable: any variable that N_{db} does not use is not treated as sensitive, so we may treat it as a constant when computing the sensitivity, reducing the total number of variables. We can find α as follows.

1. Use Lemma 2 to get $a_i, b_i \ge 0$, $p, q > 0$ satisfying the conditions $\|a_1 x_1, \dots, a_n x_n\|_q \ge \|x_1, \dots, x_n\|_{N_{qr}}$ and $\|b_1 x_1, \dots, b_n x_n\|_p \le \|x_1, \dots, x_n\|_{N_{db}}$. We have $a \cdot \|x_1, \dots, x_n\|_p \ge \|a_1 x_1, \dots, a_n x_n\|_p$ for $a = \max_i a_i$, and $b \cdot \|x_1, \dots, x_n\|_p \le \|b_1 x_1, \dots, b_n x_n\|_p$ for $b = \min_i b_i$.
   We get $a \cdot \|x_1, \dots, x_n\|_p \ge \|x_1, \dots, x_n\|_{N_{qr}}$. Since N_{db} uses all variables $x_1, \dots, x_n$, we have $b_i \ne 0$ for all i, and hence $b \ne 0$. This allows to write $\|x_1, \dots, x_n\|_p \le \frac{1}{b} \cdot \|x_1, \dots, x_n\|_{N_{db}}$.
2. If $p \le q$, we have $a \cdot \|x_1, \dots, x_n\|_q \le a \cdot \|x_1, \dots, x_n\|_p$. If $p > q$, we can use equivalence of ℓ_p-norms that gives us $a \cdot \|x_1, \dots, x_n\|_q \le n^{1/q - 1/p} \cdot a \cdot \|x_1, \dots, x_n\|_p$. Let $c = (p \le q)\ ?\ 1\ :\ n^{1/q-1/p}$.
3. We have now come up with scalings a, b, c that satisfy $c \cdot a \cdot \|x_1, \dots, x_n\|_p \ge \|x_1, \dots, x_n\|_{N_{qr}}$, and $\|x_1, \dots, x_n\|_p \le \frac{1}{b} \cdot \|x_1, \dots, x_n\|_{N_{db}}$. Putting these inequalities together, we get $c \cdot a \cdot \frac{1}{b} \cdot \|x_1, \dots, x_n\|_{N_{db}} \ge \|x_1, \dots, x_n\|_{N_{qr}}$. By construction, we always have $c > 0$. It is possible that $a = 0$ only in the case

if $a_i = 0$ for all i, i.e. the query uses no sensitive variables, which is not the case. Take $\alpha = \frac{b}{c \cdot a}$.

If $N_{qr} = N'(V_1, \ldots, V_m)$ and $N_{db} = N'(W_1, \ldots, W_m)$ for some composite seminorms N', V_i, W_i, then it suffices to apply the aforementioned procedure only to such $i \in [m]$, where $V_i \succeq W_i$. Let α_i be such, that $\alpha_i \cdot V_i \preceq W_i$. By Lemma 1, we get $N'(\alpha_1 \cdot V_1, \ldots, \alpha_m \cdot V_m) \preceq N'(W_1, \ldots, W_m)$. We can now take $\alpha = \min_i \alpha_i$.

6 Precision and Utility

6.1 Choosing the Norm and ϵ

In the standard definition of differential privacy, we are concealing an addition/removal of *one* table row. In a Banach space, the notion of *unit change* can be different. Even if we have decided on the set of sensitive rows and columns, it may be unclear whether/which scaling of norm variables is reasonable. For example, if we scale the norm by a and keep noise level the same, the ϵ will increase proportionally to a, so we need to know which ϵ is "good enough" to choose appropriate a. For this, we need to understand what the table attributes mean. For example, if the length is presented in meters, and we want to conceal a change in a kilometer, we scale the location norm by 0.001 to capture a larger change.

To give a better interpretation to ϵ, we may relate it to other security definitions such as guessing probability advantage, as done e.g. in [22]. Adapting this approach to our metrics, we could answer questions like "how likely the attacker guesses that the location of an object is within X miles from the actual location".

Let $X' \subseteq X$ be the subset of inputs for which we consider the attacker guess as successful (e.g. he guesses an object's location coordinates precisely enough). Let the posterior belief of the adversary (after seeing the output) be expressed by the probability distribution $\mathbf{Pr}_{post}[\cdot]$. Let the prior belief (before seeing the output) be $\mathbf{Pr}_{pre}[\cdot]$, and f_X the corresponding probability density function, i.e. $\mathbf{Pr}_{pre}[X'] = \int_{X'} f_X(x)dx$. We need an upper bound on $\mathbf{Pr}_{post}[X'] = \mathbf{Pr}_{pre}[X' \mid y]$, where y is the observed output. Let f_Y be the probability density function of noisy outputs. Using Bayesian inference,

$$\mathbf{Pr}_{post}[X'] = \mathbf{Pr}_{pre}[X' \mid y] \leq \frac{1}{1 + \frac{\int_{X''} f_Y(y|x) f_X(x)dx}{\int_{X'} f_Y(y|x') f_X(x')dx'}}$$

for any $X'' \subseteq X \setminus X'$. Let $X'' := \{x \mid r < d(x, x') \leq a\}$ for some $a \in \mathbb{R}$ (e.g. $a := \sup_{x \in X} d(x, x')$ if it exists). Differential privacy gives us $\frac{\mathbf{Pr}[\mathcal{M}_q(x') \in Y]}{\mathbf{Pr}[\mathcal{M}_q(x) \in Y]} \leq e^{\epsilon \cdot a}$ for all Y, and hence also $\frac{f_Y(y|x')}{f_Y(y|x)} \leq e^{\epsilon \cdot a}$. We get

$$\mathbf{Pr}_{post}[X'] \leq \frac{1}{1 + e^{-\epsilon a} \cdot \frac{\mathbf{Pr}_{pre}[X'']}{\mathbf{Pr}_{pre}[X']}} .$$

The optimal value of a depends on the distribution of X. In practice, we may use brute force search for a.

6.2 Sigmoid Precision

In Sec. 5, we mentioned that we use sigmoids $\frac{e^{\alpha x}}{e^{\alpha x}+1}$ to approximate filtering of the form $x \geq 0$, where $\alpha > 0$ can be arbitrary. The derivative $\frac{\alpha e^{\alpha x}}{(e^{\alpha x}+1)^2}$ of a sigmoid is α-smooth.

To get a higher precision than that of an ordinary sigmoid but still maintain α-smoothness, we use an extra parameter a in addition to α. We use the sigmoid $\sigma(x) = \frac{e^{ax}}{e^{ax}+1}$ (note that the precision parameter is now a) but instead of its actual sensitivity $\sigma'(x)$, which would itself be a-smooth, we use $c(x) = \frac{ae^{\alpha x}}{(e^{\alpha x}+1)^2}$, which is an α-smooth upper bound on $\sigma'(x)$. The smooth sensitivity is $\frac{a}{\alpha}$ times higher than that of the original sigmoid but the difference from the precise filter value (0 or 1) is $\frac{e^{ax}+1}{e^{\alpha x}+1}$ times smaller. If the probability density function of x is roughly constant near $x = 0$ then, for all $y \in (0, 1)$, the probability that the difference from the precise filter value is larger than y is

$$\Pr(\frac{1}{e^{ax}+1} > y) = \Pr\left(x < \frac{\ln(\frac{1}{y} - 1)}{a}\right)$$

$$\approx \frac{\alpha}{a} \Pr(x < \frac{1}{\alpha} \ln(\frac{1}{y} - 1)) = \frac{\alpha}{a} \Pr(\frac{1}{e^{\alpha x}+1} > y)$$

i.e. $\frac{a}{\alpha}$ times smaller than in the original sigmoid.

The goodness of a depends on both the query and the data. Since a sigmoid error affects each row, precision becomes more important when the number of rows grows large. An optimal a will increase proportionally to $\sqrt{n}$, where n is the number of rows. The details are given in App. A.

6.3 Comparing Laplace and Cauchy noise

While generalized Cauchy distributions have heavy tails, the heaviness can be reduced by increasing the parameter γ. Taking $\gamma = 4$ seems to give a good balance between tail heaviness and median absolute value of noise. The 99.9999% quantile of $|GenCauchy(4)|$ is about 120 times its median which does not seem too much worse than $|Lap(1)|$, whose 99.9999% quantile is about 20 times its median.

Now let us compare noise magnitudes (which are roughly proportional to median absolute noise). When using Laplace noise, the smoothness β will need to be smaller (for a fixed ϵ) than for generalized Cauchy noise. This will in general increase the β-smooth upper bound $c(\mathbf{x})$. On the other hand, the value b will also be larger. Whether the noise magnitude $\frac{c(\mathbf{x})}{b}$ will be smaller for Laplace or for generalized Cauchy noise, depends on the concrete query.

Using Laplace noise does not seem to have enough advantages over generalized Cauchy noise to justify the more complex properties and worse privacy guarantee of (ϵ, δ)-DP over ϵ-DP. Hence, in this paper we only evaluate generalized Cauchy noise.

7 Implementation and Evaluation

7.1 Implementation

Our analyzer (available on GitHub) has been implemented in Haskell. As an input, it takes an SQL query, a database schema, and a description of the norm w.r.t. which we want to achieve differential privacy. We assume that each table contains a row ID of unique keys. For each table X, we expect a table named $X\_sensRows$ that contains the same column ID of keys, and another column *sensitive* of boolean values that tell for each row whether it is sensitive or not.

The analyzer computes *another query* (as a string) that describes the way in which derivative sensitivity should be computed. This new query represents the function $c(x)$ such that the additive noise would be $\frac{c(x)}{b} \cdot \eta$ for $\eta \leftarrow GenCauchy(\gamma)$, according to Theorem 3. In our analyzer, $\gamma = 4$ is fixed (as justified in Sec. 6.3), and $b = \epsilon/(\gamma + 1) - \beta$, where ϵ is the desired differential privacy level, and β the smoothness parameter, which is provided as an additional input. The resulting query is fed to a database engine to evaluate the sensitivity on particular data.

7.2 Evaluation

We performed evaluation on 4 x Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz laptop, Ubuntu 16.04.4 LTS, using PostgreSQL 9.5.14.

We have taken the queries of TCP-H set [1] for benchmarking. Most of these queries contain GROUP BY constructions with too many possible groups. We have simplified these queries, adding a filtering that chooses *one* particular group.

Another challenge comes from the filters. If some filter is "public" (i.e. does not depend on sensitive data), it is easier to apply it beforehand, so that the remaining table with "private" filters (that do depend on sensitive data and hence cannot be applied directly) would be as small as possible. While it is easy to do with a pure AND combination of filters, in practice public and private filters can be mixed, e.g. related by OR. We had to manually rewrite the filters in such a way that public filters would be easily extractable as separate members of an AND combination.

We generated TCP-H data with scale factors (SF) 0.1, 0.5, 1.0, denoting how much data is generated for the sample database. For 1.0, the size of the largest table is ca 6 million rows. The table schema, together with numbers of rows for different tables, is given in App. B.1. To define the database metric, we have considered integer, decimal, and date columns as sensitive, assigning to them different weights, described more precisely in App. B.2. All rows are considered sensitive. Row norms have been combined using ℓ_1-norm, which ensures differential privacy w.r.t. unit change in sensitive attribute of one row.

We adjusted (as described above) the queries $Q1$ (splitting a single query with 5 aggregations to 5 separate queries), $Q2$ (splitting it to 2 queries with MIN and MAX respectively), $Q3$-$Q11$, $Q12$ (splitting 2 aggregations to 2 queries), $Q16$, $Q17$, $Q19$ of the TCP-H dataset to our analyzer. The queries that have been eventually fed to the analyzer are listed in App. B.3. We treat date as an integer, i.e. the number of days passed from the date 1980-01-01. In App. B.4, we present more evaluation results, where we treat date as a floating point number, so that sigmoids can be used for filtering.

We fix $\epsilon = 1$. For derivative sensitivity experiments, we take sigmoid precision $\alpha = 5$ and smoothness $\beta = 0.1$. This choice gives $b = 0.1$, and the additive noise with 78% probability is below $10 \cdot c(x)$, where the value 78% comes from analyzing distribution $GenCauchy(4)$ (we have $\int_{-1}^{1} GenCauchy(4)(x)\ dx = 0.78$). Too large value of β makes b (and hence the noise) larger, and too small β makes the sensitivity larger, so β is a parameter that can in general be optimized.

7.2.1 Time

The time benchmarks are given in Table 3. Let x be the database instance. For each scale factor SF, we report the execution time t_i of the *initial query* $q_i(x)$, time t_m of the *modified query* $q_m(x)$ (i.e. in which filtering is replaced with continuous approximation), and time t_s of the *sensitivity-computing query* $q_s(x)$. The time spent to generate the queries q_m and q_s is negligible (below $20ms$), and it does not depend on the database size, so we do not report it. We also do not report the execution time of sampling the noise, as it does not depend on the database size either.

The total time overhead of computing noisy output based on derivative sensitivity is $t_m + t_s$: since the sensitivity has been computed for q_m, the noise should also be added to $q_m(x)$, and not to $q_i(x)$. We estimate the total time overhead for global sensitivity as t_i, as it is sufficient to execute $q_i(x)$, and the computation of global sensitivity does not depend on the database size.

We see that in general t_m and t_s are larger than t_i. This overhead comes from filtering. While in q_i the database engine may immediately drop all rows that do not satisfy the filter, in q_m and q_s we need to compute the approximated output and the sensitivity of *each row*. In overall, the time overhead of q_m and q_s compared to q_i (and hence of derivative sensitivity compared to global sensitivity) depends on the ratio of "number of rows before filtering" and "number of rows after filtering".

7.2.2 Precision

The precision benchmarks are given in Table 4. For each scale factor SF, we report the output $q_i(x)$ of the initial query, and $q_s(x)$ of the sensitivity-computing query. We report the output $q_m(x)$ of the modified query only if it is different from $q_i(x)$. The relative error has been computed as $\frac{|(q_m(x) \pm \xi) - q_i(x)|}{q_i(x)} \cdot 100$, where $\xi = \frac{c(x)}{b} = \frac{q_s(x)}{\epsilon/(\gamma+1)-\beta} = 10 \cdot q_s(x)$. The additive noise stays below ξ with probability 78% (as discussed above), so the relative error stays below reported value also with probability 78%.

The last two columns of Table 4 demonstrate the global sensitivity of queries, which is the same for all SF values, as it does not depend on data. The left column shows global sensitivity w.r.t. the same metric as the derivative sensitivity (we call it *non-standard*), and the right column w.r.t the row difference metric (we call it *standard*). We compare these with derivative sensitivity.

Global sensitivity w.r.t. non-standard metric. In the first case, we compute the global sensitivity w.r.t. the same metric as the derivative sensitivity. Even using the same metric, we cannot compare global (GS) and derivative sensitivity (DS) directly without taking into account particular noise generating mechanisms. However, in our results we have either GS=∞ or GS=DS. If GS=∞, then the noise would be ∞ as well for any noise generating mechanism. If GS=DS, then we expect the noise of GS to be lower, as e.g. employing the same Cauchy mechanism that we use for derivative sensitivity with $\beta \approx 0$ gives 10 times less noise than with $\beta = 0.1$ for the same sensitivity. In our benchmarks, DS gives advantage over GS in the following main cases.

1. When a sensitive attribute x_1 is multiplied by another attribute x_2, and there are no bounds on x_2, we get GS=∞, as $|(x_1 \pm 1) \cdot x_2 - x1 \cdot x2| = |x_2|$.
2. The norms of rows are combined into a table norm using ℓ_1-norm. Hence, $d(t, t') = 1$ covers not only the case where the norm of one row changes by 1, but also the case where *each* row changes a little. In an extreme case, all rows of t are already very close to the filtering bound, and the filtering function returns ≈ 1 for all rows in t, and ≈ 0 for all rows in t'. This makes no difference for a COUNT query (as in $b1\_5$), as the sum of all these changes is still 1, but we get GS=∞ for the query $b1\_1$, which has the same form as $b1\_5$, except that it is a SUM query.

Global sensitivity w.r.t. standard metric. In the second case, we compute global sensitivity w.r.t. row difference metric. That is, $d(x, x') = 1$ iff there is exactly one sensitive table in databases x and x' such that the respective instances t and t' of that table differ in one row. To make the comparison more fair, we consider an input table sensitive iff the query uses at least one of its attributes that were considered sensitive by the ℓ_p-metric. For SUM, MIN, MAX queries, the effect of adding/removing a row is unbounded, and global sensitivity is ∞, as it covers the worst case. For COUNT queries, we may lose advantage as well if we consider a JOIN of tables, where adding/removing a row in an input table may result in adding/removing an unbounded number of rows in the cross product of input tables, as it happens in $b4$. Row difference metric gives smaller sensitivity in the COUNT-queries $b12\_1$, $b12\_2$, $b16$. In general, if we filter by a sensitive attribute over a single input table, then row difference metric contributes 1 to the COUNT, while defining the distance as ℓ_1-norm of rows allows to split the unit change among several rows, which may result in higher sensitivity.

	SF = 0.1			SF = 0.5			SF = 1.0		
	t_i	t_m	t_s	t_i	t_m	t_s	t_i	t_m	t_s
$b1\_1$	152.8	534.59	763.19	731.11	3.17K	4.08K	1.5K	5.6K	8.01K
$b1\_2$	151.8	559.58	1.04K	1.47K	4.02K	5.86K	1.62K	6.59K	11.92K
$b1\_3$	168.08	590.1	2.05K	862.07	3.24K	10.37K	1.75K	5.87K	19.66K
$b1\_4$	184.24	574.28	2.2K	888.35	2.98K	10.08K	1.69K	5.96K	20.45K
$b1\_5$	149.96	527.38	520.85	744.5	2.69K	2.86K	1.48K	5.46K	5.86K
$b2\_1$	19.68	45.3	144.78	134.21	294.14	1.04K	289.53	563.79	2.25K
$b2\_2$	29.04	49.37	165.62	158.94	273.06	1.28K	288.18	632.38	2.49K
$b3$	111.92	117.41	391.47	544.22	623.87	2.19K	349.06	521.31	1.2K
$b4$	131.52	379.05	778.47	799.9	2.63K	5.16K	1.56K	5.05K	10.69K
$b5$	6.66K	204.08	2.18K	696.38	685.59	3.61K	1.51K	2.2K	9.71K
$b6$	118.31	3.12K	13.21K	687.4	16.09K	67.29K	1.26K	31.73K	123.64K
$b7$	238.74	137.21	713.28	1.19K	861.55	3.9K	2.42K	1.67K	8.34K
$b8$	308.08	117.53	782.37	1.3K	1.73K	5.89K	4.08K	1.45K	8.38K
$b9$	133.34	128.58	3.82K	1.79K	728.07	4.21K	1.59K	1.42K	9.17K
$b10$	131.97	137.03	483.38	882.12	719.65	2.46K	202.05	1.48K	4.88K
$b11$	10.74	10.16	42.12	62.0	62.02	254.81	126.47	128.67	529.29
$b12\_1$	215.13	736.64	1.27K	879.2	3.65K	7.5K	1.95K	7.34K	14.04K
$b12\_2$	148.5	473.72	877.42	846.66	3.26K	6.19K	2.44K	4.8K	10.84K
$b16$	22.14	174.35	303.68	127.95	711.93	1.63K	264.52	1.66K	3.66K
$b17$	111.7	88.31	276.69	486.16	455.85	1.38K	938.62	1.12K	2.96K
$b19$	139.16	296.41	1.42K	737.53	1.47K	6.67K	1.39K	2.86K	13.56K

Table 3. Time benchmarks (ms) for the initial query (t_i), modified query (t_m), and the sensitivity query (t_s). K denotes $\cdot 10^3$

	SF = 0.10			SF = 0.50			SF = 1.00			global sens.	
	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise	non-std.	std.
$b1\_1$	3.79M	50.0	0.01	18.87M	50.0	0.0026	37.72M	50.0	0.0013	∞	∞
$b1\_2$	5.34G	95.89K	0.02	27.35G	99.65K	0.0036	56.57G	104.9K	0.0019	∞	∞
$b1\_3$	5.07G	107.36K	0.02	25.98G	111.18K	0.0043	53.74G	117.34K	0.0022	∞	∞
$b1\_4$	5.27G	114.87K	0.02	27.02G	119.06K	0.0044	55.89G	124.38K	0.0022	∞	∞
$b1\_5$	148.3K	1.0	0.0067	739.56K	1.0	0.0014	1.48M	1.0	0.0007	1	1
$b2\_1$	1.07	100.0	93.46K	1.0	100.0	100.0K	1.0	100.0	100.0K	100	∞
$b2\_2$	999.98	100.0	100.0	1.0K	100.0	100.0	1.0K	100.0	100.0	100	∞
$b3$	3.62K	41.28K	11.4K	3.21K	41.1K	12.8K	0.0	0.0	0.0	∞	∞
$b4$	2.92K	7.0	2.4	14.17K	7.0	0.49	28.07K	7.0	0.25	7	∞
$b5$	5.37M	260.44K	48.53	25.23M	359.6K	14.25	47.6M	484.12K	10.17	∞	∞
$b6$	17.45M (17.13K)	125.0K	7.14	88.13M (86.86M)	127.0K	1.44	181.93M (179.15)	130.0K	0.71	∞	∞
$b7$	22.07M	106.13K	4.81	95.63M	111.24K	1.16	212.11M	115.33K	0.54	∞	∞
$b8$	470.8K	145.15K	308.31	2.74M	172.5K	63.04	3.29M	178.96K	54.4	∞	∞
$b9$	30.32M	40.0K	1.32	137.73M	49.2K	0.36	283.82M	49.2K	0.17	∞	∞
$b10$	100.31K	357.71K	3.57K	149.6K	398.13K	2.66K	0.0	312.54K	∞	∞	∞
$b11$	1.63G	199.98K	0.12	7.73G	199.98K	0.03	15.18G	199.98K	0.01	∞	∞
$b12\_1$	3.12K	3.0	0.96	15.4K	3.0	0.19	30.83K	3.0	0.1	3	1
$b12\_2$	1.29K	3.0	2.33	6.2K	3.0	0.48	12.37K	3.0	0.24	3	1
$b16$	9.95K	4.0	0.4	49.35K	4.0	0.08	98.97K	4.0	0.04	4	1
$b17$	31.54K (31.17K)	16.8K	533	256.24K (250.83K)	17.8K	69.3	531.93K (520.87K)	18.0K	33.9	∞	∞
$b19$	155.25K	651.72K	4.2K	1.1M	813.52K	738.04	1.73M	827.69K	479.67	∞	∞

Table 4. Precision benchmarks for $\epsilon = 1$, $\beta = 0.1$, sigmoid $\alpha = 5$, where $q_i(x)$ is the initial query result, $q_m(x)$ the modified query result (if different from $q_i(x)$), $q_s(x)$ is the sensitivity query result, and %noise $= \frac{|(q_m(x) \pm 10 \cdot q_s(x)) - q_i(x)|}{q_i(x)} \cdot 100$. The last two columns show global sensitivity w.r.t. the same non-standard ℓ_p-induced metric as derivative sensitivity (non-std.) and the standard "row difference" metric (std.). K denotes $\cdot 10^3$, M denotes $\cdot 10^6$, and G denotes $\cdot 10^9$

8 Discussion

Let us summarize the limits and advantages of the framework proposed in this paper. We compare ℓ_p-metric vs row distance metric, and local sensitivity vs global sensitivity. In the following, we mark with + the clear advantages, and with − some caveats.

Applicability.

+ Metrics induced by ℓ_p-norms allow to state different privacy goals, and can be useful in cases where the standard row distance metric is not applicable.
– Computation of derivative sensitivity requires a particular data instance. This is similar to local sensitivity. Since execution of the sensitivity-computing query can be deferred, and the data will anyway be needed at the point where a noisy output is released, we do not treat it as an applicability issue.
– Derivative sensitivity is limited to continuous functions. This is not a problem as far as there exist efficiently computable approximations. We can still cover a wide range of SQL queries.

Complexity.

+ In the first phase of the analysis, the initial query it transformed to sensitivity-computing query. The execution time of this transformation is negligible and does not depend on the data.
– In the second phase of the analysis, when the output is ready to be released, we need to execute the sensitivity-computing query to estimate amount of noise. Compared to the initial query, additional time overhead comes from filtering, as we cannot ignore the rows that have been discarded by the filter.

Amount of noise

+ Changing a numeric attribute of a row in general has smaller effect on the query result than adding/removing an entire row.
+ As global sensitivity always covers the worst-case data instance, it is in general larger than local and derivative sensitivity.
– Compared to global sensitivity w.r.t. standard metric, there are more parameters to be tuned in order to optimize the amount of noise, such as smoothness and sigmoid precision.
– While ℓ_p-norms allow to define a variety of metrics over databases, they are not a superset of standard metrics, and for some privacy goals we can get less noise using standard metric.

Possible improvements. Adding noise *before* filtering is the path towards solving the issue of complexity and noise overhead that comes from filtering over sensitive attributes. While we believe that our framework allows to locate the points where the noise has to be added, this is not the topic of the current paper.

So far, similarly to Blowfish [17], we have assumed that the number of rows in the tables is fixed. It is actually possible to define the derivative sensitivity w.r.t. number of rows, treating it as a real number. Our framework would then cover the row difference metric as well. We defer this research to future work.

9 Conclusion

We have started the study of complete norms to define the quantitative privacy properties of information release mechanisms, and have discovered their high expressivity for different kinds of numeric inputs, as well as the principles of the parallel composition of norms in a manner that allows the sensitivity of the information release mechanism to be found. Our results show how the similarity of local sensitivity and the derivative can be exploited in constructing differentially private mechanisms. The result is also practically significant because of the need to precisely model the privacy requirements of data owner(s), for which the flexibility of specifying the metric over possible inputs is a must.

Our results open up the study of the combinations of metrics over more varied types of input data, including categorical and structured data, or data with consistency constraints. Such study would also look for possibilities to express the constraints through suitable combinations of metrics over components. We note that inputs with constraints [20] or with particular structure (sequences indexed by time points) [7, 10] have been considered in the literature. We hope that it is possible to find suitable complete norms that define the metrics used in the privacy definitions for such data, and thereby express these constructions inside our framework.

Acknowledgements. This research has been funded by Estonian Research Council, grant no. IUT27-1, by ERDF through the Centre of Excellence EXCITE, and by the Air Force Research laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under contract FA8750-16-C-0011. The views expressed are those of the author(s) and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

[1] TPC BENCHMARK^{TM} H, revision 2.17.3. Transaction Processing Performance Council, 2017. http://www.tpc.org/TPC_Documents_Current_Versions/pdf/tpc-h_v2.17.3.pdf.

[2] Myrto Arapinis, Diego Figueira, and Marco Gaboardi. Sensitivity of counting queries. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 120:1–120:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[3] Jordan Awan and Aleksandra Slavkovic. Structure and sensitivity in differential privacy: Comparing k-norm mechanisms. *arXiv preprint arXiv:1801.09236*, 2018.

[4] L.W. Baggett. *Functional Analysis: A Primer*. Chapman & Hall Pure and Applied Mathematics. Taylor & Francis, 1991.

[5] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In Emiliano De Cristofaro and Matthew Wright, editors, *Privacy Enhancing Technologies - 13th International Symposium, PETS 2013, Bloomington, IN, USA, July 10-12, 2013. Proceedings*, volume 7981 of *Lecture Notes in Computer Science*, pages 82–102. Springer, 2013.

[6] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. Geo-indistinguishability: A principled approach to location privacy. In Raja Natarajan, Gautam Barua, and Manas Ranjan Patra, editors, *Distributed Computing and Internet Technology - 11th International Conference, ICDCIT 2015, Bhubaneswar, India, February 5-8, 2015. Proceedings*, volume 8956 of *Lecture Notes in Computer Science*, pages 49–72. Springer, 2015.

[7] Yan Chen, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. Pegasus: Data-adaptive differentially private stream processing. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1375–1388. ACM, 2017.

[8] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In Schulman [27], pages 715–724.

[11] Hamid Ebadi and David Sands. Featherweight PINQ. *Journal of Privacy and Security*, 7(2):159–184, 2016.

[12] Hamid Ebadi, David Sands, and Gerardo Schneider. Differential privacy: Now it's getting personal. In Sriram K. Rajamani and David Walker, editors, *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015*, pages 69–81. ACM, 2015.

[13] Ehab ElSalamouny, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Generalized differential privacy: Regions of priors that admit robust optimal mechanisms. In Franck van Breugel, Elham Kashefi, Catuscia Palamidessi, and Jan Rutten, editors, *Horizons of the Mind. A Tribute to Prakash Panangaden - Essays Dedicated to Prakash Panangaden on the Occasion of His 60th Birthday*, volume 8464 of *Lecture Notes in Computer Science*, pages 292–318. Springer, 2014.

[14] Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce. Linear dependent types for differential privacy. In Roberto Giacobazzi and Radhia Cousot, editors, *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013*, pages 357–370. ACM, 2013.

[15] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In Schulman [27], pages 705–714.

[16] Michael Hay, Ashwin Machanavajjhala, Gerome Miklau, Yan Chen, and Dan Zhang. Principled evaluation of differentially private algorithms using dpbench. In Fatma Özcan, Georgia Koutrika, and Sam Madden, editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 139–154. ACM, 2016.

[17] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish privacy: tuning privacy-utility trade-offs using policies. In Curtis E. Dyreson, Feifei Li, and M. Tamer Özsu, editors, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 1447–1458. ACM, 2014.

[18] Noah M. Johnson, Joseph P. Near, and Dawn Song. Towards practical differential privacy for SQL queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

[19] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing graphs with node differential privacy. In Amit Sahai, editor, *Theory of Cryptography*, pages 457–476, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[20] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1):3:1–3:36, 2014.

[21] Ker-I. Ko and Harvey Friedman. Computational complexity of real functions. *Theoretical Computer Science*, 20:323–352, 1982.

[22] Jaewoo Lee and Chris Clifton. How much is enough? choosing ϵ for differential privacy. In Xuejia Lai, Jianying Zhou, and Hui Li, editors, *Information Security, 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings*, volume 7001 of *Lecture Notes in Computer Science*, pages 325–340. Springer, 2011.

[23] Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In Ugur Çetintemel, Stanley B. Zdonik, Donald Kossmann, and Nesime

Tatbul, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, pages 19–30. ACM, 2009.

[24] Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. Smooth sensitivity and sampling in private data analysis. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 75–84. ACM, 2007.

[25] Catuscia Palamidessi and Marco Stronati. Differential privacy for relational algebra: Improving the sensitivity bounds via constraint systems. In Herbert Wiklicky and Mieke Massink, editors, *Proceedings 10th Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2012, Tallinn, Estonia, 31 March and 1 April 2012.*, volume 85 of *EPTCS*, pages 92–105, 2012.

[26] Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pages 157–168. ACM, 2010.

[27] Leonard J. Schulman, editor. *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. ACM, 2010.

[28] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey F. Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In Semih Salihoglu, Wenchao Zhou, Rada Chirkova, Jun Yang, and Dan Suciu, editors, *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1307–1322. ACM, 2017.

A Sigmoid Precision

Using sigmoids instead of the original filters causes a precision loss, as we use the value $\frac{e^{\alpha x}}{e^{\alpha x}+1}$ instead of 0 or 1. Suppose we have a SUM query using ℓ_1-norm to join row norms and input rows are chosen from a certain distribution that does not change when input size (n, the number of rows) changes, which would be a quite common scenario. Then the relative error from added noise goes to 0 but the relative error from sigmoids is roughly constant as $n \to \infty$. Thus the total error does not go to zero as $n \to \infty$.

To improve on this, we replace sigmoids with precise sigmoids, keeping α constant but increasing the second parameter a as n increases, allowing a tradeoff that (assuming the probability density function of the input is roughly constant near the pivot point of the filter) increases the relative error from added noise $\sqrt{n}$ times and decreases that from sigmoids $\sqrt{n}$ times, making the total error inversely proportional to $\sqrt{n}$, thus going to zero as $n \to \infty$. This also holds if more than one sigmoid is used. If no sigmoids are used then no tradeoff is needed and the total error inversely proportional to n.

We can get similar results for tauoids. If the filtered values are integers then we do not have to use sigmoids and tauoids but instead use the precise functions as described before. Then the only error is from adding noise and it would be inversely proportional to n.

B Evaluation Details

B.1 Database Schema

The TPC-H testset [1] puts forth the following database schema, as given below. The tables are (randomly) filled with a number of rows, generated by a program that accompanies the schema. The number of rows depends on the *scaling factor* SF. The tables, and the numbers of rows in them are the following:

Part: $SF \cdot 200,000$ rows.

column	type
P_PARTKEY	identifier
P_NAME	text
P_MFGR	text
P_BRAND	text
P_TYPE	text
P_SIZE	integer
P_CONTAINER	text
P_RETAILPRICE	decimal
P_COMMENT	text

Customer: $SF \cdot 150,000$ rows.

column	type
C_CUSTKEY	identifier
C_NAME	text
C_ADDRESS	text
C_NATIONKEY	identifier
C_PHONE	text
C_ACCTBAL	decimal
C_MKTSEGMENT	text
C_COMMENT	text

Partsupp: $SF \cdot 800,000$ rows.

column	type
PS_PARTKEY	identifier
PS_SUPPKEY	identifier
PS_AVAILQTY	integer
PS_SUPPLYCOST	decimal
PS_COMMENT	text

Supplier: $SF \cdot 10,000$ rows.

column	type
S_SUPPKEY	identifier
S_NAME	text
S_ADDRESS	text
S_NATIONKEY	identifier
S_PHONE	text
S_ACCTBAL	decimal
S_COMMENT	text

Orders: $SF \cdot 1,500,000$ rows

column	type
O_ORDERKEY	identifier
O_CUSTKEY	identifier
O_ORDERSTATUS	text
O_TOTALPRICE	decimal
O_ORDERDATE	date
O_ORDERPRIORITY	text
O_CLERK	text
O_SHIPPRIORITY	integer
O_COMMENT	text

Lineitem: $SF \cdot 6,000,000$ rows

column	type
L_ORDERKEY	identifier
L_PARTKEY	identifier
L_SUPPKEY	identifier
L_LINENUMBER	integer
L_QUANTITY	decimal
L_EXTENDEDPRICE	decimal
L_DISCOUNT	decimal $\in [0,1]$
L_TAX	decimal
L_RETURNFLAG	text
L_LINESTATUS	text
L_SHIPDATE	date
L_COMMITDATE	date
L_RECEIPTDATE	date
L_SHIPINSTRUCT	text
L_SHIPMODE	text
L_COMMENT	text

Nation: 25 rows

column	type
N_NATIONKEY	identifier
N_NAME	text
N_REGIONKEY	identifier
N_COMMENT	text

Region: 5 rows

column	type
R_REGIONKEY	identifier
R_NAME	text
R_COMMENT	text

B.2 Sensitive Components

In all tables except **Lineitem**, we consider the change that is the scaled sum of changes in all sensitive attributes. All attributes that are not a part of the norm are considered insensitive. We assumed that textual fields as well as the keys (ordinal data) are not sensitive. The columns of type `date` (e.g. $o\_orderdate$) have been converted to a floating-point number, which is the number of months passed from the date 1980-01-01.

– **Part:** $\|p\_size, 0.01 \cdot p\_retailprice\|_1$. The values of $p\_retailprice$ are measured in hundreds, so we consider larger changes (i.e. make such change causing a change of 1 in the output correspond to unit sensitivity).
– **Partsupp:** $\|ps\_availqty, 0.01 \cdot ps\_supplycost\|_1$.
– **Orders:** $\|30 \cdot o\_orderdate, 0.01 \cdot o\_totalprice\|_1$.
– **Customer:** $\|0.01 \cdot c\_acctbal\|_1$.
– **Supplier:** $\|0.01 \cdot s\_acctbal\|_1$.
– **Nation:** no sensitive columns.
– **Region:** no sensitive columns.

In table **Lineitem**, several different norms would make sense and it is up to the data owner to choose the "right" one. We could again add up the sensitive attributes of a row, after suitably scaling them. But we could also think that the three different dates would probably move rather synchronously, and it is the maximum change among them that really matters. Hence we performed the tests with the row norm

$$\|l\_quantity, 0.0001 \cdot l\_extendedprice, 50 \cdot l\_discount,$$
$$30 \cdot \|l\_shipdate, l\_commitdate, l\_receiptdate\|_\infty\|_1 \ .$$

Here the values of $l\_discount$ are very small (all around 0.1), so we aim to protect the change in 0.02 units. On the other hand, $l\_extendedprice$ can be tens of thousands, and we want to capture larger changes for it. The dates are measured in *months*, so we capture a change of one day. which we treat as 1/30 of a month. Alternatively, we may use the number of *days*, and get the same result removing the scaling by 30.0.

B.3 Benchmark Queries

We list the rewritten queries of TCP-H dataset that were used in benchmarking, to give an impression of what we actually feed to the analyzer. The constant 0.142857 in b_17 comes from translating an AVG query to a SUM query, and $7 = 1/0.142857$ is the number of rows to sum, which is public, as the filter does not use sensitive attributes.

```
--b1_1
SELECT SUM(lineitem.l_quantity)
FROM lineitem
WHERE
  lineitem.l_shipdate <= 230.3 - 30
  AND lineitem.l_returnflag = 'R'
  AND lineitem.l_linestatus = 'F'
;
--b1_2
SELECT SUM(lineitem.l_extendedprice)
FROM lineitem
WHERE
  lineitem.l_shipdate <= 230.3 - 30
  AND lineitem.l_returnflag = 'R'
  AND lineitem.l_linestatus = 'F'
;
--b1_3
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount))
FROM lineitem
WHERE
  lineitem.l_shipdate <= 230.3 - 30
  AND lineitem.l_returnflag = 'R'
  AND lineitem.l_linestatus = 'F'
;
--b1_4
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount)
           *(1+lineitem.l_tax))
FROM lineitem
WHERE
  lineitem.l_shipdate <= 230.3 - 30
  AND lineitem.l_returnflag = 'R'
  AND lineitem.l_linestatus = 'F'
;
--b1_5
SELECT COUNT(*)
FROM lineitem
WHERE
  lineitem.l_shipdate <= 230.3 - 30
  AND lineitem.l_returnflag = 'R'
  AND lineitem.l_linestatus = 'F'
;
--b2_1
SELECT MIN(partsupp.ps_supplycost)
FROM partsupp, supplier,
     nation, region, part
WHERE
  part.p_partkey = partsupp.ps_partkey
  AND supplier.s_suppkey = partsupp.ps_suppkey
  AND supplier.s_nationkey = nation.n_nationkey
  AND nation.n_regionkey = region.r_regionkey
  AND region.r_name = 'ASIA'
;
--b2_2
SELECT MAX(partsupp.ps_supplycost)
FROM partsupp, supplier,
     nation, region, part
WHERE
  part.p_partkey = partsupp.ps_partkey
  AND supplier.s_suppkey = partsupp.ps_suppkey
  AND supplier.s_nationkey = nation.n_nationkey
  AND nation.n_regionkey = region.r_regionkey
  AND region.r_name = 'ASIA'
;
--b3
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount))
FROM customer, orders, lineitem
WHERE
  customer.c_mktsegment = 'BUILDING'
  AND customer.c_custkey = orders.o_custkey
  AND lineitem.l_orderkey = orders.o_orderkey
  AND orders.o_orderdate < 190
  AND lineitem.l_shipdate > 190
  AND lineitem.l_orderkey = '162'
  AND orders.o_shippriority = '0'
;
--b4
SELECT COUNT(*)
FROM orders, lineitem
WHERE
  orders.o_orderdate >= 180
  AND orders.o_orderdate < 180 + 3
  AND lineitem.l_orderkey = orders.o_orderkey
  AND lineitem.l_commitdate < lineitem.l_receiptdate
  AND orders.o_orderpriority = '1-URGENT'
;
--b5
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount))
FROM customer, orders,
     lineitem, supplier,
     nation, region
WHERE
  customer.c_custkey = orders.o_custkey
  AND lineitem.l_orderkey = orders.o_orderkey
  AND lineitem.l_suppkey = supplier.s_suppkey
  AND customer.c_nationkey = supplier.s_nationkey
  AND supplier.s_nationkey = nation.n_nationkey
  AND nation.n_regionkey = region.r_regionkey
  AND region.r_name = 'ASIA'
  AND orders.o_orderdate >= 213.3
  AND orders.o_orderdate < 213.3 + 12
  AND nation.n_name = 'JAPAN'
;
--b6
SELECT SUM(lineitem.l_extendedprice
           *lineitem.l_discount)
FROM lineitem
WHERE
  lineitem.l_shipdate >= 170.5
  AND lineitem.l_shipdate < 170.5 + 12
  AND lineitem.l_discount BETWEEN 0.09 - 0.01
                          AND 0.09 + 0.01
  AND lineitem.l_quantity < 24
;
--b7
SELECT SUM(lineitem.l_extendedprice
           *(1 - lineitem.l_discount))
```

```
FROM supplier, lineitem, orders, customer,
     nation as n1,
     nation as n2
WHERE
  supplier.s_suppkey = lineitem.l_suppkey
  AND orders.o_orderkey = lineitem.l_orderkey
  AND customer.c_custkey = orders.o_custkey
  AND supplier.s_nationkey = n1.n_nationkey
  AND customer.c_nationkey = n2.n_nationkey
  AND (
    (n1.n_name = 'JAPAN' and n2.n_name = 'INDONESIA')
    OR
    (n1.n_name = 'INDONESIA' and n2.n_name = 'JAPAN')
  )
  AND lineitem.l_shipdate between 182.6 and 207
;
--b8
SELECT SUM(lineitem.l_extendedprice
        *(1 - lineitem.l_discount))
FROM
  part, supplier, lineitem,
  orders, customer,
  nation AS n1, nation AS n2, region
WHERE
  part.p_partkey = lineitem.l_partkey
  AND supplier.s_suppkey = lineitem.l_suppkey
  AND lineitem.l_orderkey = orders.o_orderkey
  AND orders.o_custkey = customer.c_custkey
  AND customer.c_nationkey = n1.n_nationkey
  AND n1.n_regionkey = region.r_regionkey
  AND region.r_name = 'ASIA'
  AND supplier.s_nationkey = n2.n_nationkey
  AND orders.o_orderdate >= 5478
  AND orders.o_orderdate <= 6210
  AND part.p_type = 'MEDIUM BRUSHED COPPER'
  AND n2.n_name = 'JAPAN'
;
--b9
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount)
       - partsupp.ps_supplycost*lineitem.l_quantity)
FROM
    part,     supplier,
    lineitem, partsupp,
    orders,   nation
WHERE
  supplier.s_suppkey = lineitem.l_suppkey
  AND partsupp.ps_suppkey = lineitem.l_suppkey
  AND partsupp.ps_partkey = lineitem.l_partkey
  AND part.p_partkey = lineitem.l_partkey
  AND orders.o_orderkey = lineitem.l_orderkey
  AND supplier.s_nationkey = nation.n_nationkey
  AND part.p_name LIKE '%violet%'
  AND nation.n_name = 'UNITED KINGDOM'
;
--b10
SELECT SUM(lineitem.l_extendedprice
           *(1 - lineitem.l_discount))
FROM
  customer, orders,
  lineitem, nation
WHERE
  customer.c_custkey = orders.o_custkey
  AND lineitem.l_orderkey = orders.o_orderkey
  AND orders.o_orderdate >= 183.3
  AND orders.o_orderdate <  183.3 + 3
  AND lineitem.l_returnflag = 'R'
  AND customer.c_nationkey = nation.n_nationkey
  AND customer.c_custkey = '64'
  AND nation.n_name = 'CANADA'
;
--b11
SELECT SUM(partsupp.ps_supplycost
           * partsupp.ps_availqty * 0.2)
FROM partsupp, supplier, nation
WHERE
  partsupp.ps_suppkey = supplier.s_suppkey
  AND supplier.s_nationkey = nation.n_nationkey
  AND nation.n_name = 'JAPAN'
;
--b12_1
SELECT COUNT(*)
FROM orders, lineitem
WHERE
  orders.o_orderkey = lineitem.l_orderkey
  AND (orders.o_orderpriority <> '1-URGENT'
       OR orders.o_orderpriority <> '2-HIGH')
  AND lineitem.l_shipmode in ('TRUCK', 'SHIP')
  AND lineitem.l_commitdate < lineitem.l_receiptdate
  AND lineitem.l_shipdate < lineitem.l_commitdate
  AND lineitem.l_receiptdate >= 183.3
  AND lineitem.l_receiptdate < 183.3 + 12
;
--b12_2
SELECT COUNT(*)
FROM orders, lineitem
WHERE
  orders.o_orderkey = lineitem.l_orderkey
  AND (orders.o_orderpriority = '1-URGENT'
       OR orders.o_orderpriority = '2-HIGH')
  AND lineitem.l_shipmode in ('TRUCK', 'SHIP')
  AND lineitem.l_commitdate < lineitem.l_receiptdate
  AND lineitem.l_shipdate < lineitem.l_commitdate
  AND lineitem.l_receiptdate >= 183.3
  AND lineitem.l_receiptdate < 183.3 + 12
;
--b16
SELECT COUNT(partsupp.ps_suppkey)
FROM partsupp, part, supplier
WHERE
  part.p_partkey = partsupp.ps_partkey
  AND partsupp.ps_suppkey = supplier.s_suppkey
  AND part.p_brand <> 'Brand#34'
  AND NOT (part.p_type LIKE '%COPPER%')
  AND part.p_size in (5, 10, 15, 20, 25, 30, 35, 40)
  AND NOT (supplier.s_comment LIKE
                            '%Customer%Complaints%')
  AND part.p_brand = 'Brand#14'
  AND part.p_type  = 'LARGE ANODIZED TIN'
;
```

```
--b17
SELECT SUM(lineitem.l_extendedprice * 0.142857)
FROM lineitem, part
WHERE
  part.p_partkey = lineitem.l_partkey
  AND part.p_brand = 'Brand#34'
  AND part.p_container = 'JUMBO PKG'
  AND lineitem.l_quantity < 0.2 * 32
;
--b19
SELECT SUM(lineitem.l_extendedprice
           *(1-lineitem.l_discount))
FROM lineitem, part
WHERE
  part.p_partkey = lineitem.l_partkey
  AND lineitem.l_shipmode IN ('AIR', 'AIR REG')
  AND lineitem.l_shipinstruct = 'DELIVER IN PERSON'
  AND part.p_size >= 1
  AND
  ((
    part.p_brand = 'Brand#34'
    AND part.p_container IN ('SM CASE', 'SM BOX',
                             'SM PACK', 'SM PKG')
    AND lineitem.l_quantity >= 35
    AND lineitem.l_quantity <= 35 + 10
    AND part.p_size <= 5
    )
    OR
    (
    part.p_brand = 'Brand#22'
    AND part.p_container IN ('MED BAG', 'MED BOX',
                             'MED PKG', 'MED PACK')
    AND lineitem.l_quantity >= 12
    AND lineitem.l_quantity <= 12 + 10
    AND part.p_size  <= 10
    )
    OR
    (
    part.p_brand = 'Brand#14'
    AND part.p_container IN ('LG CASE', 'LG BOX',
                             'LG PACK', 'LG PKG')
    AND lineitem.l_quantity >= 90
    AND lineitem.l_quantity <= 90 + 10
    AND part.p_size  <= 15
));
```

B.4 Integer vs Float Type Filtering

Since the *date* datatype of SQL is measured within day precision, it makes sense to treat is as an integer. However, we could as well represent it as a floating-point number. This allows us to use sigmoids, as discussed in Sec. 5.1.2. For sigmoids, we have to choose precision in such a way that the noise would be smaller. Since precision itself cannot depend on the data, we have empirically evaluated appropriate precision level on an independently generated TCP-H instance with scale factor SF=0.05. As described in Sec. A, the precision has to be increased proportionally with $\sqrt{n}$, where n is the number of analyzed rows. Hence, the sigmoid precisions for the cases of SF 0.1, 0.5, 1.0 had to be multiplied with $\sqrt{2}$, $\sqrt{10}$ and $\sqrt{20}$ respectively.

While Table 3 and Table 4 use integer approximation for date filtering, the tables Table 5 and Table 6 show the results for sigmoid approach. The results have been computed for different β and α values, where $\beta \approx 0$ means that the sensitivity could be computed for an arbitrarily small β, and the third column shows the base α that has been computed for SF=0.05. The computing time for a modified query is much higher for floating points, since the SQL engine now needs to compute exponentiation for each row and each private filter, so for the most complicated queries we present the results up to SF=0.5. We see that, except the queries $b12\_1$ and $b12\_2$, the error gets smaller compared to integer datatype approach. The problem of $b12\_1$ and $b12\_2$ seems to be that the sigmoid precision that we found for SF=0.05 is not the best for SF=0.1, which indeed may happen as the final result depends not only on the number of rows, but also on the actual data, so even though the sensitivity of these queries is smaller, they suffer from precision error. The disadvantages of sigmoid approach are that it takes more time to execute the modified query, and that exponentiations tend to cause overflow errors in PSQL engine when the exponents get large. The time overheads are more significant for the cases with many private filters, where sigmoid error gets larger due to multiplication, so it seems more reasonable to use integer datatype there.

B.5 Examples of Analyzer Output

We give some examples of shorter queries that have been output by the analyzer. We present the modified query and the sensitivity query, demonstrating where computation overhead may come from.

Query $b1\_1$

Modified query.

```
SELECT sum((lineitem.l_quantity * (exp((0.1 * (200.3 +
((-1.0) * lineitem.l_shipdate)))) / (exp((0.1 * (200.3 +
((-1.0) * lineitem.l_shipdate)))) + 1.0))))
FROM lineitem
WHERE (lineitem.l_linestatus = 'F')
  AND (lineitem.l_returnflag = 'R');
```

Sensitivity query.

	SF = 0.1			SF = 0.5			SF = 1.0		
	t_i	t_m	t_s	t_i	t_m	t_s	t_i	t_m	t_s
$b1\_1$	144.36	11.43K	1.74K	761.36	157.38K	9.49K	1.47K	535.97K	18.51K
$b1\_2$	141.57	11.43K	1.74K	742.35	163.8K	9.57K	1.46K	518.45K	19.08K
$b1\_3$	154.66	11.75K	1.89K	886.02	157.73K	9.93K	1.67K	538.28K	21.61K
$b1\_4$	165.84	11.64K	2.81K	851.84	154.48K	15.4K	1.97K	558.75K	29.17K
$b1\_5$	149.43	6.6K	1.06K	769.5	65.65K	6.16K	1.43K	189.94K	11.0K
$b2\_1$	19.08	36.87	139.14	141.1	268.5	1.28K	269.94	524.61	2.24K
$b2\_2$	19.11	42.29	135.36	146.93	278.97	1.07K	265.5	724.23	2.07K
$b3$	96.29	110.48	376.65	567.6	645.44	2.15K	288.72	451.16	1.03K
$b4$	127.72	59.0K	2.37K	653.65	1.8M	12.67K	–	–	–
$b5$	6.25K	242.1	2.13K	715.05	877.75	3.37K	1.32K	2.49K	7.96K
$b6$	115.36	284.1K	9.37K	586.04	6.34M	46.5K	–	–	–
$b7$	218.87	329.15	623.63	1.16K	2.93K	3.74K	2.45K	9.05K	7.13K
$b8$	238.62	113.11	637.68	1.15K	719.8	3.9K	3.43K	1.22K	6.54K
$b9$	127.56	121.25	3.23K	746.92	719.1	4.22K	1.29K	1.3K	7.83K
$b10$	129.26	134.37	445.86	676.09	790.71	2.4K	205.47	1.45K	4.92K
$b11$	10.49	9.87	38.92	62.34	61.11	244.71	125.13	126.73	524.31
$b12\_1$	157.39	110.13K	4.62K	849.04	2.49M	26.3K	–	–	–
$b12\_2$	146.71	44.21K	2.12K	727.03	961.28K	10.6K	–	–	–
$b16$	21.45	130.13	216.23	155.18	735.92	1.65K	237.25	1.42K	3.38K
$b17$	86.52	83.86	269.3	481.85	463.56	1.43K	872.76	861.25	2.55K
$b19$	130.1	264.35	1.21K	718.78	1.48K	7.74K	1.3K	2.72K	12.73K

Table 5. Time benchmarks (ms) for the initial query (t_i), modified query (t_m), and the sensitivity query (t_s). K denotes $\cdot 10^3$, and M denotes $\cdot 10^6$

```
SELECT max(abs(sdsg)) FROM (
  SELECT sum(abs(greatest(abs((exp((0.1 * (200.3
+ ((-1.0) * lineitem.l_shipdate)))) / (exp((0.1 *
(200.3 + ((-1.0) * lineitem.l_shipdate)))) + 1.0))),
abs(case when ((((0.1 * exp((0.1 * (200.3 + ((-1.0)
* lineitem.l_shipdate))))) / ((exp((0.1 * (200.3 +
((-1.0) * lineitem.l_shipdate)))) + 1.0) ^ 2.0)) *
0.03) = 0.0) then 0.0 else ((((0.1 * exp((0.1 * (200.3
+ ((-1.0) * lineitem.l_shipdate))))) / ((exp((0.1 *
(200.3 + ((-1.0) * lineitem.l_shipdate)))) + 1.0) ^
2.0)) * 0.03) * case when (abs(lineitem.l_quantity) >=
10.0) then abs(lineitem.l_quantity) else (exp(((0.1 *
abs(lineitem.l_quantity)) - 1.0)) / 0.1) end) end)))) AS
sdsg
  FROM lineitem, lineitem_sensRows
  WHERE ((lineitem.l_linestatus = 'F')
     AND(lineitem.l_returnflag = 'R')
     AND lineitem_sensRows.ID = lineitem.ID)
     AND lineitem_sensRows.sensitive
GROUP BY lineitem_sensRows.ID) AS sub;
```

Query $b1\_5$

Modified query.

```
SELECT sum(abs((exp((0.1 * (200.3 + ((-1.0) *
lineitem.l_shipdate)))) / (exp((0.1 * (200.3 + ((-1.0) *
lineitem.l_shipdate)))) + 1.0))))
FROM lineitem
WHERE (lineitem.l_linestatus = 'F')
  AND (lineitem.l_returnflag = 'R');
```

Sensitivity query.

```
SELECT max(sdsg) FROM (
  SELECT sum(abs((((0.1 * exp((0.1 * (200.3 + ((-1.0) *
lineitem.l_shipdate))))) / ((exp((0.1 * (200.3 + ((-1.0)
* lineitem.l_shipdate)))) + 1.0)^2.0)) * 0.03))) AS sdsg
  FROM lineitem, lineitem_sensRows
  WHERE ((lineitem.l_linestatus = 'F')
    AND (lineitem.l_returnflag = 'R')
    AND lineitem_sensRows.ID = lineitem.ID)
    AND lineitem_sensRows.sensitive
  GROUP BY lineitem_sensRows.ID) AS sub;
```

Query $b16$

Modified query.

```
SELECT sum(abs((((((((2.0 / (exp(((-0.1) * (part.p_size
- 10.0))) + exp((0.1 * (part.p_size - 10.0))))) + (2.0
/ (exp(((-0.1) * (part.p_size - 15.0))) + exp((0.1
* (part.p_size - 15.0)))))) + (2.0 / (exp(((-0.1) *
(part.p_size - 20.0))) + exp((0.1 * (part.p_size -
20.0)))))) + (2.0 / (exp(((-0.1) * (part.p_size -
25.0))) + exp((0.1 * (part.p_size - 25.0)))))) + (2.0
/ (exp(((-0.1) * (part.p_size - 30.0))) + exp((0.1
* (part.p_size - 30.0)))))) + (2.0 / (exp(((-0.1) *
(part.p_size - 35.0))) + exp((0.1 * (part.p_size -
35.0)))))) + (2.0 / (exp(((-0.1) * (part.p_size -
40.0))) + exp((0.1 * (part.p_size - 40.0)))))) + (2.0
/ (exp(((-0.1) * (part.p_size - 5.0))) + exp((0.1 *
(part.p_size - 5.0))))))))
```

			SF = 0.1			SF = 0.5			SF = 1.0		
	β	sigmoid prec. α	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise	$q_i(x)$ ($q_m(x)$)	$q_s(x)$	%noise
$b1\_1$	≈ 0	0.1	3.79M	1.8	0.0002	18.87M	4.03	0.0001	37.72M	5.7	7.6e-05
$b1\_2$	≈ 0	0.1	5.34G	10.0K	0.0009	27.35G	10.0K	0.0002	56.57G	11.18K	9.9e-05
$b1\_3$	≈ 0	0.1	5.07G	19.0K	0.0019	25.98G	19.0K	0.0004	53.74G	21.24K	0.0002
$b1\_4$	0.1	0.1	5.27G	12.11K	0.0023	27.02G	12.11K	0.0004	55.89G	13.91K	0.0002
$b1\_5$	≈ 0	0.05	148.3K	0.02	6e-05	739.56K	0.04	2.7e-05	1.48M	0.06	1.9e-05
$b2\_1$	0.1	0.1	1.07	100.0	93.46K	1.0	100.0	100.0K	1.0	100.0	100.0K
$b2\_2$	0.1	0.1	999.98	100.0	100.0	1.0K	100.0	100.0	1.0K	100.0	100.0
$b3$	≈ 0	0.01	3.62K (2.1K)	19.0K	2.58K	3.21K (2.17K)	19.0K	2.93K	0.0	0.0	0.0
$b4$	≈ 0	0.5	2.92K (2.96K)	1.24	1.73	14.17K	3.32	0.76	–	–	–
$b5$	0.1	0.01	5.37M (4.82M)	11.21K	8.1	25.23M (24.74M)	11.21K	1.5	47.6M (46.88M)	11.21K	1.27
$b6$	≈ 0	0.4	17.45M (17.91M)	105.0K	5.67	88.13M (89.71M)	105.0K	1.81	–	–	–
$b7$	≈ 0	0.1	22.07M (21.99M)	19.0K	0.06	95.63M (95.44M)	19.0K	0.1	212.11M (212.02M)	21.24K	0.0078
$b8$	0.1	0.1	470.8K (470.86K)	11.21K	23.83	2.74M (2.75M)	13.64K	5.31	3.29M (3.3M)	20.01K	6.44 6.44
$b9$	0.1	0.1	30.32M	40.0K	1.32	137.73M	49.2K	0.36	283.82M	49.2K	0.17
$b10$	0.1	0.1	100.31K (100.06K)	12.65K	125.84	149.6K (143.52K)	31.48K	206.34	0.0	34.94K	∞
$b11$	0.1	0.1	1.63G	199.98K	0.12	7.73G	199.98K	0.03	15.18G	199.98K	0.01
$b12\_1$	≈ 0	0.5	3.12K (3.38K)	0.53	8.43	15.4K (16.52K)	1.19	7.32	–	–	–
$b12\_2$	≈ 0	0.5	1.29K (1.4K)	0.53	8.88	6.2K (6.67K)	1.19	7.61	–	–	–
$b16$	0.1	0.1	9.95K	4.0	0.4	49.35K	4.0	0.08	98.97K	4.0	0.04
$b17$	≈ 0	0.5	31.54K (40.57K)	2.53K	68.66	256.24K (253.19K)	5.65K	9.83	531.93K (519.68K)	7.99K	5.21
$b19$	0.1	0.1	155.25K	651.72K	4.2K	1.1M	813.52K	738.04	1.73M	827.69K	479.67

Table 6. Precision benchmarks for $\epsilon = 1$, where $q_i(x)$ is the initial query result, $q_m(x)$ the modified query result (if different from $q_i(x)$), $q_s(x)$ is the sensitivity query result, and %noise $= \frac{|(q_m(x) \pm 10 \cdot q_s(x)) - q_i(x)|}{q_i(x)} \cdot 100$. K denotes $\cdot 10^3$, M denotes $\cdot 10^6$, and G denotes $\cdot 10^9$

```
FROM part, partsupp, supplier
WHERE NOT((supplier.s_comment LIKE
             '%Customer%Complaints%'))
  AND NOT((part.p_type LIKE '%COPPER%'))
  AND NOT((part.p_brand = 'Brand#34'))
  AND (part.p_partkey = partsupp.ps_partkey)
  AND (partsupp.ps_suppkey = supplier.s_suppkey);
```

Sensitivity query.

```
SELECT max(sdsg) FROM (
  SELECT sum(abs(((((((((0.1 * (2.0 / (exp(((-0.1)
* (part.p_size - 10.0))) + exp((0.1 * (part.p_size
- 10.0)))))) * 8.0) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 15.0))) + exp((0.1 * (part.p_size -
15.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 20.0))) + exp((0.1 * (part.p_size -
20.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 25.0))) + exp((0.1 * (part.p_size -
25.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 30.0))) + exp((0.1 * (part.p_size -
30.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 35.0))) + exp((0.1 * (part.p_size -
35.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 40.0))) + exp((0.1 * (part.p_size -
40.0)))))) * 8.0)) + ((0.1 * (2.0 / (exp(((-0.1) *
(part.p_size - 5.0))) + exp((0.1 * (part.p_size -
5.0)))))) * 8.0)))) AS sdsg
  FROM part, partsupp, supplier, part_sensRows
  WHERE NOT((supplier.s_comment LIKE
             '%Customer%Complaints%'))
    AND NOT((part.p_type LIKE '%COPPER%'))
    AND NOT((part.p_brand = 'Brand#34'))
    AND (part.p_partkey = partsupp.ps_partkey)
    AND (partsupp.ps_suppkey = supplier.s_suppkey)
    AND part_sensRows.ID = part.ID
    AND part_sensRows.sensitive
  GROUP BY part_sensRows.ID) AS sub;
```

C Postponed Proofs

C.1 Useful Facts and Lemmas

Fact 10. *For all $(x_1,\ldots,x_m)\in\mathbb{R}^m$, $p\geq q$, we have*

$$\|x_1,\ldots,x_m\|_p \leq \|x_1,\ldots,x_m\|_q \leq m^{1/q-1/p}\cdot\|x_1,\ldots,x_m\|_p \enspace .$$

Fact 11. *For all $x\in\mathbb{R}^+$, $\|x\|_p = x$, and $\|x\|_\infty = x$.*

Fact 12 (a part of Lemma 2.3 of [24])**.** *Let $f: X\to\mathbb{R}$. For $\beta>0$, a β-smooth upper bound on f is*

$$g_{f,\beta} = \max_{x'\in X}(f(x)\cdot e^{-\beta\cdot d(x,x')}) \enspace .$$

Lemma 13. *Let $\mathbf{x} = (x_1,\ldots,x_k)\in\mathbb{R}^k$, $\mathbf{y} = (y_1,\ldots,y_n)\in\mathbb{R}^n$, $\mathbf{z} = (z_1,\ldots,z_m)\in\mathbb{R}^m$. If $p\geq q\geq 1$, then*
1. $\left\|\|\mathbf{x}\|_q,\|\mathbf{y}\|_q,z_1,\ldots,z_m\right\|_p \leq \left\|\|\mathbf{x}|\mathbf{y}\|_q,z_1,\ldots,z_m\right\|_p$;
2. $\left\|\|\mathbf{x}\|_p,\|\mathbf{y}\|_p,z_1,\ldots,z_m\right\|_q \geq \left\|\|\mathbf{x}|\mathbf{y}\|_p,z_1,\ldots,z_m\right\|_q$;
where $\mathbf{x}|\mathbf{y}$ denotes concatenation. If $p=q$, then the inequalities become equalities.

Proof. Since $p,q\geq 1$, we may raise both sides of equations to the powers p or q. The main inequalities that we use in the proof are $a^n+b^n\leq(a+b)^n$ for $n\geq 1$, and $a^n+b^n\geq(a+b)^n$ for $n\leq 1$.

$$\begin{aligned}
&\left\|\|\mathbf{x}\|_q,\|\mathbf{y}\|_q,z_1,\ldots,z_m\right\|_p^p\\
&= \left(\sum_{i=1}^k x_i^q\right)^{\frac{p}{q}} + \left(\sum_{i=1}^n y_i^q\right)^{\frac{p}{q}} + \sum_{i=1}^m z_i^p\\
&\leq \left(\sum_{i=1}^k x_i^q + \sum_{i=1}^n y_i^q\right)^{\frac{p}{q}} + \sum_{i=1}^m z_i^p\\
&= \|\mathbf{x}|\mathbf{y}\|_q^p + \sum_{i=1}^m z_i^p = \left\|\|\mathbf{x}|\mathbf{y}\|_q,z_1,\ldots,z_m\right\|_p^p \enspace .
\end{aligned}$$

$$\begin{aligned}
&\left\|\|\mathbf{x}\|_p,\|\mathbf{y}\|_p,z_1,\ldots,z_m\right\|_q^q\\
&= \left(\sum_{i=1}^k x_i^p\right)^{\frac{q}{p}} + \left(\sum_{i=1}^n y_i^p\right)^{\frac{q}{p}} + \sum_{i=1}^m z_i^q\\
&\geq \left(\sum_{i=1}^k x_i^p + \sum_{i=1}^n y_i^p\right)^{\frac{q}{p}} + \sum_{i=1}^m z_i^q\\
&= \|\mathbf{x}|\mathbf{y}\|_p^q + \sum_{i=1}^m z_i^q = \left\|\|\mathbf{x}|\mathbf{y}\|_p,z_1,\ldots,z_m\right\|_q^q \enspace .
\end{aligned}$$

If $p=q$, then all inequalities in these derivations are equalities. ◻

Lemma 14. *For all $x\in\mathbb{R}$, $(\alpha_1,\ldots,\alpha_k)\in\mathbb{R}^k$, $(y_1,\ldots,y_m)\in\mathbb{R}^m$: $\|\alpha_1 x,\ldots,\alpha_k x,y_1,\ldots,y_m\|_p = \left\|\sqrt[p]{\sum_{i=1}^k\alpha_i^p}x,y_1,\ldots,y_m\right\|_p$.*

Proof. Since an ℓ_p-norm is defined for $p\geq 1$, we may raise both sides of equation to the power p. We use the definition of ℓ_p-norm and rewrite the term.

$$\begin{aligned}
&\|\alpha_1 x,\ldots,\alpha_k x,y_1,\ldots,y_m\|_p^p\\
&= \sum_{i=1}^k(\alpha_i x)^p + \sum_{i=1}^m y_i^p\\
&= \left(\sum_{i=1}^k \alpha_i^p\right)x^p + \sum_{i=1}^m y_i^p\\
&= \left\|\sqrt[p]{\sum_{i=1}^k\alpha_i^p}x,y_1,\ldots,y_m\right\|_p^p \enspace .
\end{aligned}$$

◻

Putting $\alpha_i = 1$ for all $i\in[n]$, we get the following corollary.

Corollary 15. *We have*

$$\left\|\overbrace{x,\ldots,x}^{k},y_1,\ldots,y_m\right\|_p = \left\|\sqrt[p]{k}\cdot x,y_1,\ldots,y_m\right\|_p \enspace .$$

Lemma 16. *Let X_i for $i\in\{1,\ldots,n\}$ be Banach spaces, $f_i: X_i\to\mathbb{R}$. Let $x=(x_1,\ldots,x_n)$, and let $f(x_1,\ldots,x_n) = \|f_1(x_1),\ldots,f_n(x_n)\|_p$. Then $\frac{\partial f}{\partial x_i}(x) = \frac{\partial f_i}{\partial x_i}(x_i)\cdot\left(\frac{f_i(x_i)}{f(x)}\right)^{p-1} \leq \frac{\partial f_i}{\partial x_i}(x_i)$.*

Proof. Let $y_i = f_i(x_i)$ and $y=(y_1,\ldots,y_n)$. We have

$$\frac{\partial f}{\partial x_i}(x) = \frac{\partial f}{\partial y_i}(y)\cdot\frac{\partial f_i}{\partial x_i}(x_i) = \left(\frac{y_i}{\|y\|_p}\right)^{p-1}\cdot\frac{\partial f_i}{\partial x_i}(x_i) \enspace .$$

Since $\frac{y_i}{\|y\|_p} = \frac{f_i(x_i)}{f(x)} = \frac{f_i(x_i)}{\|(f_i(x_i))_{i=1}^n\|_p}$, we have $\frac{f_i(x_i)}{f(x)}\leq 1$, and hence also $\left(\frac{f_i(x_i)}{f(x)}\right)^{p-1}\leq 1$, getting $\frac{\partial f}{\partial x_i}(x)\leq\frac{\partial f_i}{\partial x_i}(x_i)$. ◻

Lemma 17. *Let X_i for $i\in\{1,\ldots,n\}$ be Banach spaces. Let $x=(x_1,\ldots,x_n)$, and let $f(x) = \|f_1(x),\ldots,f_n(x)\|_p$. Then $\frac{\partial f}{\partial x_i}(x) = \sum_{j=1}^n\left(\frac{f_j(x)}{f(x)}\right)^{p-1}\cdot\frac{\partial f_j}{\partial x_i}(x)$. This can be upper bounded as:*
1. $\sum_{j=1}^n\frac{\partial f_j}{\partial x_i}(x)$;
2. $\max_{j=1}^n\frac{f(x)}{f_j(x)}\cdot\frac{\partial f_j}{\partial x_i}(x)$.

Proof. Let $y_j = f_j(x)$, $z = \sum_{j=1}^n y_j^p$. We have

$$\begin{aligned}\frac{\partial f}{\partial x_i}(x) &= \frac{\partial f}{\partial z}(z) \cdot \sum_{j=1}^n \left(\frac{\partial f_j(x)^p}{\partial y_j}(y_j) \cdot \frac{\partial f_j}{\partial x_i}(x) \right) \\ &= \sum_{j=1}^n \left(\frac{y_j}{\|y\|_p} \right)^{p-1} \cdot \frac{\partial f_j}{\partial x_i}(x) \\ &= \sum_{j=1}^n \left(\frac{f_j(x)}{f(x)} \right)^{p-1} \cdot \frac{\partial f_j}{\partial x_i}(x) \ .\end{aligned}$$

As in the proof of Lemma 16, $\left(\frac{y_j}{\|y\|_p}\right)^{p-1} = \left(\frac{f_j(x)}{f(x)}\right)^{p-1} \leq 1$. We get $\frac{\partial f}{\partial x_i}(x) \leq \sum_{j=1}^n \frac{\partial f_j}{\partial x_i}(x)$. We can proceed with the inequality in another way.

$$\begin{aligned}&\textstyle\sum_{j=1}^n \left(\frac{f_j(x)}{f(x)}\right)^{p-1} \cdot \frac{\partial f_j}{\partial x_i}(x) \\ &= \frac{\sum_{j=1}^n f_j(x)^{p-1} \cdot \frac{\partial f_j}{\partial x_i}(x)}{f(x)^{p-1}} \\ &= \frac{\sum_{j=1}^n f_j(x)^p \cdot \frac{\partial f_j}{\partial x_i}(x) \cdot \frac{1}{f_j(x)}}{f(x)^{p-1}} \\ &\leq \max_{j=1}^n \left(\frac{1}{f_j(x)} \cdot \frac{\partial f_j}{\partial x_i}(x) \right) \cdot \frac{\sum_{j=1}^n f_j(x)^p}{f(x)^{p-1}} \\ &= \max_{j=1}^n \left(\frac{1}{f_j(x)} \cdot \frac{\partial f_j}{\partial x_i}(x) \right) \cdot \frac{f(x)^p}{f(x)^{p-1}} \\ &= \max_{j=1}^n \left(\frac{f(x)}{f_j(x)} \cdot \frac{\partial f_j}{\partial x_i}(x) \right) \ . \quad \square\end{aligned}$$

C.2 DP from Cauchy Noise

Proof. Let d_{dp} be defined as in Sec. 3.1. The construction of differentially private mechanisms based on adding noise distributed according to a generalized Cauchy distribution, with the magnitude depending on the smooth upper bounds on the derivative sensitivity of the original query, was given in Theorem 3. Let $\eta \sim GenCauchy(\gamma)$. The generalized Cauchy distribution is relatively stable under shifts and stretchings, satisfying the following inequalities for all $a_1, a_2, c_1, c_2 \in \mathbb{R}$ [24]:

$$d_{\mathrm{dp}}(a_1 + c_1 \cdot \eta, a_2 + c_1 \cdot \eta) \leq (\gamma+1) \cdot \left| \frac{a_2 - a_1}{c_1} \right|$$
$$d_{\mathrm{dp}}(c_1 \cdot \eta, c_2 \cdot \eta) \leq (\gamma+1) \cdot \left| \ln \frac{c_2}{c_1} \right| \ .$$

The combination of these two inequalities gives

$$d_{\mathrm{dp}}(a_1 + c_1 \cdot \eta, a_2 + c_2 \cdot \eta) \leq \\ \leq (\gamma+1) \cdot \left(\frac{|a_2 - a_1|}{\max\{|c_1|, |c_2|\}} + \left| \ln \frac{c_2}{c_1} \right| \right) \ . \quad (5)$$

Let $\mathbf{x}, \mathbf{x}' \in X$. Denote $L = \|\mathbf{x}' - \mathbf{x}\|$. We have to show that $d_{\mathrm{dp}}(g(\mathbf{x}'), g(\mathbf{x})) \leq \epsilon L = (\gamma+1)(b+\beta)L$.

Let $n \in \mathbb{N}$ be arbitrary. Let $\mathbf{v}_0 = \mathbf{x}$, $\mathbf{v}_n = \mathbf{x}'$, and $\mathbf{v}_i = \frac{n-i}{n} \cdot \mathbf{x} + \frac{i}{n} \cdot \mathbf{x}'$, i.e. the values $\mathbf{v}_0, \dots, \mathbf{v}_n$ are evenly distributed in the segment connecting $\mathbf{x}$ to $\mathbf{x}'$. These values are in X because X is convex. Theorem 2 implies that there exist $\mathbf{t}_1, \dots, \mathbf{t}_n$ with $\mathbf{t}_i$ in the segment connecting $\mathbf{v}_{i-1}$ to $\mathbf{v}_i$, satisfying

$$\begin{aligned}|f(\mathbf{v}_i) - f(\mathbf{v}_{i-1})| &\leq \|df_{\mathbf{t}_i}\| \cdot \|\mathbf{v}_i - \mathbf{v}_{i-1}\| \\ &\leq c(\mathbf{t}_i) \cdot \frac{L}{n} \leq e^{\beta L/n} \cdot c(\mathbf{v}_{i-1}) \cdot \frac{L}{n}\end{aligned}$$

for all $i \in \{1, \dots, n\}$. Here the last inequality follows from the β-smoothness of c. We can use these claims together with the triangle inequality and obtain

$$\begin{aligned}&d_{\mathrm{dp}}(g(\mathbf{x}), g(\mathbf{x}')) \leq \textstyle\sum_{i=1}^n d_{\mathrm{dp}}(g(\mathbf{v}_{i-1}), g(\mathbf{v}_i)) \\ &= \sum_{i=1}^n d_{\mathrm{dp}}(f(\mathbf{v}_{i-1}) + \frac{c(\mathbf{v}_{i-1})}{b} \cdot \eta, f(\mathbf{v}_i) + \frac{c(\mathbf{v}_i)}{b} \cdot \eta) \\ &\leq (\gamma+1) \sum_{i=1}^n \left(b \cdot \frac{|f(\mathbf{v}_i) - f(\mathbf{v}_{i-1})|}{|c(\mathbf{v}_{i-1})|} + \frac{\beta L}{n} \right) \\ &\leq (\gamma+1) \sum_{i=1}^n \left(b \cdot e^{\beta L/n} \cdot \frac{L}{n} + \frac{\beta L}{n} \right) \\ &= (\gamma+1)(be^{\beta L/n} + \beta)L \ .\end{aligned}$$

This inequality holds for any $n \in \mathbb{N}$. If $n \to \infty$ then $e^{\beta L/n} \to 1$ and we obtain the inequality that we had to show. $\square$

C.3 DP from Laplace Noise

C.3.1 A Metric for (ϵ, δ)-Differential Privacy

In Sec. 3.1, we defined a distance d_{dp} that is related to ϵ-differential privacy. We would like to define a similar distance d_{DP} that is related to (ϵ, δ)-differential privacy.

Usually, we measure the distances with non-negative real numbers, but it is possible to be more general. In principle, the distances may come from any set equipped with addition and a *partial* order, where the addition is monotone and has the neutral element that is also the least element in this set. With this signature, we can state all axioms of a metric.

Consider the set $\mathbb{R}_+ \times \mathbb{R}_+$ of pairs of non-negative real numbers, where addition and ordering is componentwise. This set can serve as the set of distances, and we could consider it as the range of d_{DP}; the components somehow corresponding to "ϵ" and "δ". However, two probability distributions can be (ϵ,δ)-far from each other for different values of ϵ and δ; one can be traded for the other.

For a partially ordered set V, let $\mathcal{F}(V)$ denote the set of all *upwards closed* subsets of V. I.e. $U \subseteq V$ is an element of $\mathcal{F}(V)$ if $u \in U$ and $u \leq v$ imply $v \in U$ for all $u, v \in V$. For an arbitrary $U \subseteq V$ we let $\uparrow U$ denote the *upwards closure* of U, i.e. the smallest upwards closed set that contains U as a subset.

If V is a set of distances, then $\mathcal{F}(V)$ can also be turned into a set of distances. Let $Z_1, Z_2 \in \mathcal{F}(V)$. We have $Z_1 \leq Z_2$ iff $Z_2 \subseteq Z_1$. In this way, the entire set V (this is the only element of $\mathcal{F}(V)$ that contains the least element of V) is the least element. The addition on $\mathcal{F}(V)$ is defined by

$$Z_1 + Z_2 = \uparrow\{v_1 + v_2 \mid v_1 \in Z_1, v_2 \in Z_2\} \ . \tag{6}$$

It is easy to see that the operation $+$ is associative, commutative, has the zero element $\mathcal{F}(V)$ and is compatible with ordering (meaning that $Z_1 \leq Z_2$ implies $Z_1 + Z_3 \leq Z_2 + Z_3$ for any Z_3).

We let $\mathcal{F}(\mathbb{R}_+ \times \mathbb{R}_+)$ be the range of d_{DP}. Note that for this set, the $\uparrow$-operation can be left out from (6), due to the continuousness of $\mathbb{R}$. If X is a set, and $\chi, \chi' \in \mathcal{D}(X)$, then define

$$d_{\mathrm{DP}}(\chi,\chi') = \bigcap_{X' \subseteq X} \left\{(\epsilon,\delta) \middle| P(X',\chi,\chi',\epsilon,\delta)\right\} \ , \tag{7}$$

where

$$\begin{aligned} &P(X',\chi,\chi',\epsilon,\delta) = \\ &\quad \Pr[x \in X' \mid x \leftarrow \chi] \leq e^{\epsilon}\big(\Pr[x \in X' \mid x \leftarrow \chi'] + \delta\big) \\ &\wedge \Pr[x \in X' \mid x \leftarrow \chi'] \leq e^{\epsilon}\big(\Pr[x \in X' \mid x \leftarrow \chi] + \delta\big) \ . \end{aligned}$$

Clearly, $d_{\mathrm{DP}}(\chi,\chi') \in \mathcal{F}(\mathbb{R}_+ \times \mathbb{R}_+)$. Now, a mapping $f : X \to \mathcal{D}(Y)$ from a metric space X is (ϵ,δ)-differentially private, if it is $\uparrow\{(\epsilon,\delta)\}$-sensitive for the distance d_{DP} being used on $\mathcal{D}(Y)$.

The following proposition shows that d_{DP} satisfies the triangle inequality.

Proposition 18. *Let* $\chi_1, \chi_2, \chi_3 \in \mathcal{D}(X)$. *Then* $d_{\mathrm{DP}}(\chi_1,\chi_3) \leq d_{\mathrm{DP}}(\chi_1,\chi_2) + d_{\mathrm{DP}}(\chi_2,\chi_3)$.

Proof. Let $(\epsilon_1,\delta_1) \in d_{\mathrm{DP}}(\chi_1,\chi_2)$ and $(\epsilon_2,\delta_2) \in d_{\mathrm{DP}}(\chi_2,\chi_3)$. According to the definition of $+$, the pair $(\epsilon_1{+}\epsilon_2, \delta_1{+}\delta_2)$ is a member of $d_{\mathrm{DP}}(\chi_1,\chi_2){+}d_{\mathrm{DP}}(\chi_2,\chi_3)$. We have to show that it is also a member of $d_{\mathrm{DP}}(\chi_1,\chi_3)$ according to (7). Let $X' \subseteq X$. Then

$$\begin{aligned} &\Pr[x \in X' \mid x \leftarrow \chi_1] \\ &\quad\leq e^{\epsilon_1}\big(\Pr[x \in X' \mid x \leftarrow \chi_2] + \delta_1\big) \\ &\quad\leq e^{\epsilon_1}\Big(\big(e^{\epsilon_2}\big(\Pr[x \in X' \mid x \leftarrow \chi_3] + \delta_2\big) + \delta_1\Big) \\ &\quad= e^{\epsilon_1+\epsilon_2}\Pr[x \in X' \mid x \leftarrow \chi_3] + e^{\epsilon_1+\epsilon_2}\delta_2 + e^{\epsilon_1}\delta_1 \\ &\quad\leq e^{\epsilon_1+\epsilon_2}\big(\Pr[x \in X' \mid x \leftarrow \chi_3] + \delta_2 + \delta_1\big) \end{aligned}$$

as necessary. □

C.3.2 Relating the Metric to (ϵ,δ)-Differential Privacy

Comparing (7) to the definition of (ϵ,δ)-DP (Def. 1), we see the difference in one important aspect. Namely, in (7), the quantity δ is multiplied with e^{ϵ}, while in (1), it is not. While the difference of the factor e^{ϵ} seems small in first glance, it is not if we start considering "group privacy", i.e. distances (in X) different from 1. Let $d_X(x,x') = L$. If $f : X \to \mathcal{D}(Y)$ is $\uparrow\{(\epsilon,\delta)\}$-sensitive with respect to the distance d_{DP} on $\mathcal{D}(Y)$, then we know that $(L\epsilon, L\delta) \in d_{\mathrm{DP}}(f(x), f(x'))$. But if f is (ϵ,δ)-differentially private, then we only get

$$\Pr[f(x) \in Y'] \leq e^{L\epsilon}\Pr[f(x') \in Y'] + \frac{e^{L\epsilon}-1}{e^{\epsilon}-1}\delta$$

from Definition 1.

It is not difficult to show that if we do not multiply δ with e^{ϵ}, then d_{DP} is no longer a distance; in particular, it would not satisfy the triangle inequality. For example, let us pick

$$\begin{array}{lll} \chi_1 = \mathit{Ber}(0.01) & \chi_2 = \mathit{Ber}(0.03) & \chi_3 = \mathit{Ber}(0.07) \\ \epsilon = \ln 2 & \delta = 0.01 \ . & \end{array}$$

Here $\mathit{Ber}(p)$ is the *Bernoulli distribution.* It returns 1 with probability p and 0 with probability $1-p$. We have $(\epsilon,\delta) \in d_{\mathrm{DP}}(\chi_1,\chi_2)$ and also $(\epsilon,\delta) \in d_{\mathrm{DP}}(\chi_2,\chi_3)$, but not $(2\epsilon, 2\delta) \in d_{\mathrm{DP}}(\chi_1,\chi_3)$. Indeed,

$$\begin{aligned} \Pr[x = 1 \mid x \leftarrow \chi_2] &= 0.03 = 2 \cdot 0.01 + 0.01 \\ &= e^{\epsilon} \cdot \Pr[x = 1 \mid x \leftarrow \chi_1] + \delta \\ \Pr[x = 1 \mid x \leftarrow \chi_3] &= 0.07 = 2 \cdot 0.03 + 0.01 \\ &= e^{\epsilon} \cdot \Pr[x = 1 \mid x \leftarrow \chi_2] + \delta \\ \Pr[x = 1 \mid x \leftarrow \chi_3] &= 0.07 > 4 \cdot 0.01 + 0.02 \\ &= e^{2\epsilon} \cdot \Pr[x = 1 \mid x \leftarrow \chi_1] + 2\delta \ . \end{aligned}$$

C.3.3 Self-similarity of Laplace Distribution

Theorem 4 makes use of the distribution $\mathit{Lap}(1) \in \mathcal{D}(\mathbb{R})$, defined by $\mathit{Lap}(1)(x) \propto e^{-|x|}$. We first have to state the results about the self-similarity of $Lap(1)$ under shifting and stretching.

Lemma 19. *Let $\eta \sim Lap(1)$. Let $a_1, a_2 \in \mathbb{R}$, $c_1, c_2 \in \mathbb{R}_+$, $c_1 \leq c_2$. Define $\beta = \ln(c_2/c_1)$ and let $\epsilon \geq \beta$. Let $\delta \geq e^{-\epsilon-(\epsilon+\beta)/(e^\beta-1)}$. Then the following holds.*

$$\left(\frac{|a_2-a_1|}{c_1}, 0\right) \in d_{\mathrm{DP}}(a_1 + c_1\cdot\eta, a_2 + c_1\cdot\eta)$$
$$(\epsilon,\delta) \in d_{\mathrm{DP}}(c_1\cdot\eta, c_2\cdot\eta) \ .$$

Proof. The probability density functions (pdf) and the cumulative density functions (cdf) of the distributions named above are the following:

$$\begin{aligned}\mathrm{pdf}_{c_1\cdot\eta}(x) &= \frac{1}{2c_1}e^{-|x|/c_1}\\ \mathrm{pdf}_{a_1+c_1\cdot\eta}(x) &= \frac{1}{2c_1}e^{-|x-a_1|/c_1}\\ \mathrm{pdf}_{c_2\cdot\eta}(x) &= \frac{1}{2c_2}e^{-|x|/c_2}\\ \mathrm{pdf}_{a_2+c_1\cdot\eta}(x) &= \frac{1}{2c_1}e^{-|x-a_2|/c_1}\end{aligned}$$

and

$$\mathrm{cdf}_{c_1\cdot\eta}(x) = \begin{cases} e^{x/c_1}/2, & \text{if } x<0\\ 1-e^{-x/c_1}/2, & \text{if } x\geq 0\end{cases}$$

$$\mathrm{cdf}_{c_2\cdot\eta}(x) = \begin{cases} e^{x/c_2}/2, & \text{if } x<0\\ 1-e^{-x/c_2}/2, & \text{if } x\geq 0\end{cases}$$

The first claim of the lemma is shown by

$$\begin{aligned}\max_{x\in\mathbb{R}}\left|\ln\frac{\mathrm{pdf}_{a_1+c_1\cdot\eta}(x)}{\mathrm{pdf}_{a_2+c_1\cdot\eta}(x)}\right| &= \max_{x\in\mathbb{R}}\left|\ln\frac{e^{-|x-a_1|/c_1}}{e^{-|x-a_2|/c_1}}\right|\\ &\leq \ln e^{|a_1-a_2|/c_1} = \frac{|a_2-a_1|}{c_1}, \quad (8)\end{aligned}$$

showing that $\frac{|a_2-a_1|}{c_1} \geq d_{\mathrm{dp}}(a_1+c_1\cdot\eta, a_2+c_1\cdot\eta)$. To show the second claim, consider the following function f:

$$f(x) = \left|\ln\frac{\mathrm{pdf}_{c_1\cdot\eta}(x)}{\mathrm{pdf}_{c_2\cdot\eta}(x)}\right|.$$

We are interested in the set of x-s that satisfy $f(x)\leq\epsilon$. We have

$$f(x) = \left|\ln\left(\frac{c_2}{c_1}\cdot e^{|x|/c_2-|x|/c_1}\right)\right| = \left|\beta - \frac{c_2-c_1}{c_1c_2}|x|\right| \ .$$

The condition $f(x)\leq\epsilon$ is equivalent to

$$|x| \leq (\epsilon+\beta)\frac{c_1c_2}{c_2-c_1} \ . \quad (9)$$

To obtain the distance $\uparrow\{(\epsilon,\delta)\}$, it is sufficient to take δ equal to $e^{-\epsilon}$ times the probability that either x, when sampled according to either $c_2\cdot\eta$ or $c_1\cdot\eta$, does not satisfy (9). This probability is larger for $c_2\cdot\eta$ because $c_2\geq c_1$. Let us compute this probability.

$$\begin{aligned}\Pr[x<0\wedge f(x)>\epsilon \,|\, x\leftarrow c_2\cdot\eta] &= \frac{1}{2}e^{(\epsilon+\beta)\frac{c_1}{c_2-c_1}}\\ &= \frac{1}{2}e^{(\epsilon+\beta)\frac{1}{e^\beta-1}} \ . \quad (10)\end{aligned}$$

The probability that we are looking for is twice the quantity above. Multiplying it with $e^{-\epsilon}$ gives us the statement of the lemma. □

The next lemma provides a more coarse, but simpler upper bound for the DP-distance between stretched versions of the Laplace distribution.

Lemma 20. *Let $\eta\sim Lap(1)$. Let $c_1,c_2\in\mathbb{R}_+$, $c_1\leq c_2$. Define $\beta=\ln(c_2/c_1)$. Let $\epsilon\geq\beta$. Let $k=1+\epsilon/\beta$. Then $(\epsilon, e^{-k})\in d_{\mathrm{DP}}(c_1\cdot\eta, c_2\cdot\eta)$.*

Proof. Let $\delta = e^{-\epsilon-\frac{\epsilon+\beta}{e^\beta-1}}$. By the previous lemma, $(\epsilon,\delta)\in d_{\mathrm{DP}}(c_1\cdot\eta,c_2\cdot\eta)$. We will now show that $e^{-k}\geq\delta$.

Indeed,

$$\begin{aligned}\delta\leq e^{-k} \quad &\Leftrightarrow \quad e^{-\epsilon-\frac{\epsilon+\beta}{e^\beta-1}} \leq e^{-k} \Leftrightarrow -\epsilon - \frac{\epsilon+\beta}{e^\beta-1}\leq -k\\ &\Leftrightarrow \quad (k-1)\beta + \frac{k\beta}{e^\beta-1}\geq k\\ &\Leftrightarrow \quad \frac{1}{e^\beta-1}\geq\frac{1}{\beta}-\frac{k-1}{k}\\ &\Leftarrow \quad \frac{1}{e^\beta-1}\geq\frac{1}{\beta}-\frac{1}{2}\Leftrightarrow\frac{1}{\beta}\leq\frac{1}{2}+\frac{1}{e^\beta-1}\\ &\Leftrightarrow \quad \beta\geq\frac{2(e^\beta-1)}{e^\beta+1}\Leftrightarrow\beta+\frac{4}{e^\beta+1}\geq 2 \ ,\end{aligned}$$

where the "$\Leftarrow$" claim holds because $k\geq 2$. Consider now the function $f(x)=x+4/(e^x+1)$. We have $f(0)=2$. Also, f is a monotone function. Hence the claim $\beta+4/(e^\beta+1)\geq 2$ holds. □

C.3.4 Proof of Theorem 4

Let us restate the theorem about achieving (ϵ,δ)-DP using Laplace noise.

Theorem 4. *Let $b,\beta,\epsilon\in\mathbb{R}_+$, $b>0$, $b+\beta\leq\epsilon$. Define $k=1+(\epsilon-b)/\beta$. Let $\delta=e^{-k}$. Let η be a random variable distributed according to $Lap(1)$. Let c be a β-smooth upper bound on $\mathrm{DS}[f]$ for a function $f:X\to\mathbb{R}$, where X is Banach space and d_X is the distance corresponding to the norm of X. Define $g(\mathbf{x}) := f(\mathbf{x})+\frac{c(\mathbf{x})}{b}\cdot\eta$. Then*

– *for any $\mathbf{x}_1,\mathbf{x}_2\in X$, $(\epsilon\cdot L, 2\delta)\in d_{\mathrm{DP}}(g(\mathbf{x}_1),g(\mathbf{x}_2))$, where $L=d_X(\mathbf{x}_1,\mathbf{x}_2)$;*

– *(in particular,)* g *is* $(\epsilon, 2e^{\epsilon}\delta)$*-differentially private.*

If, additionally for any two points $\mathbf{x}_1, \mathbf{x}_2 \in X$ *there exists a shortest path* h *in* X*, such that* c *is monotonic along that path, then the factor "2" in previous statements can be removed..*

Proof. Let b, β, ϵ be as in the statement of the theorem. Let $\eta \sim Lap(1)$ and $\mathbf{x}_1, \mathbf{x}_2 \in X$. Let $L = d_X(\mathbf{x}_1, \mathbf{x}_2)$. Let h be a shortest path from $\mathbf{x}_1$ to $\mathbf{x}_2$ (note that a shortest path always exists in a Banach space). Let $\mathbf{x}_\mu$ be a point on the path h, such that $c(\mathbf{x}_\mu) = \max_{t\in[0,1]} c(h(t))$ (the maximum exists because c and h are continuous). Let $L_1 = d_X(\mathbf{x}_1, \mathbf{x}_\mu)$ and $L_2 = d_X(\mathbf{x}_\mu, \mathbf{x}_2)$. Note that $L = L_1 + L_2$. Define the following probability distributions:

$$\begin{aligned}
\chi_1 &= f(\mathbf{x}_1) + \frac{c(\mathbf{x}_1)}{b}\cdot\eta \\
\chi_2 &= f(\mathbf{x}_1) + \frac{c(\mathbf{x}_\mu)}{b}\cdot\eta \\
\chi_3 &= f(\mathbf{x}_2) + \frac{c(\mathbf{x}_\mu)}{b}\cdot\eta \\
\chi_4 &= f(\mathbf{x}_2) + \frac{c(\mathbf{x}_2)}{b}\cdot\eta \ .
\end{aligned}$$

We want to show that $(\epsilon L, 2\delta) \in d_{\mathrm{DP}}(\chi_1, \chi_4)$.

By Lemma 19, $(b \cdot |f(\mathbf{x}_2) - f(\mathbf{x}_1)|/c(\mathbf{x}_\mu), 0) \in d_{\mathrm{DP}}(\chi_2, \chi_3)$. The difference between $f(\mathbf{x}_2)$ and $f(\mathbf{x}_1)$ can be upper-bounded as follows:

$$\begin{aligned}
&|f(\mathbf{x}_2) - f(\mathbf{x}_1)| \\
&= \left|\int_h f'(h)\,ds\right| \le \left|\int_0^1 f'(h(t))\|h'(t)\|\,dt\right| \\
&\le \left|\int_0^1 c(h(t))\|h'(t)\|\,dt\right| \le \int_0^1 c(\mathbf{x}_\mu)\|h'(t)\|\,dt \\
&= c(\mathbf{x}_\mu)\cdot d_X(h(1) - h(0)) = L\cdot c(\mathbf{x}_\mu),
\end{aligned}$$

hence $(bL, 0) \in d_{\mathrm{DP}}(\chi_2, \chi_3)$. Note that, in $\int_h f'(h)\,ds$, the integral is over the path h, the derivative f' is w.r.t. distance d_X along the path h (the existence of this derivative follows from the existence of the Fréchet derivative in the definition of DS[f]), the ds is an infinitesimal distance (according to d_X) along the path h, and the h in $f'(h)$ denotes the point on the path h.

We will now compare χ_1 and χ_2. We use the previous lemma with the following instantiations:

Quantity in Lemma 20	Instantiation
c_1	$c(\mathbf{x}_1)$
c_2	$c(\mathbf{x}_\mu)$
β	$\ln(c(\mathbf{x}_\mu)/c(\mathbf{x}_1))$
ϵ	$(\epsilon - b)L_1$
k	$1 + (\epsilon - b)L_1/\ln(c(\mathbf{x}_\mu)/c(\mathbf{x}_1))$

Lemma 20 required that $\epsilon \ge \beta$. This condition is translated to

$$(\epsilon - b)L_1 \ge \ln\frac{c(\mathbf{x}_\mu)}{c(\mathbf{x}_1)} \ .$$

Let us verify that it holds:

$$\ln\frac{c(\mathbf{x}_\mu)}{c(\mathbf{x}_1))} \le \beta\cdot d_X(\mathbf{x}_1, \mathbf{x}_\mu) \le \beta L_1 \le (\epsilon - b)L_1 \qquad \blacksquare \ .$$

We also lower-bound the value of k from Lemma 20, in order to simplify its expression:

$$1 + \frac{(\epsilon - b)L_1}{\ln\frac{c(\mathbf{x}_\mu)}{c(\mathbf{x}_1))}} \ge 1 + \frac{(\epsilon - b)L_1}{\beta L_1} = k \ .$$

We obtain

$$((\epsilon - b)L_1, e^{-k}) \in d_{\mathrm{DP}}(\chi_1, \chi_2) \ .$$

Similarly, we can obtain

$$((\epsilon - b)L_2, e^{-k}) \in d_{\mathrm{DP}}(\chi_3, \chi_4) \ .$$

Using the triangle inequality, we can combine $d_{\mathrm{DP}}(\chi_i, \chi_{i+1})$ for $i \in \{1, 2, 3\}$:

$$(\epsilon L, 2e^{-k}) \in d_{\mathrm{DP}}(\chi_1, \chi_4)$$

as required. If $d_X(\mathbf{x}_1, \mathbf{x}_2) = 1$, then $(\epsilon, 2e^{-k}) \in d_{\mathrm{DP}}(g(\mathbf{x}_1), g(\mathbf{x}_2))$, i.e. g is $(\epsilon, 2\delta)$-differentially private.

If c is monotone along the path h, then the point $\mathbf{x}_\mu$ coincides with either $\mathbf{x}_1$ or $\mathbf{x}_2$. W.l.o.g. assume $\mathbf{x}_\mu = \mathbf{x}_1$. Then $\chi_1 = \chi_2$ and $(0, 0) \in d_{\mathrm{DP}}(\chi_1, \chi_2)$. The triangle inequality now gives $(\epsilon L, e^{-k}) \in d_{\mathrm{DP}}(\chi_1, \chi_4)$. □

C.4 Domination between Norms

C.4.1 Proof of Lemma 1

Let $N = N'(V_1, \ldots, V_m)$. The relation $V_i \preceq W_i$ implies $\|x_1, \ldots, x_n\|_{V_i} \le \|x_1, \ldots, x_n\|_{W_i}$ for all $x_1, \ldots, x_n \in \mathbb{R}^n$. Define a new norm $M = N'(W_1, \ldots, W_m)$. By definition of a composite norm, we have the three cases for N'.

- If $N' = |x_j|$ for some $j \in [n]$, then $m = 0$, and hence $\|x_1, \ldots, x_n\|_N = \|x_1, \ldots, x_n\|_M = |x_j|$.
- If $N' = \alpha z$, then $m = 1$, and we have $\|x_1, \ldots, x_n\|_N = \alpha\,\|x_1, \ldots, x_n\|_{V_1}$, and $\|x_1, \ldots, x_n\|_M = \alpha\,\|x_1, \ldots, x_n\|_{W_1}$, so $N \preceq M$.
- If $N' = \|z_1, \ldots, z_m\|_p$, then $\|x_1, \ldots, x_n\|_N = \left\|\|x_1, \ldots, x_n\|_{V_1}, \ldots, \|x_1, \ldots, x_n\|_{V_m}\right\|_p \le \left\|\|x_1, \ldots, x_n\|_{W_1}, \ldots, \|x_1, \ldots, x_n\|_{W_m}\right\|_p = \|x_1, \ldots, x_n\|_M$, so $N \preceq M$.

In any case, we get $N \preceq M$, which is equivalent to $N'(V_1, \ldots, V_m) \preceq N'(W_1, \ldots, W_m)$.

C.4.2 Proof of Lemma 2

Without loss of generality, we assume that all scalings in N are applied directly to the variables, as we can always apply the equality $\alpha\,\|\mathbf{x}\| = \|\alpha\mathbf{x}\|$ to push all scalings as deep as possible, directly in front of variables. Let the variable x_i occur k_i times in N, and let α_{ij} be the scaling of the j-th occurrence of x_i. We define α_i and β_i of Lemma 2 as follows.

- $\alpha_i = \sqrt[p]{\sum_{j=1}^{k_i} \alpha_{ij}}$.
- $\beta_i = \sqrt[q]{\sum_{j=1}^{k_i} \alpha_{ij}}$.

We prove the first inequality, and the proof would be analogous for the second one. Let $N = \|M_1, \ldots, M_k\|_r$. Since p is the largest ℓ_p-norm used as a term constructor of N, we have $\|M_1, \ldots, M_k\|_r \succeq \|M_1, \ldots, M_k\|_p$. Repeat the same procedure with all $M_1, \ldots, M_k$ recursively, substituting all instances of ℓ_r with ℓ_p. By Lemma 1, each step of the transformation keeps the resulting norm smaller (or equal). Finally, we are left with a composite norm N' that only contains $\|\cdot\|_p$ for the same $p \geq 1$ as a term constructor. We can now apply Lemma 13 and get a norm of the form $N' = \|\alpha_{11}x_1, \ldots, \alpha_{nk_n}x_n\|_p$, such that $N' \preceq N$.

Some variables x_i used by N' may repeat if they were repeating in N before. We may now use Lemma 14 to merge repeating variables into one, rewriting

$$\left\| \alpha_{11}x_1, \ldots, \overbrace{\alpha_{i1}x_i, \ldots, \alpha_{ik_i}x_i}^{k_i}, \ldots, \alpha_{nk_n}x_n \right\|_p = \left\| \alpha_{11}x_1, \ldots, \sqrt[p]{\sum_{j=1}^{k_i} \alpha_{ij}}\,x_i, \ldots, \alpha_{nk_n}x_n \right\|_p .$$

After doing it for all $i \in [n]$, we get a norm $N'' = \|\alpha_1 x_1, \ldots, \alpha_n x_n\|_p$, which satisfies $N'' \preceq N$.

C.5 Basic Results of Derivative Sensitivity

C.5.1 Proof of Lemma 3

Let $\nabla f(\mathbf{x}) = (a_i)_{i=1}^n$. Assuming $a_i \neq 0$ for all i (otherwise remove the indices i for which $a_i = 0$ from the summations containing a_i):

$$\begin{aligned}
|df_{\mathbf{x}}(\mathbf{y})| &= |\nabla f(\mathbf{x}) \cdot \mathbf{y}| \leq \sum_{i=1}^n |a_i||y_i| \\
&= \sum |a_i|^{\frac{p}{p-1}} \cdot \frac{|y_i|}{|a_i|^{\frac{1}{p-1}}} \\
&\leq \left(\sum |a_i|^{\frac{p}{p-1}}\right) \left(\frac{\sum |y_i|^p}{\sum |a_i|^{\frac{p}{p-1}}}\right)^{\frac{1}{p}} \\
&= \left(\sum |a_i|^{\frac{p}{p-1}}\right)^{\frac{p-1}{p}} \left(\sum |y_i|^p\right)^{\frac{1}{p}} \\
&= \|\nabla f(\mathbf{x})\|_q \cdot \|\mathbf{y}\|_p
\end{aligned}$$

for all $\mathbf{y} \in X$. The second inequality used here is the weighted power means inequality with exponents 1 and p. Equality is achievable (and not only for $\mathbf{y} = 0$): for example, by taking $y_i = |a_i|^{\frac{1}{p-1}}$. Thus $\|\nabla f(\mathbf{x})\|_q$ is the smallest value of c such that for all $\mathbf{y}$, $|df_{\mathbf{x}}(\mathbf{y})| \leq c \cdot \|\mathbf{y}\|_p$, i.e. it is the operator norm $\|df_{\mathbf{x}}\|$.

The cases $p = 1$ and $p = \infty$ can be achieved as limits of the general case.

C.5.2 Proof of Lemma 4

The derivative sensitivity of f at $\mathbf{x}$ is the operator norm of a particular linear operator $df_{\mathbf{x}}$. It is equal to the minimal possible c, such that for all vectors $\mathbf{y}$, the absolute value of of $df_{\mathbf{x}}(\mathbf{y}) \in \mathbb{R}$ is at most c times larger than the norm $\|\mathbf{y}\|_N$. If we replace N with $a \cdot N$, then the norm $\|\mathbf{y}\|_{a \cdot N}$ is increased by a times. Hence we may now reduce c by a times and still have the inequality.

C.5.3 Proof of Lemma 5

(a) We first prove that $(V, \|\cdot\|_V)$ is a normed vector space. We prove only the triangle inequality. The rest of the properties of norm are easy to check.

$$\begin{aligned}
&\|(v_1, v_2) + (v_1', v_2')\|_V = \|(v_1 + v_1', v_2 + v_2')\|_V \\
&\|(\|v_1 + v_1'\|_{V_1}, \|v_2 + v_2'\|_{V_2})\|_p \leq \\
&\|(\|v_1\|_{V_1} + \|v_1'\|_{V_1}, \|v_2\|_{V_2} + \|v_2'\|_{V_2})\|_p \leq \\
&\|(\|v_1\|_{V_1}, \|v_2\|_{V_2})\|_p + \|(\|v_1'\|_{V_1}, \|v_2'\|_{V_2})\|_p = \\
&\|(v_1, v_2)\|_V + \|(v_1', v_2')\|_V
\end{aligned}$$

The first inequality uses the triangle inequalities of $\|\cdot\|_{V_1}$ and $\|\cdot\|_{V_2}$ and the monotonicity of $\|\cdot\|_p$ in the absolute values of the coordinates of its argument vector. The second inequality uses the triangle inequality of $\|\cdot\|_p$.

Thus $(V, \|\cdot\|_V)$ is a normed vector space. It remains to prove that it is complete. Consider a Cauchy sequence $\{x_n\}$ in V. Then

$$\forall \epsilon > 0.\ \exists N \in \mathbb{N}.\ \forall m, n > N.\ \|x_m - x_n\|_V < \epsilon$$

Let $x_n = (y_n, z_n)$ where $y_n \in V_1$ and $z_n \in V_2$. Note that

$$\|y_m - y_n\|_{V_1} = \|(y_m - y_n, 0)\|_V \le \|(y_m - y_n, z_m - z_n)\|_V = \|x_m - x_n\|_V$$

Thus

$$\forall \epsilon > 0.\ \exists N \in \mathbb{N}.\ \forall m, n > N.\ \|y_m - y_n\|_{V_1} < \epsilon$$

i.e. $\{y_n\}$ is a Cauchy sequence in V_1. Because V_1 is a Banach space, there exists $y \in V_1$ such that

$$\lim_{n\to\infty} \|y_n - y\|_{V_1} = 0$$

Similarly, we get that there exists $z \in V_2$ such that

$$\lim_{n\to\infty} \|z_n - z\|_{V_2} = 0$$

Let $x = (y, z)$. Note that

$$\|x_n - x\|_V = \|(y_n - y, z_n - z)\|_V = \|(\|y_n - y\|_{V_1}, \|z_n - z\|_{V_2})\|_p$$

Then, because $\|\ \|_p$ is continuous,

$$\lim_{n\to\infty} \|x_n - x\|_V = \|(\lim_{n\to\infty} \|y_n - y\|_{V_1}, \lim_{n\to\infty} \|z_n - z\|_{V_2})\|_p = \|(0,0)\|_p = 0$$

Thus V is a Banach space.

(b) Let $c_1 = \|dg_{v_1}\|$, $c_2 = \|dh_{v_2}\|$. Note that

$$\lim_{x_1\to 0_{V_1}} \frac{|g(v_1 + x_1) - g(v_1) - df_v(x_1, 0)|}{\|x_1\|_{V_1}} = \lim_{x_1\to 0_{V_1}} \frac{|f(v_1 + x_1, v_2) - f(v_1, v_2) - df_v(x_1, 0)|}{\|(x_1, 0)\|_V} = \lim_{(x_1,0)\to 0_V} \frac{|f(v + (x_1, 0)) - f(v) - df_v(x_1, 0)|}{\|(x_1, 0)\|_V} = \lim_{x\to 0_V} \frac{|f(v + x) - f(v) - df_v(x)|}{\|x\|_V} = 0$$

The last equality holds by the definition of Fréchet derivative. The equality before that holds because the limit on the right-hand side exists. Then, again by the definition of Fréchet derivative, we get that the linear map that maps x_1 to $df_v(x_1, 0)$, is dg_{v_1}. Thus $df_v(x_1, 0) = dg_{v_1}(x_1)$. Similarly, we get $df_v(0, x_2) = dh_{v_2}(x_2)$. Now

$$\begin{aligned}|df_v(x_1, x_2)| &= |df_v(x_1, 0) + df_v(0, x_2)|\\ &= |dg_{v_1}(x_1) + dh_{v_2}(x_2)| \le |dg_{v_1}(x_1)| + |dh_{v_2}(x_2)|\\ &\le c_1\|x_1\|_{V_1} + c_2\|x_2\|_{V_2}\\ &\le \|(c_1, c_2)\|_q \cdot \|(\|x_1\|_{V_1}, \|x_2\|_{V_2})\|_p\\ &= \|(c_1, c_2)\|_q \cdot \|(x_1, x_2)\|_V \ .\end{aligned}$$

The last inequality follows from the weighted power means inequality, similarly to the proof of Lemma 3. Equality is also achievable: because $c_1 = \|dg_{v_1}\|$ and $c_2 = \|dh_{v_2}\|$, there exist x_1 and x_2 that achieve equality in the second inequality. Then scale x_1 and x_2 by constants such that $\|x_1\|_{V_1}$ and $\|x_2\|_{V_2}$ (which scale by the same constants) achieve equality in the third inequality. To achieve equality in the first inequality, we may further need to multiply x_1 and/or x_2 by -1. Thus $\|df_v\| = \|(c_1, c_2)\|_q$.

C.6 Alternative Definition of Smoothness

In this Section, we prove Lemma 6. By Def. 13, the mapping f is *β-smooth*, if $f(x) \le e^{\beta\cdot\|x'-x\|} \cdot f(x')$ for all $x, x' \in X$. We may rewrite it as $\frac{f(x)}{f(x')} \le e^{\beta\cdot\|x'-x\|} \cdot f(x')$. Applying ln to both sides, it suffices to prove that $\ln(f(x)) - \ln(f(x')) \le \beta \cdot \|x' - x\|$, which is $\frac{\ln(f(x))-\ln(f(x'))}{\|x'-x\|} \le \beta$.

Applying mean value theorem to the function $\ln \circ f : X \to \mathbb{R}$, we get $\frac{|\ln(f(x))-\ln(f(x'))|}{\|x'-x\|} = \|d(\ln\circ f)_v\|$ for some $v \in X$. Applying derivative chain rule, since $\frac{\partial \ln}{\partial x}(x) = \frac{1}{|x|}$, we get $\|d(\ln\circ f)_v\| = \frac{\|df_v\|}{|f(v)|} = \frac{\mathrm{DS}[f](v)}{|f(v)|} \le \beta$, where the last inequality comes from the lemma statement.

C.7 Smoothness of Composite Functions

C.7.1 Proof of Lemma 7

We have:

1. $\left|\frac{(f(x)+g(x))'}{f(x)+g(x)}\right| = \frac{|f'(x)+g'(x)|}{|f(x)|+|g(x)|} \le \frac{|f'(x)|+|g'(x)|}{|f(x)|+|g(x)|}$ $\le \max\left(\left|\frac{f'(x)}{f(x)}\right|, \left|\frac{g'(x)}{g(x)}\right|\right) \le \max(\beta_f, \beta_g)$.
2. $\left|\frac{(f(x)\cdot g(x))'}{f(x)\cdot g(x)}\right| = \left|\frac{f'(x)\cdot g(x)+f(x)\cdot g'(x)}{f(x)\cdot g(x)}\right|$ $\le \left|\frac{f'(x)}{f(x)}\right| + \left|\frac{g'(x)}{g(x)}\right| \le \beta_f + \beta_g$.
3. $\left|\frac{(f(x)/g(x))'}{f(x)/g(x)}\right| = \left|\frac{f'(x)\cdot g(x)-f(x)\cdot g'(x)}{g(x)^2} \cdot \frac{g(x)}{f(x)}\right|$ $= \left|\frac{f'(x)}{f(x)} - \frac{g'(x)}{g(x)}\right| \le \left|\frac{f'(x)}{f(x)}\right| + \left|\frac{g'(x)}{g(x)}\right| \le \beta_f + \beta_g$.

C.7.2 Proof of Lemma 8

Let $X = \prod_{i=1}^n X_i$ and $x = (x_1, \ldots, x_n)$. Let $\beta = \max_i \beta_i$. By Lemma 16, an upper bound on $\frac{\partial f}{\partial x_i}(x)$ is $c_i(x) =$

$f_i'(x_i)$. We have

$$|c_i(x)| = \left|f_i'(x_i)\right| \le \mathrm{DS}[f_i](x_i) = |f_i(x_i)| \cdot \frac{\mathrm{DS}[f_i](x_i)}{|f_i(x_i)|} \le |f_i(x_i)| \cdot \beta_i \ .$$

By Lemma 3 and Lemma 5, the derivative sensitivity of f in $(X, \ell_{\frac{p}{p-1}})$ is

$$\mathrm{DS}[f](x) = \|(c_1(x), \ldots, c_n(x))\|_p$$

Using inequality $|f_i(x_i)| \le |f(x)|$, we get

$$\frac{\mathrm{DS}[f](x)}{|f(x)|} \le \frac{\left\|(|f_i(x_i)| \cdot \beta_i)_{i=1}^n\right\|_p}{|f(x)|} \le \frac{|f(x)| \cdot \left\|(\beta_i)_{i=1}^n\right\|_p}{|f(x)|} \le \|(\beta_i)_{i=1}^n\|_p \ .$$

On the other hand, using inequality $\beta_i \le \beta$, we get

$$\frac{\mathrm{DS}[f](x)}{|f(x)|} \le \frac{\left\|(|f_i(x_i)| \cdot \beta)_{i=1}^n\right\|_p}{|f(x)|} \le \frac{\beta \cdot \|(|f_i(x_i)|)_{i=1}^n\|_p}{|f(x)|} = \beta$$

C.7.3 Proof of Lemma 9

Let $X = \prod_{i=1}^n X_i$ and $x = (x_1, \ldots, x_n)$. By Lemma 17, an upper bound on $\frac{\partial f}{\partial x_i}(x)$ is $c_i(x) = \max_j^n \frac{f(x)}{f_j(x)} \cdot \frac{\partial f_j}{\partial x_i}(x)$. We have

$$|c_i(x)| = \left|\max_j \frac{f(x)}{f_j(x)} \cdot \frac{\partial f_j}{\partial x_i}(x)\right| \le |f(x)| \cdot \max_j^n \left|\frac{\partial f_j}{\partial x_i}(x) \cdot \frac{1}{f_j(x)}\right| \le |f(x)| \cdot \max_j \beta_i^j \ .$$

By Lemma 3 and Lemma 5, the derivative sensitivity of f in $(X, \ell_{\mathsf{dual}(p)})$ is

$$\mathrm{DS}[f](x) = \|(c_1(x), \ldots, c_n(x))\|_p$$

We get

$$\frac{\mathrm{DS}[f](x)}{|f(x)|} \le \frac{|f(x)| \cdot \left\|\left(\max_j \beta_i^j\right)_{i=1}^n\right\|_p}{|f(x)|} \le \left\|(\max_j \beta_i^j)_{i=1}^n\right\|_p \ .$$

C.8 Derivative Sensitivity of Simple Functions

The following functions were considered in Table 2.

Power function. Let $f(x) = x^r, r \in \mathbb{R}_+, x > 0$. We have

$$\frac{f'(x)}{f(x)} = \frac{rx^{r-1}}{x^r} = \frac{r}{x} \ ; \qquad \left|\frac{r}{x}\right| \le \beta \Leftrightarrow x \ge \frac{|r|}{\beta} \ .$$

For $x \le \frac{r}{\beta}$, the function $f'(x)$ achieves its maximum at the point $\frac{r}{\beta}$. By Lemma 12, a β-smooth upper bound on f is

$$\mathrm{UB}_f(x) = \begin{cases} x^r & \text{if } x \ge \frac{r}{\beta} \\ e^{\beta x - r}\left(\frac{r}{\beta}\right)^r & \text{otherwise} \end{cases}$$

If $r \ge 1$, we may also find a smooth upper bound on the derivative sensitivity $\mathrm{DS}[f]$ of f. We have

$$\mathrm{DS}[f](x) = |f'(x)| = |r| x^{r-1} \ .$$

A β-smooth upper bound on $\mathrm{DS}[f]$ is

$$\mathrm{UB}_{\mathrm{DS}_f}(x) = \begin{cases} rx^{r-1} & \text{if } x \ge \frac{r-1}{\beta} \\ re^{\beta x - (r-1)}\left(\frac{r-1}{\beta}\right)^{r-1} & \text{otherwise} \end{cases}$$

Exponent. Let $f(x) = e^{rx}, r \in \mathbb{R}, x \in \mathbb{R}$. We have $\mathrm{DS}[f](x) = |f'(x)| = |r|e^{rx}$, hence:

– $\left|\frac{f'(x)}{f(x)}\right| = \frac{re^{rx}}{e^{rx}} = r$; $\left|\frac{f''(x)}{f(x)}\right| = \frac{r^2 e^{rx}}{re^{rx}} = r$.

Thus both f and $\mathrm{DS}[f]$ are β-smooth if $|r| \le \beta$.

Sigmoid. Consider the (sigmoid) function $\sigma(x) = \frac{e^{\alpha x}}{e^{\alpha x}+1}$. This function can be viewed as a continuous approximation of the indicator function $I_{\mathbb{R}_+} : \mathbb{R} \to \{0,1\}$, which is less precise for values close to 0, and the error decreases when α increases. We have:

– $\sigma'(x) = \frac{\alpha e^{\alpha x}}{(e^{\alpha x}+1)^2}$; $\sigma''(x) = \frac{\alpha^2 e^{\alpha x}(e^{\alpha x}-1)}{(e^{\alpha x}+1)^3}$;
– $\left|\frac{\sigma'(x)}{\sigma(x)}\right| = \left|\alpha \cdot \frac{1}{e^{\alpha x}+1}\right| \le \alpha$; $\left|\frac{\sigma''(x)}{\sigma'(x)}\right| = \left|\alpha \cdot \frac{e^{\alpha x}-1}{e^{\alpha x}+1}\right| \le \alpha$.

Thus both $\sigma(x)$ and $DS_{\sigma(x)} = |\sigma'(x)|$ are α-smooth. If we want less DP noise, we should decrease α, which in turn makes the sigmoid itself less precise, so there is a tradeoff.

Tauoid. Consider the function $\tau(x) = \frac{2}{e^{-\alpha x}+e^{\alpha x}}$ (let us call it a *tauoid*). This function can be viewed as a continuous approximation of the indicator function $I_{\{0\}} : \mathbb{R} \to \{0,1\}$, which works similarly to a sigmoid.

We have:

$$\begin{aligned}
\tau'(x) &= -\frac{2\alpha(e^{\alpha x}-e^{-\alpha x})}{(e^{-\alpha x}+e^{\alpha x})^2} \\
&= \frac{2\alpha(e^{-\alpha x}-e^{\alpha x})}{(e^{-\alpha x}+e^{\alpha x})^2} = \frac{2\alpha e^{\alpha x}(1-e^{2\alpha x})}{(1+e^{2\alpha x})^2} \\
\left|\frac{\tau'(x)}{\tau(x)}\right| &= \frac{|\alpha|\cdot|e^{-\alpha x}-e^{\alpha x}|}{e^{-\alpha x}+e^{\alpha x}} \le |\alpha| \\
|\tau'(x)| &\le \frac{2|\alpha|e^{\alpha x}}{1+e^{2\alpha x}} = \frac{2|\alpha|}{e^{-\alpha x}+e^{\alpha x}} \\
&= |\alpha|\tau(x) =: \mathrm{UB}_{\mathrm{DS}_\tau}(x) \\
\mathrm{UB}'_{\mathrm{DS}_\tau}(x) &= |\alpha|\tau'(x) \\
\left|\frac{\mathrm{UB}'_{\mathrm{DS}_\tau}(x)}{\mathrm{UB}_{\mathrm{DS}_\tau}(x)}\right| &= \left|\frac{\tau'(x)}{\tau(x)}\right| \le |\alpha| \ .
\end{aligned}$$

Thus both τ itself and $\mathrm{UB}_{\mathrm{DS}_\tau}$, an upper bound on its derivative sensitivity, are α-smooth.

An ℓ_p-norm. Consider the function $f(x) = \|x\|_p = \left(\sum x_i^p\right)^{1/p}$, $x \in \mathbb{R}^n, x = (x_1, \ldots, x_n)$. We have

$$\frac{\partial f}{\partial x_i}(x) = \frac{p x_i^{p-1}}{p\left(\sum x_i^p\right)^{(p-1)/p}} = \left(\frac{x_i^p}{\sum x_i^p}\right)^{(p-1)/p} .$$

By Lemma 3, the derivative sensitivity of f in $(\mathbb{R}^n, \ell_p)$ is

$$\mathrm{DS}[f](x) = \left(\sum \frac{x_i^p}{\sum x_i^p}\right)^{\frac{p-1}{p}} = 1 \ .$$

This is constant and thus β-smooth for all β. The function f itself is β-smooth if $\frac{1}{\|x\|_p} \le \beta$, i.e. if $\|x\|_p \ge \frac{1}{\beta}$. By Lemma 12, a β-smooth upper bound on f is

$$\mathrm{UB}_f(x) = \begin{cases} \|x\|_p & \text{if } \|x\|_p \ge \frac{1}{\beta} \\ \frac{e^{\beta\|x\|_p - 1}}{\beta} & \text{otherwise} \end{cases}$$

This also holds for $p = \infty$.

The ℓ_∞-norm. Let $f(x) = \|x\|_\infty = \max_i |x_i|$. We have

$$\frac{\partial f}{\partial x_i}(x) = \begin{cases} 1 & \text{if } i = \mathrm{argmax}_j\, |x_j| \\ \text{undefined} & \text{if } \mathrm{argmax}_j\, |x_j| \text{ is not unique} \\ 0 & \text{otherwise} \end{cases}$$

The derivative sensitivity of f in $(\mathbb{R}^n, \ell_\infty)$ is

$$\mathrm{DS}[f](x) = \begin{cases} 1 & \text{if } \mathrm{argmax}_j\, |x_j| \text{ is unique} \\ \text{undefined} & \text{if } \mathrm{argmax}_j\, |x_j| \text{ is not unique} \end{cases}$$

Because we are interested in upper bounds on the derivative sensitivity, we define

$$\mathrm{DS}[f](x_0) := \limsup_{x \to x_0} \mathrm{DS}[f](x) = 1$$

for those x_0 for which $\mathrm{DS}[f](x_0)$ is undefined. Thus $\mathrm{DS}[f](x) = 1$, which is constant and β-smooth for all β. The smooth upper bound on the function f itself can be found similarly to the ℓ_p-norm case.

C.9 Composing Derivative Sensitivity

The following constructions are considered in Fig. 1.

Product. Let $f : \prod_{i=1}^n X_i \to \mathbb{R}, f(x_1, \ldots, x_n) = \prod_{i=1}^n f_i(x_i)$ where X_i are Banach spaces. Let $X = \prod_{i=1}^n X_i$ and $x = (x_1, \ldots, x_n)$. First, suppose that variables x_i are independent. We have $\frac{\partial f}{\partial x_i}(x) = \prod_{i \ne j=1}^n f_j(x_j) \cdot f'_i(x_i)$, and $\left|\frac{\partial f}{\partial x_i}(x) \cdot \frac{1}{f(x)}\right| = \left|\frac{f'_i(x_i)}{f_i(x_i)}\right|$, hence:

- If $\left|\frac{f'_i(x_i)}{f_i(x_i)}\right| \le \beta$, then f is β-smooth w.r.t. x_i.
- By Lemmas 3 and 5,

$$\|df_x\| = \left\| \left(\prod_{i \ne j = 1}^{n} f_j(x_j) \cdot f'_i(x_i) \right)_{i=1}^{n} \right\|_{\frac{p}{p-1}}$$

  in (X, ℓ_p), so we have $\frac{\|df_x\|}{|f(x)|} = \frac{\|df_x\|}{\left|\prod_{i=1}^n f_i(x_i)\right|} = \left\|\left(\frac{f'_i(x_i)}{f_i(x_i)}\right)_{i=1}^n\right\|_{\frac{p}{p-1}} \le \left\|(\beta_i)_{i=1}^n\right\|_{\frac{p}{p-1}}$, where β_i is the smoothness of f_i. Hence, if f_i is β-smooth w.r.t. x_i for all i, then f is β-smooth in (X, ℓ_1) and $n\beta$-smooth in (X, ℓ_∞).

The derivative sensitivity of f w.r.t. x_i is $c_i(x) = \mathrm{DS}[f_i](x_i) \cdot \left|\frac{f(x)}{f_i(x_i)}\right|$. The derivative sensitivity of f in (X, ℓ_p) is, by Lemma 5, $\mathrm{DS}[f](x) = \|(c_1(x), \ldots, c_n(x))\|_{\frac{p}{p-1}} = \left\|\left(\frac{\mathrm{DS}[f_i](x_i)}{|f_i(x_i)|}\right)_{i=1}^n\right\|_{\frac{p}{p-1}} \cdot |f(x)|$.

We have $c_i(x) = \mathrm{DS}[f_i](x_i) \cdot \left|\frac{f(x)}{f_i(x_i)}\right| = \mathrm{DS}[f_i](x_i) \cdot \prod_{j \ne i} |f_j(x_j)|$. Since $\prod_{j \ne i} |f_j(x_j)|$ does not depend on x_i and $\mathrm{DS}[f_i](x_i) \ge 0$, by Lemma 7, if $\mathrm{DS}[f_i]$ is β-smooth in X_i then $c_i(x)$ is also β-smooth in X_i. Similarly, if $f_j(x_j)$ is β-smooth, then $c_i(x)$ is also β-smooth in X_j. Hence, if f_i and $\mathrm{DS}[f_i]$ are β-smooth for all i, by Lemma 9, $\mathrm{DS}[f]$ is β-smooth in (X, ℓ_1) and $n\beta$-smooth in (X, ℓ_∞). If $\mathrm{DS}[f_i]$ are not all β-smooth then we can use their β-smooth upper bounds when computing c_i. Then we get a β-smooth upper bound on $\mathrm{DS}[f]$ instead of the actual $\mathrm{DS}[f]$. This shows the correctness of the rules $(*^1_D)$ and $(*^1_S)$.

We may also consider the case where the variables x_i are fully dependent, i.e. equal (the case where they are partially dependent is currently not considered). Consider a function $f(x) = g(x) \cdot h(x)$ where $g, h : X \to \mathbb{R}_+$ and X is a Banach space. We have

$$\mathrm{DS}[f](x) = g(x) \cdot \mathrm{DS}[h](x) + h(x) \cdot \mathrm{DS}[g](x) \ .$$

By Lemma 7, if g is β_g-smooth, h is β_h-smooth, $\mathrm{DS}[g]$ is $\beta_{g'}$-smooth, and $\mathrm{DS}[h]$ is $\beta_{h'}$-smooth, then $\mathrm{DS}[f]$ is

$\max(\beta_g + \beta_{h'}, \beta_h + \beta_{g'})$-smooth. The function f itself is $(\beta_g + \beta_h)$-smooth. This shows the correctness of the rules $(*_D)$ and $(*_S)$.

Sum. Let $f : \prod_{i=1}^n X_i \to \mathbb{R}, f(x_1, \ldots, x_n) = \sum_{i=1}^n f_i(x_i)$ where X_i are Banach spaces. Let $X = \prod_{i=1}^n X_i$ and $x = (x_1, \ldots, x_n)$. First, suppose that the variables x_i are independent. The derivative sensitivity of f w.r.t. x_i is $\mathrm{DS}[f_i](x_i)$. By Lemmas 3 and 5, the derivative sensitivity of f in (X, ℓ_p) is $\mathrm{DS}[f](x) = \|\mathrm{DS}[f_1](x_1), \ldots, \mathrm{DS}[f_n](x_n)\|_{\frac{p}{p-1}}$.

- Let $f_i \geq 0$ for all $i \in \{1, \ldots, n\}$ (or $f_i \leq 0$ for all $i \in \{1, \ldots, n\}$) and β_i-smooth w.r.t. X_i. Now we have $|f(x)| = \sum_{i=1}^n |f_i(x_i)| = \left\| |f_i(x_i)|_{i=1}^n \right\|_1$. By Lemma 8, $f(x)$ is $\beta := \max_i(\beta_i)$-sensitive in (X, ℓ_p). We do not get a good bound in the case when f_i may have different signs, since then $f_i(x)$ may cancel each other out and make $f(x)$ arbitrarily small even if $|f_i(x)|$ are large.
- Let $\mathrm{DS}[f_i]$ be β_i-smooth for $i \in \{1, \ldots, n\}$. By Lemma 8, $\mathrm{DS}[f]$ is $\|(\beta_i)_{i=1}^n\|_{\frac{p}{p-1}}$-smooth in (X, ℓ_p), and if all $\mathrm{DS}[f_i]$ are β-smooth, then $\mathrm{DS}[f]$ is also β-smooth. This shows the correctness of the rule $(+_D^{\perp})$.

Consider the case where x_i are equal: $f(x) = \sum_{i=1}^n g_i(x)$ where $g_i : X \to \mathbb{R}$ and X is a Banach space. Then

$$\mathrm{DS}[f](x) = \sum_{i=1}^n \mathrm{DS}[g_i](x)$$

By Lemma 7, if all $\mathrm{DS}[g_i]$ are β-smooth then $\mathrm{DS}[f]$ is β-smooth. If all g_i are non-negative and β-smooth then f is β-smooth. This shows the correctness of the rules $(+_D)$ and $(+_S)$.

Min / max. Let $f : \prod_{i=1}^n X_i \to \mathbb{R}, f(x_1, \ldots, x_n) = \min_{i=1}^n f_i(x_i)$ where X_i are Banach spaces (the case with max instead of min is similar). Let $X = \prod_{i=1}^n X_i$ and $x = (x_1, \ldots, x_n)$. Let the variables x_i be independent.

If for all i, f_i is β-smooth in X_i then f is β-smooth in (X, ℓ_p). The same holds with max or sum (with non-negative f_i) or $\ell_{p'}$-norm instead of min.

The derivative sensitivity of f w.r.t. x_i is $\mathrm{DS}[f_i](x_i)$ if $i = \operatorname{argmin} f_i(x_i)$ and 0 otherwise. The derivative sensitivity of f in (X, ℓ_p) is $\mathrm{DS}[f](x) = \mathrm{DS}[f_i](x_i)$ where $i = \operatorname{argmin} f_i(x_i)$. In general, $\mathrm{DS}[f]$ is discontinuous at points where $\operatorname{argmin} f_i(x_i)$ is not unique.

A possible valid β-smooth (in (X, ℓ_p)) upper bound on $\mathrm{DS}[f]$ is $\max c_i(x_i)$ where c_i is a β-smooth upper bound on $\mathrm{DS}[f_i]$. This shows the correctness of the rules $(\min_D^{\perp})$ and $(\max_D^{\perp})$.

Norm scaling. Let $f : X \to \mathbb{R}$ in the Banach space $(X, \|\cdot\|)$. Scaling the norm by a scales the derivative $f'(x)$ by $\frac{1}{a}$ while keeping the value of $f(x)$ the same. Hence, if f is β-smooth in $(X, \|\cdot\|)$ then it is $\frac{\beta}{a}$-smooth in $(X, a \cdot \|\cdot\|)$. Hence the rule (N_S) is correct.

Let $c(x)$ be a β-smooth upper bound on the derivative sensitivity of f at x in $(X, \|\cdot\|)$. Then $\frac{c(x)}{a}$ is a $\frac{\beta}{a}$-smooth upper bound on the derivative sensitivity of f at x in $(X, a \cdot \|\cdot\|)$ by Lemma 4. Hence the rule (N_D) is correct.

Sensitivity w.r.t. a larger norm. Let $f : X \to \mathbb{R}$ in the Banach space $(X, \|\cdot\|_N)$. Let $\|\cdot\|_M \succeq \|\cdot\|_N$.

If f is β-smooth in $(X, \|\cdot\|_N)$, then $f(x) \leq e^{\beta\|x-x'\|_N} \cdot f(x') \leq e^{\beta\|x-x'\|_M} \cdot f(x')$ for all $x, x' \in X$, so f is also β-smooth in $(X, \|\cdot\|_M)$. The same holds about any function that is β-smooth in $(X, \|\cdot\|_N)$, including a β-smooth upper bound on the derivative sensitivity of f.

Let us show that the derivative sensitivity of f w.r.t. $\|\cdot\|_N$ is a valid upper bound on the derivative sensitivity of f w.r.t. $\|\cdot\|_M$. First, note that $\|\cdot\|_{\mathsf{dual}(N)} \succeq \|\cdot\|_{\mathsf{dual}(M)}$. Indeed, by definition of a dual norm, $\|T\|_{\mathsf{dual}(M)} = \sup\{T(x) \mid \|x\|_M \leq 1\}$ for an operator T from the dual space $X \to \mathbb{R}$ of X. Since $\|x\|_N \leq \|x\|_M$, we have $\forall x : \{T(x) \mid \|x\|_N \leq 1\} \supseteq \{T(x) \mid \|x\|_M \leq 1\}$. Hence, $\|T\|_{\mathsf{dual}(N)} = \sup\{T(x) \mid \|x\|_N \leq 1\} \geq \sup\{T(x) \mid \|x\|_M \leq 1\} = \|T\|_{\mathsf{dual}(M)}$.

By definition, we have $\mathrm{DS}[f](x) = \|df_x\|_{\mathsf{dual}(N)}$, where df_x is the Fréchet derivative of f at x. Since $\|\cdot\|_{\mathsf{dual}(N)} \succeq \|\cdot\|_{\mathsf{dual}(M)}$, we have $\|df_x\|_{\mathsf{dual}(N)} \geq \|df_x\|_{\mathsf{dual}(M)}$. Hence the rule $\preceq_D$ is correct.

Composition with a real function. Let $f(x) = h(g(x)), x \in X$ where $g : X \to \mathbb{R}, h : \mathbb{R} \to \mathbb{R}$ and X is a Banach space.

$$\mathrm{DS}[f](x) = |h'(g(x))| \cdot \mathrm{DS}[g](x)$$

$$\frac{\mathrm{DS}[f](x)}{|f(x)|} = \frac{|h'(g(x))|}{|h(g(x))|} \cdot \mathrm{DS}[g](x)$$

Suppose that h is β_h-smooth and $\mathrm{DS}[g](x) \leq B$ for all x. Then f is $\beta_h B$-smooth. We have

$$\begin{aligned}&\mathrm{DS}[\mathrm{DS}[f]](x) \\ &= |h''(g(x))|(\mathrm{DS}[g](x))^2 + |h'(g(x))| \cdot \mathrm{DS}[\mathrm{DS}[g]](x) \ ,\end{aligned}$$

$$\frac{\mathrm{DS}[\mathrm{DS}[f]](x)}{\mathrm{DS}[f](x)} = \frac{|h''(g(x))|}{|h'(g(x))|} \cdot \mathrm{DS}[g](x) + \frac{\mathrm{DS}[\mathrm{DS}[g]](x)}{\mathrm{DS}[g](x)} \ .$$

By Lemma 7, if h' is $\beta_{h'}$-smooth, $\mathrm{DS}[g]$ is $\beta_{g'}$-smooth, and $\mathrm{DS}[g](x) \leq B$ for all x then $\mathrm{DS}[f]$ is $(\beta_{h'} B + \beta_{g'})$-smooth. Hence the rules $(\circ_S)$ and $(\circ_D)$ are correct.