DUET: A Language and Type System for Statically Enforcing (ϵ, δ)-Differential Privacy

Joseph P. Near
University of Vermont
jnear@uvm.edu

David Darais
University of Vermont
David.Darais@uvm.edu

Tim Stevens
University of Vermont
Timothy.Stevens@uvm.edu

Pranav Gaddamadugu
University of California, Berkeley
pranavsaig@berkeley.edu

Lun Wang
University of California, Berkeley
wanglun@berkeley.edu

Neel Somani
University of California, Berkeley
neel@berkeley.edu

Mu Zhang
Cornell University
mz496@cornell.edu

Nikhil Sharma
University of California, Berkeley
ennsharma@berkeley.edu

Alex Shan
University of California, Berkeley
alexshan@berkeley.edu

Dawn Song
University of California, Berkeley
dawnsong@cs.berkeley.edu

Abstract

During the past decade, differential privacy has become the gold standard for protecting the privacy of individuals. However, verifying that a particular program provides differential privacy remains a manual task to be completed by an expert in the field. Language-based techniques have been proposed for fully automating proofs of differential privacy, but prior work does not support the more powerful (ϵ, δ) variant of differential privacy or does not give optimal bounds.

We propose DUET, a language and type system for automatically verifying (ϵ, δ)-differential privacy that statically derives optimal bounds on privacy cost for many practical programs, including stochastic gradient descent for machine learning. We formalize DUET's type system and complete key proofs about privacy for well-typed programs. We show how to extend DUET to support several recent variants of differential privacy which can vastly improve the accuracy of many algorithms. We apply DUET to implement several differentially private machine learning algorithms, and present experimental results which demonstrate the benefits of supporting these more powerful variants of differential privacy.

1 Introduction

Advances in big data and machine learning have achieved large-scale societal impact over the past decade. This impact is accompanied by a growing demand for data collection, aggregation and analysis at scale. This resulting explosion in the amount of data collected by organizations, however, has raised important new security and privacy concerns.

Differential privacy [17, 19, 20] is a promising technique for addressing these issues. Differential privacy allows general statistical analysis of data while protecting *data about individuals* with a strong formal guarantee of privacy. Because of its desirable formal guarantees, differential privacy has received growing attention, with ongoing real-world deployments at organizations including Google [21], Apple [1], and the US Census [25, 28]. A number of systems for performing differentially private data analytics have been built and demonstrated to be effective [26, 30, 32, 33, 35, 37].

Differential privacy also plays an increasingly important role in machine learning, as recent work has shown that a trained model can leak information about the sensitive data it was trained on [22, 40, 50]. Differential privacy provides a robust solution to this problem, and as a result, a number of differentially private algorithms have been developed for machine learning [2, 10, 14, 23, 34, 42, 44, 51].

Few practical approaches exist, however, for automatically proving that an *arbitrary program* satisfies differential privacy—an increasingly desirable goal, since many machine learning pipelines are expressed as programs that combine existing algorithms with custom code. Enforcing differential privacy for a new program currently requires a new, manually-written privacy proof. This process is arduous, and must be performed by an expert in differential privacy (and re-performed, each time the program is modified).

Type-system-based solutions to proving that a program adheres to differential privacy began with Reed and Pierce's *Fuzz* language [24, 36], which is based on linear typing. *Fuzz*, as well as subsequent work based on linear types, supports the original and most basic variant of differential privacy called *ϵ-differential privacy*. More recent variants, like (ϵ, δ)-differential privacy [20] and others [12, 13, 31], improve on ϵ-differential privacy by providing vastly more accurate answers for the same amount of privacy "cost."

Supporting these more powerful variants of differential privacy is problematic for linear type systems because the privacy bounds they provide do not scale linearly with the distance between the program's inputs—the key property that allows ϵ-differential privacy to be enforced via linear types. The definition of pure ϵ-differential privacy is equivalent to a bound on *function sensitivity*, and can be encoded as a true metric which satisfies the triangle inequality. A similar encoding of (ϵ, δ)-differential privacy, on the other hand, does *not* satisfy the triangle equality, and therefore cannot easily be bounded by a linear type system like *Fuzz*. Recent work by Azevedo de Amorim et al. [16] resolves this challenge by using a *path construction* to encode non-linear scaling, and can be used to extend *Fuzz* with support for (ϵ, δ)-differential privacy; however, this approach internalizes the use of *group privacy* [20] for most programs, and therefore provides sub-optimal bounds on privacy cost.

This paper presents Duet, a language and type system for expressing and statically verifying privacy-preserving programs. Duet supports all of the forms of differential privacy described above, and can be easily extended to new ones. Unlike previous work, Duet requires only that the privacy definition support sequential composition and be closed under post-processing. Duet automatically provides state-of-the-art privacy bounds for many interesting programs, including the Rényi differential privacy bound for gradient descent described above.

Duet supports (ϵ, δ)-differential privacy by providing *two* languages, each with its own type system—the *sensitivity language* uses linear types (as in *Fuzz*) to bound function sensitivity, and the *privacy language* is used to compose differentially private computations. Our key insight is to *disallow scaling* of privacy cost bounds in the privacy language; this difference with previous work enables the use of a distance measure which is not a true metric (as in (ϵ, δ)-differential privacy), and avoids the undesirable use of group privacy, which gives non-optimal privacy bounds. The basic privacy mechanisms of the underlying privacy definition (e.g. the Gaussian mechanism [20]) form the interface between the two languages; they share syntax and types, so programmers do not need to be aware of the split.

In addition to basic differential-privacy primitives like the Gaussian mechanism, we provide a core language design for matrix-based data analysis tasks, such as aggregation, clipping and gradients. A key challenge that we overcome in our design is how each of these features compose in terms of function sensitivity, and in a way that is general enough to support a wide range of useful applications.

We demonstrate the usefulness of Duet by implementing and verifying several differentially private machine learning algorithms from the literature, including private stochastic gradient descent [10] and private Frank-Wolfe [44]. We also implement a variant of stochastic gradient descent suitable for deep learning. For all of these programs, Duet automatically provides privacy bounds that equal *or improve upon* previously published manual privacy proofs.

We have implemented a typechecker and interpreter for Duet, and we use these to perform an empirical evaluation comparing the accuracy of models trained using our implementations. The results demonstrate the importance of supporting multiple privacy definitions: for a given level of privacy, choosing the best definition consistently results in substantially better accuracy of the trained model.

Contributions. In summary, we make the following contributions:

- We present Duet, a novel language and type system for expressing programs and automatically verifying their privacy guarantees. Unlike previous work, Duet supports multiple state-of-the-art definitions for privacy, and is capable of verifying state-of-the-art static bounds on privacy cost.
- We formalize Duet's type system and semantics, and complete key proofs about privacy of well-typed programs.
- In several case studies, we show that Duet automatically derives proofs of privacy for a number of interesting programs, including differentially private machine learning algorithms. In one case, Duet provides a bound that improves on the best previously published result.
- We conduct an experimental evaluation to demonstrate Duet's feasibility, running two machine learning algorithms implemented on several real-world datasets. Our results demonstrate the dramatic effect of improved privacy bounds on the accuracy of the trained models.

2 Preliminaries

2.1 Background: Differential Privacy

This section briefly summarizes the basics of differential privacy. See Dwork and Roth's excellent reference [20] for more information. Differential privacy considers sensitive input data represented by a vector $x \in D^n$, in which x_i represents the data contributed by user i. The *distance* between two inputs $x, y \in D^n$ is $d(x, y) = |\{i | x_i \neq y_i\}|$ (i.e. the Hamming distance between the two). Two inputs x, y are *neighbors* if $d(x, y) = 1$. A randomized mechanism $\mathcal{K} : D^n \to \mathbb{R}^d$ preserves (ϵ, δ)-differential privacy if for any neighbors $x, y \in D^n$ and any set S of possible outputs:

$$\Pr[\mathcal{K}(x) \in S] \leq e^{\epsilon} \Pr[\mathcal{K}(y) \in S] + \delta$$

The ϵ parameter, also called the *privacy budget*, controls the strength of the privacy guarantee. The δ parameter allows for a non-zero probability that the guarantee fails, and is typically set to a negligible value. The case when $\delta = 0$ is called *pure* or ϵ-differential privacy; the case when $\delta > 0$ is called *approximate* or (ϵ, δ)-differential privacy.

The *sensitivity* of a query is the amount its results can change when the database changes. The *global L1 sensitivity* of a query $f : D^n \to \mathbb{R}^d$ is $GS_f = \max_{x,y:d(x,y)=1} |f(x) - f(y)|_1$. The $L2$ sensitivity is analogous, using the $L2$ norm.

Differential privacy mechanisms. Two basic differential privacy mechanisms are the *Laplace mechanism* [19], which preserves $(\epsilon, 0)$-differential privacy, and the *Gaussian mechanism* [20], which preserves (ϵ, δ)-differential privacy. For a function $f : D^n \to \mathbb{R}^d$ with $L1$ sensitivity of Δ, the Laplace mechanism adds noise drawn from $\text{Lap}(\frac{\Delta}{\epsilon})$ to each element of the output. For f with $L2$ sensitivity of Δ_2, the Gaussian mechanism adds noise drawn from $\mathcal{N}(0, \frac{2\Delta_2^2 \ln(1.25/\delta)}{\epsilon^2})$ to each element.

The exponential mechanism [29] selects an element of a set based on the scores assigned to each element by a *scoring function*. Let $u : D^n \times \mathcal{R} \to \mathbb{R}$ be a scoring function with $L1$ sensitivity Δ. The mechanism selects and outputs an element $r \in \mathcal{R}$ with probability proportional to $\exp(\frac{\epsilon u(x,r)}{2\Delta})$, and preserves $(\epsilon, 0)$-differential privacy.

Composition. A key property of differential privacy is that differentially private computations *compose*. The sequential composition theorem says that if $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfy (ϵ, δ)-differential privacy, then their combination satisfies $(2\epsilon, 2\delta)$-differential privacy.

Tighter bounds on privacy cost can be achieved using the advanced composition theorem [20], at the expense of increasing δ. The advanced composition theorem says that for $0 < \epsilon' < 1$ and $\delta' > 0$, the class of (ϵ, δ)-differentially private mechanisms satisfies $(\epsilon', k\delta + \delta')$-differential privacy under k-fold adaptive composition for $\epsilon' = 2\epsilon\sqrt{2k \ln(1/\delta')}$.

The moments accountant was introduced by Talwar et al. [2] specifically for stochastic gradient descent in deep learning applications. It provides tight bounds on privacy loss in iterative applications of the Gaussian mechanism, as in SGD. However, the moments accountant is not compositional: it cannot be applied to nested loops, for example.

Recent variations of differential privacy aim to generalize the tighter composition bounds of the moments accountant. Rényi differential privacy [31], zero-concentrated differential privacy [13], and truncated concentrated differential privacy [12] are compositional variants of differential privacy that provide tighter composition bounds for iterative algorithms.

2.2 Differentially Private Gradient Descent

As a running example, we implement a differentially-private gradient descent algorithm first proposed by Song et al. [42] and later refined by Bassily et al. [11]. Gradient descent is a simple but effective training algorithm in machine learning, and has been applied in a wide range of contexts, from simple linear models to deep neural networks.

```
private-gradient-descent X y k η ϵ δ ≜
  let X1 = clip-matrix X in
  let θ0 = zeros (cols X1) in
  loop [δ'] k on θ0 <X1, y> {t, θ ⇒
      gp ← noisy-grad θ X1 y ϵ δ ;
      return θ − η · gp   }
```

Figure 1. Differentially Private Gradient Descent in DUET

Machine learning problems are typically defined in terms of a *loss function* $\mathcal{L}(\theta; X, y)$ on a *model* θ, *training examples* $X = (x_1, x_2, ..., x_n)$ (in which each example is typically represented as a *feature vector*) and corresponding *labels* $y = (y_1, y_2, ..., y_n)$ (i.e. the prediction target). The training task is to find a model $\hat{\theta}$ which minimizes the loss on the training examples (i.e. $\hat{\theta} = \text{argmin}_\theta \mathcal{L}(\theta; X, y)$).

Gradient descent solves the training problem by starting with an initial guess for θ and iteratively moving in the direction of an improved θ until the current setting is close to $\hat{\theta}$. To determine which direction to move, the algorithm evaluates the gradient of the loss, which yields a vector representing the direction of greatest *increase* in $\mathcal{L}(\theta; X, y)$. Then, the algorithm moves in the *opposite* direction.

Figure 1 contains a DUET implementation of differentially private gradient descent similar to the one proposed by Bassily et al. [11] (this version does not use minibatching, though we will extend it to do so in Section 6). It performs k iterations of gradient descent, starting from an initial guess θ_0 consisting of all zeros. At each iteration, the algorithm computes a noisy gradient using noisy-grad, scales the gradient by the *learning rate* η, and subtracts the result from the current model θ to arrive at the updated model.

To ensure differential privacy, this definition relies on adding noise directly to the gradient in each iteration. The added noise can reduce the accuracy of the final model, so we would like to use the smallest amount of noise sufficient to ensure privacy. A manual analysis of the privacy cost of this algorithm, like the one performed by Bassily et al. [11], relies on three components: (1) the sensitivity of the gradient computation, (2) the type and amount of noise added to the gradient, and (3) the composition strategy used to determine the total privacy cost over all iterations.

We take advantage of a Lipschitz property of the gradient to bound its sensitivity: for many loss functions (e.g., logistic loss, which we use here), if the $L2$ norm of each example $x_i \in X$ is bounded by 1, then the gradient has $L2$ sensitivity of 1. We ensure that this condition is satisfied by *clipping* each example (using clip-matrix). The final gradient is the average over all examples, so its global sensitivity is $\frac{1}{n}$, where n is the number of examples in the training set.

This bound on sensitivity is standard, but we have several choices in how to approach the privacy proof. In fact, this definition satisfies a number of *different* bounds on privacy cost, based on the strategy used for the other two parts of the

Approach	Privacy Bound
Std. comp. [20]	$(k\epsilon, 0)$-DP
Adv. comp. [20]	$(2\epsilon\sqrt{2k\log(1/\delta)}, \delta)$-DP
Adv. comp. [11]	$(2\epsilon\sqrt{2k\log(1/\delta')}, \delta + k\delta')$-DP
Mom. acct. [2]	$(4\epsilon\sqrt{k\log(1/\delta)}, \delta)$-DP
RDP [31]	$(\alpha, k\epsilon)$-RDP
zCDP [13]	$k\rho$-zCDP
tCDP [12]	$(k\rho, \omega)$-tCDP hline

Table 1. Statically Derivable Privacy Bounds for the Duet Implementation of Gradient Descent in Figure 1

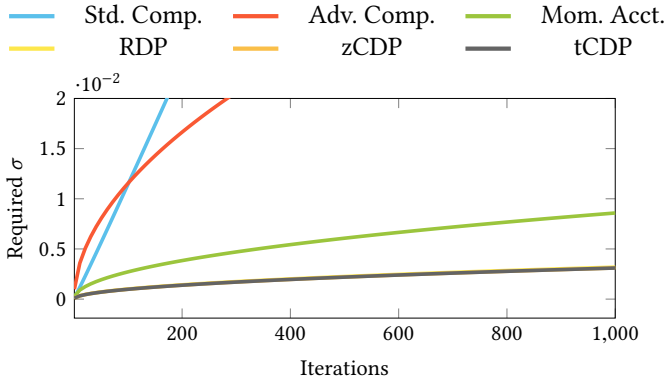

Figure 2. Noise Necessary to Achieve $(1, 10^{-5})$-Differential Privacy for Gradient Descent on a Dataset of 50,000 Samples, Under Variants of Differential Privacy

analysis. Table 1 summarizes the associated privacy bounds under the various possible strategies.

Each of the bounds in Table 1 can be converted to (ϵ, δ)-differential privacy, allowing a direct comparison. For each one, Figure 2 plots the variance of the per-iteration noise required to ensure $(1, 10^{-5})$-differential privacy for the whole training process. The plot demonstrates that a good choice of composition theorem can produce the desired level of privacy using *much less noise*, yielding a more accurate model.

Note that the Duet implementation of private gradient descent in Figure 1 runs for a fixed number of iterations (k), so it is guaranteed to terminate. This is a general property of differentially private programs with loops: each iteration consumes some fraction of the privacy budget, so the program *must* terminate in order to have finite privacy cost.

2.3 Language-Based Techniques for Automatic Bounds on Privacy Cost

Our primary contribution is a language and type system which, for programs like gradient descent, is capable of *automatically* deriving all of the privacy bounds shown in Table 1.Previous language-based techniques share our goal of automatically deriving tight bounds on privacy cost for iterative algorithms, but none is capable of statically deriving state-of-the-art bounds for algorithms like gradient descent.

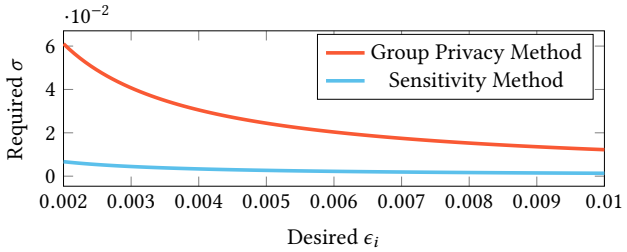

Figure 3. Noise Necessary for Desired Per-iteration ϵ_i under Sensitivity Method and Group Privacy Method, for Gradient Descent on a Dataset of 50,000 Samples

Fuzz and DFuzz. To automate proofs of differential privacy, Reed and Pierce developed *Fuzz* [36], a type system that uses linear types [45] to prove bounds on the sensitivity of arbitrary functions. Gaboardi et al. extended *Fuzz* to *DFuzz* [24], which adds *dependent types* (types which may depend on values) to *Fuzz*. This allows the privacy cost of a function to depend on one of its inputs at runtime—for example, the number of iterations in gradient descent.

Both *Fuzz* and *DFuzz* are restricted to $(\epsilon, 0)$-differentially private algorithms (i.e. they cannot prove (ϵ, δ)-differential privacy). As pointed out by Azevedo De Amorim et al. [16], the definition of ϵ-differential privacy is based on max divergence, which is a proper metric that satisfies the triangle inequality. ϵ-differential privacy can therefore be stated as a sensitivity bound. Most other notions of divergence used in privacy definitions—including the skew divergence for (ϵ, δ)-differential privacy and the Rényi divergence for several other variants—*do not* satisfy the triangle inequality, and so they cannot be expressed in a system like *Fuzz* which is based on distance metrics. *Fuzz* and *DFuzz* are therefore capable of deriving only the "Laplace+Std. Comp." privacy bounds plotted in Figure 2.

Adaptive Fuzz. In follow-on work, Winograd-Cort and Haeberlen developed *Adaptive Fuzz* [49], which adds support for (ϵ, δ)-differential privacy via a *runtime* component. Adaptive Fuzz uses *Fuzz* to prove $(\epsilon, 0)$-differential privacy for each *iteration* of an iterative algorithm, then uses the runtime component to apply advanced composition. This approach is therefore capable of achieving the "Laplace+Adv. Comp." privacy bounds plotted in Figure 2. However, the Adaptive Fuzz approach outputs the total (ϵ, δ) cost *after* the program has finished executing. Adaptive Fuzz is not capable of providing static bounds on privacy cost.

Path Metrics. More recently, Azevedo de Amorim et al. [16] developed a type system for (ϵ, δ)-differential privacy using a *path metric* construction combined with a type system based on *Fuzz*. This approach is capable of statically deriving the "Gaussian+Adv. Comp." privacy bounds plotted in Figure 2. We conjecture that the path construction approach could also be extended to support other variants like RDP, zCDP, and tCDP.

$$m,n \in \mathbb{N} \qquad r \in \mathbb{R} \qquad \dot{r},\epsilon,\delta \in \mathbb{R}^+ \qquad x,y \in \text{var}$$
$$s \in \text{sens} ::= \dot{r} \mid \infty \qquad p \in \text{priv} ::= \epsilon,\delta \mid \infty$$

$\tau \in \text{type}$	$::= \mathbb{N}[n] \mid \mathbb{R}^+[\dot{r}] \mid \mathbb{N} \mid \mathbb{R} \mid \texttt{data}$	*numeric types*
	$\mid \tau \multimap_s \tau \mid (\tau@p,\ldots) \multimap^* \tau$	*functions*
$\Gamma_s \in \text{tcxt}_s$	$\triangleq \text{var} \rightharpoonup \text{sens} \times \text{type} ::= \{x :_s \tau,\ldots\}$	*sens. contexts*
$\Gamma_p \in \text{tcxt}_p$	$\triangleq \text{var} \rightharpoonup \text{priv} \times \text{type} ::= \{x :_p \tau,\ldots\}$	*priv. contexts*
$e \in \text{exp}$	$::= \mathbb{N}[n] \mid \mathbb{N}[\dot{r}] \mid n \mid r \mid \texttt{real}\ e$	*numeric literals*
	$\mid e + e \mid e - e \mid e \cdot e \mid 1/e \mid e\ \texttt{mod}\ e$	*arithmetic*
	$\mid \lambda\ x{:}\tau \Rightarrow e \mid p\lambda\ (x{:}\tau,\ldots) \Rightarrow e$	*sens./priv. fun.*
	$\mid x \mid \texttt{let}\ x = e\ \texttt{in}\ e \mid e\ e$	*let/sens. app.*
$e \in \text{exp}$	$::= \texttt{return}\ e \mid x \leftarrow e\ ;\ e \mid e(x,\ldots)$	*ret/bind/priv. app.*
	$\mid \texttt{loop}[e]\ e\ \texttt{on}\ e\ \langle x,\ldots\rangle\ \{x,x \Rightarrow e\}$	*finite iteration*
	$\mid \texttt{gauss}[e,e,e]\ \langle x,\ldots\rangle\ \{e\}$	*gaussian noise*
	$\mid \texttt{laplace}[e,e]\ \langle x,\ldots\rangle\ \{e\}$	*laplacean noise*

Figure 4. Core Types and Terms

However, the path metric approach works by leveraging group privacy, which often requires adding much more noise than necessary in the context of variants other than pure $(\epsilon, 0)$-differential privacy. Figure 3 compares the scale of Gaussian noise (σ^2) required under the traditional *sensitivity method* (in which noise is scaled to function sensitivity) and the *group privacy method* used in the path metric approach (in which sensitivity is set to 1, and group privacy is used to derive new privacy bounds for the function's true sensitivity). For an example application of gradient descent with a dataset of size $n = 50,000$, a per-iteration $\epsilon_i = 0.004$ yields a total $\epsilon = 1.21$ under advanced composition. To achieve this level of privacy, the group privacy method adds nearly *four times more noise* than the sensitivity method. For this reason, group privacy is not typically used in manual privacy proofs, and it is likewise undesirable for automated verification approaches.

3 DUET: A Language for Privacy

This section describes the syntax, type system and formal properties of DUET, as applied to (ϵ, δ)-differential privacy. We extend DUET to other variants of differential privacy in Section 5.

Language Design. The DUET language and type system is the result of three key insights. First, *scaling in linear type systems is not a good fit for (ϵ, δ)-differential privacy*, since privacy cost is not a true metric. This insight led us to develop a relaxed linear type system for DUET that *does not allow scaling*.

Second, *sensitivity and privacy cost are distinct concepts*, each with its own properties, and the two should be tracked separately. For example, sensitivity *is* a true metric, and a type system like DFuzz [24] is appropriate for tracking it. This insight led us to design DUET as a combination of *two* languages, sharing types but each with its own typing rules. DUET's *sensitivity language* is similar to DFuzz, and is described in Section 3.1; its *privacy language* is designed specifically for tracking privacy, and is described in Section 3.2.

Third, verifying differential privacy for practical machine learning programs requires *new datatypes* which encode useful properties for that domain. For example, verifying the DUET implementation of gradient descent from the last section requires a matrix datatype capable of recording a bound on the $L2$ norm of each of its rows. We discuss these new datatypes and corresponding language features for machine learning in Section 4.

Syntax & Typing Rules. Figure 4 shows a core subset of syntax for both languages. We will show select extensions to this core throughout this section—see the appendix for the complete presentation of the full language. The sensitivity and privacy languages share syntax for variables and types, which are typeset in blue. Expressions in the sensitivity language are typeset in green, while expressions in the privacy language are typeset in red. Type contexts in the sensitivity language track the *sensitivity* s of each variable whereas in the privacy language they track *privacy cost* p. Sensitivities are non-negative reals $\dot{r}$ extended with infinity, and privacy costs are pairs of non-negative reals—representing (ϵ, δ)—also extended with infinity. The sensitivity function space (a la Fuzz) is $\tau_1 \multimap_s \tau_2$ which encodes an s-sensitive function from τ_1 to τ_2. The privacy function space (novel in DUET) is $(\ldots,\tau_n@p_n) \multimap^* \tau$ which encodes a multi-arity function that preserves p_i-privacy for its ith argument.

Figure 5 shows a core subset of typing rules for both languages. In the typing rules, the languages embed within each other—sensitivity typing contexts are transformed into privacy contexts, and vice versa.

3.1 Sensitivity Language

DUET's sensitivity language is similar to that of *DFuzz* [24], except that we extend it with significant new tools for machine learning in Section 4. We do not present standard linear logic connectives (a la Fuzz) or symbolic type-level expressions (a la DFuzz) here. We reuse notation conventions from Fuzz, e.g., $\Gamma_1 + \Gamma_2$ is a partial function which assumes that each context agrees on types associated with each variable and takes the sum of variable sensitivities, and $s\Gamma$ is the context with each sensitivity in Γ scaled by s.

The only new term in our sensitivity language is the privacy lambda $p\lambda\ (x{:}\tau,\ldots) \Rightarrow e$. Privacy lambdas are mult-arity (as opposed to single-arity sensitivity lambdas) because the privacy language does not support currying to recover multi-argument functions (i.e., it is not cartesian closed). Privacy lambdas are created in the sensitivity language and applied in the privacy language with $e(x,\ldots)$. This structure follows an "effect" discipline where the sensitivity language is pure, and creating an effectful function does not perform effects.

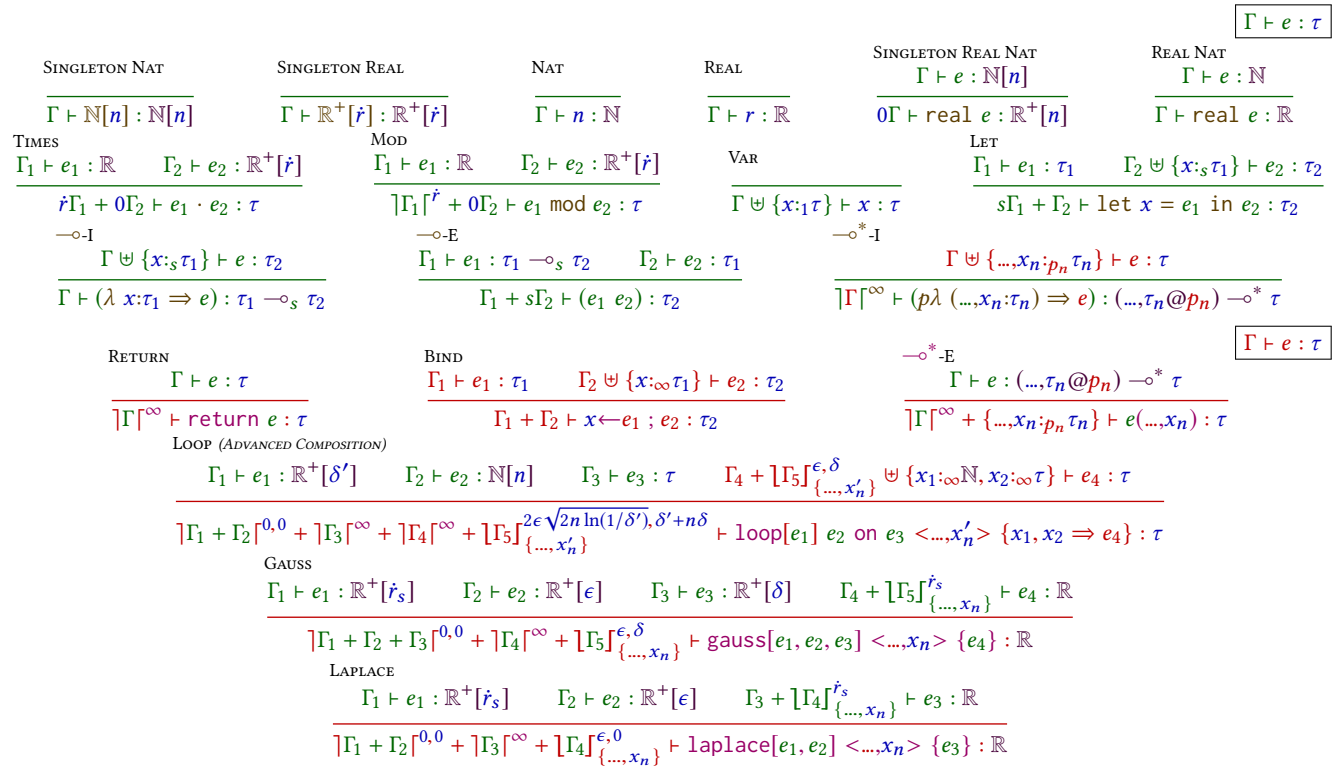

Figure 5. Core Typing Rules

The typing rule for privacy lambdas (⊸*-I) typechecks the body of the lambda in a privacy type context extended with its formal parameters, and the privacy cost of each parameter is annotated on its function argument type. Unlike sensitivity lambdas, the privacy cost of variables in the closure environment are not preserved in the resulting typing judgment. Because the body of the lambda is checked in a privacy type context, and the resulting context must be a sensitivity one, we make no guarantees about sensitivity for any variables that were found to be non-private. (0-sensitivity and $(0,0)$-differential privacy coincide in their interpretation, and all other instances of sensitivity and privacy are unrelated.) We notate the "no guarantees" operation $\rceil\Gamma\lceil^{\infty}$, which is defined for both privacy costs p and privacy context Γ:

$$\rceil\_\lceil^{-} \in (\text{priv} \times \text{sens} \to \text{sens}) \uplus (\text{tcxt}_p \times \text{sens} \to \text{tcxt}_s)$$

$$\rceil\epsilon,\delta\lceil^{s} \triangleq \begin{cases} 0 & \text{if} \quad \epsilon{=}0 \wedge \delta{=}0 \\ s & \text{if} \quad \epsilon{\neq}0 \vee \delta{\neq}0 \end{cases} \qquad \rceil\infty\lceil^{s} \triangleq s \qquad \rceil\Gamma\lceil^{s}(x) \triangleq \rceil\Gamma(x)\lceil^{s}$$

We use this operation frequently in typing judgments, including in the sensitivity-to-privacy direction.

3.2 Privacy Language

Duet's privacy language is designed specifically to enable the composition of individual differentially private computations. It has a linear type system, but unlike the sensitivity language, annotations instead track privacy cost, and the privacy language *does not allow scaling* of these annotations, that is, $p\Gamma$ is not used and cannot be defined. Syntax `return` e and x←e;e (pronounced "bind") are standard from Fuzz. Privacy application $e(x,\ldots)$ applies a $p\lambda$ (created in the sensitivity language) to a sequence of *variable* arguments. `loop` is loop iteration fixed to a statically known number of iterations, and `gauss` and `laplace` are mechanisms of differential privacy. These syntactic forms take a list of variables (<x,...>) to indicate which should be considered when calculating final privacy costs, as explained shortly.

Typing Rules for Return and Bind are standard from Fuzz, although note our explicit conversion from a sensitivity Γ to a privacy Γ in Return. Bind encodes exactly the post-processing property of differential privacy—it allows e_2 to use the value computed by e_1 without additional privacy cost.

Privacy application (⊸*-E) checks that the sensitivity term produces a privacy function and applies its privacy costs to function arguments which are syntactically restricted to be variables. This restriction is crucial for type soundness—arbitrary terms cannot be given privacy bounds statically due to the lack of scaling in the model for (ϵ,δ)-differential privacy. Both `return` and privacy application make use of the "makes no guarantees" operation discussed previously, but in the sensitivity-to-privacy direction.

The typing rule Loop encodes advanced composition for (ϵ,δ)-differential privacy. e_1 is the δ' parameter to the advanced composition bound and e_2 is the number of loop iterations—each of these values must be statically known,

Domain	*Metric:* $\lvert \_ - \_ \rvert \in X \to X \to \mathbb{R} \uplus \{\infty\}$
real	$\lvert r_1 - r_2 \rvert \triangleq \lvert r_1 - r_2 \rvert_{\mathbb{R}}$
data	$\lvert r_1 - r_2 \rvert \triangleq \begin{cases} 0 & \text{when } r_1 = r_2 \\ 1 & \text{when } r_1 \neq r_2 \end{cases}$
$\text{matrix}[n_1, n_2]^{\ell_c}_{L\infty}(D)$	$\lvert m_1 - m_2 \rvert \triangleq \sum_i \max_j \lvert m_1[i,j] - m_2[i,j] \rvert_D$
$\text{matrix}[n_1, n_2]^{\ell_c}_{L1}(D)$	$\lvert m_1 - m_2 \rvert \triangleq \sum_{i,j} \lvert m_1[i,j] - m_2[i,j] \rvert_D$
$\text{matrix}[n_1, n_2]^{\ell_c}_{L2}(D)$	$\lvert m_1 - m_2 \rvert \triangleq \sum_i \sqrt{\sum_j (\lvert m_1[i,j] - m_2[i,j] \rvert_D)^2}$

Figure 6. Distance Metrics for Matrices

which we encode with singleton types (a la DFuzz). Statically known values are fixed, resulting in a 0,0 privacy cost for those subexpressions, written $\lceil \Gamma_1 + \Gamma_2 \rceil^{0,0}$. e_3 is the initial value passed to the loop—for which no claim is made of privacy, indicated by $\lceil \Gamma_3 \rceil^{\infty}$. e_4 is a loop body with free variables x_1 and x_2 which will be iterated e_2 times on e_3 as the starting value. The loop body e_4 is checked in a privacy context $\Gamma_4 + \lfloor \Gamma_5 \rfloor^{\epsilon,\delta}_{\{\ldots,x'_n\}}$, shorthand for $\lceil \lfloor \Gamma_5 \rfloor_{\{\ldots,x'_n\}} \rceil^{\epsilon,\delta}$ where $\lfloor \Gamma_5 \rfloor_{\{\ldots,x'_n\}}$ is a context restricted to only the variables $\ldots, x'_n$. ϵ, δ is an upper bound on the privacy cost of the variables x'_i in the loop body, and the resulting privacy bound is restricted to only those variables. This allows variables for which the programmer is not interested in tracking privacy to appear in Γ_4 in the premise, and the rule's conclusion makes no claims about privacy for these variables.

The typing rules GAUSS and LAPLACE are similar in spirit to LOOP: they take parameters to the mechanism which must be statically known (encoded as singleton types), a list of variables to consider for the purposes of the resulting privacy bound, and a term $\{e\}$ for which there is a bound $\dot{r}$ on the sensitivity of free variables $x_1, \ldots, x_n$. The resulting privacy guarantee is that the term in brackets $\{e\}$ is (ϵ, δ)-differentially private. Whereas `loop` and advanced composition considers a *privacy term* loop body with an upper bound on *privacy leakage*, `gauss` and `laplace` considers a *sensitivity term* body with an upper bound on its *sensitivity*.

3.3 Metatheory

We denote our sensitivity language $e \in \text{exp}$ into total functional maps between metric spaces—the same model as the terminating fragment of Fuzz. Every term in our language terminates by design, which dramatically simplifies our models and proofs. This restriction poses no issues in implementing differentially private machine learning alrogorithms, because such algorithms trivially terminate in a fininte number of loop iterations in order to achieve a fixed, finite privacy cost. Additionally, modeling nontermination for a language which both enforces (ϵ, δ)-differential privacy and supports nontermination is a very challenging open problem that even the path metric technique from Azevedo de Amorim et al. [16] is not equipped to solve.

Types in DUET denote metric spaces, as in Fuzz. We notate metric spaces D, their underlying carrier set $\|D\|$, and their distance metric $|x-y|_D$, or $|x-y|$ where D is can be inferred from context. Sensitivity typing judgments $\Gamma \vdash e : \tau$ denote linear maps (also as in Fuzz) from a scaled cartesian product interpretation of Γ:

$$[\![\overbrace{\{x_1 :_{s_1} \tau_1, \ldots, x_n :_{s_n} \tau_n\}}^{\Gamma} \vdash \tau]\!] \triangleq !_{s_1}[\![\tau_1]\!] \otimes \cdots \otimes !_{s_n}[\![\tau_n]\!] \multimap [\![\tau]\!]$$

Although we do not make metric space scaling explicit in our syntax (because scaling for linear types types does not interact well with type inference), it becomes apparent explicitly in our model. Privacy judgments $\Gamma \vdash e : \tau$ denote *probabilistic, privacy preserving maps* from an *unscaled* product interpretation of Γ:

$$[\![\overbrace{\{x_1 :_{p_1} \tau_1, \ldots, x_n :_{p_n} \tau_n\}}^{\Gamma} \vdash \tau]\!] \triangleq ([\![\tau_1]\!]@p_1, \ldots, [\![\tau_n]\!]@p_n) \multimap^* \|[\![\tau]\!]\|$$

The multi-arity (ϵ, δ)-differential-privacy-preserving map is non-standard and defined:

$$\begin{aligned} &(D_1@(\epsilon_1, \delta_1), \ldots, D_n@(\epsilon_n, \delta_n)) \multimap^* X \triangleq \\ &\quad \{\, f \in \|D_1\| \times \cdots \times \|D_n\| \to \mathcal{D}(X) \\ &\quad \mid |x_i - y|_{D_i} \leq 1 \Rightarrow \\ &\qquad \Pr[f(x_1, \ldots, x_i, \ldots, x_n) = d] \leq \\ &\qquad e^{\epsilon_i} \Pr[f(x_1, \ldots, y, \ldots, x_n) = d] + \delta_i \;\} \end{aligned}$$

where $\mathcal{D}(X)$ is a distribution over elements in X.

We give a full semantic account of typing and type contexts in the appendix, as well as prove key lemmas about type soundness, many of which appeal to well-known differential privacy bounds from the literature.

The final soundness theorem, proven by induction, is that the denotation for a well-typed open term e in well-typed environment mapping variables to values γ is contained in the denotation of its type τ.

Theorem 3.1.

1. *If* $\Gamma \vdash e : \tau$ *and* $\gamma : \Gamma$ *then* $[\![e]\!]^{\gamma} \in [\![\Gamma \vdash \tau]\!]$
2. *If* $\Gamma \vdash e : \tau$ *and* $\gamma : \Gamma$ *then* $[\![e]\!]^{\gamma} \in [\![\Gamma \vdash \tau]\!]$

A corollary is that any well-typed privacy lambda function satisfies (ϵ, δ)-differential privacy for the privacy annotation used in typing.

4 Language Tools for Machine Learning

As described in Section 2.2, machine learning algorithms typically operate over a training set of samples, and differentially private algorithms often rely on special properties of gradient computations. Implementations of these algorithms often represent datasets using matrices and gradients using vectors. To allow expressing these algorithms, we add a $\mathbb{M}^c_{\ell}[m, n]\ \tau$ datatype to DUET, encode vectors as single-row matrices, and add typing rules for gradient computations that encode desirable properties.

$\ell \in \text{norm} ::= L1 \mid L2 \mid L\infty \qquad c \in \text{clip} ::= \ell \mid U$

$\tau \in \text{type} ::= \ldots \mid \texttt{idx}[n] \mid \mathbb{M}^{c}_{\ell}[n,n]\ \tau$ *matrix types*

$g \in \text{grad} ::= LS \mid LR \mid GR \mid HSVM$ *gradients*

$e \in \text{exp} ::= \ldots \mid \texttt{mcreate}_{\ell}[e,e]\{x,x \Rightarrow e\}$ *matrix creation*
$\mid\ e\#[e,e] \mid e\#[e,e \mapsto e]$ *index/update*
$\mid\ \texttt{rows}\ e \mid \texttt{cols}\ e \mid \texttt{clip}^{\ell}\ e \mid \texttt{conv}\ e$ *dim./clip/conv*
$\mid\ L\nabla^{g}_{\ell}[e;e,e] \mid U\nabla[e;e,e]$ *gradients*
$\mid\ \texttt{map}\ e\ \{x \Rightarrow e\}$ *matrix map*
$\mid\ \texttt{map-row}\ e\ \{x \Rightarrow e\}$ *map row*
$\mid\ \texttt{fld-row}\ e\ \texttt{on}\ e\ \{x,x \Rightarrow e\}$ *fold row*

$e \in \text{exp} ::= \ldots \mid \texttt{mgauss}[e,e,e]\ \texttt{<}x,\ldots\texttt{>}\ \{e\}$ *matrix gaussian*
$\mid\ \texttt{exponential}[e,e]\ e\ \texttt{<}x,\ldots\texttt{>}\ \{x \Rightarrow e\}$ *exp. mechanism*

Figure 7. Matrix Types and Terms

The m and n parameters refer to the number of rows and columns in the matrix, respectively. The ℓ parameter determines the distance metric used for the matrix metric for the purposes of sensitivity analysis; the c parameter is used to specify a norm bound on each row of the matrix, which will be useful when applying gradient functions.

4.1 Distance Metrics for Matrices

Differentially private machine learning algorithms typically move from one distance metric on matrices and vectors to another as the algorithm progresses. For example, two input training datasets are neighbors if they differ on exactly one sample (i.e. one row of the matrix), but they may differ arbitrarily in that row. After computing a gradient, the algorithm may consider the $L2$ sensitivity of the resulting vector—i.e. two gradients g_1 and g_2 are neighbors if $\|g_1 - g_2\|_2 \leq 1$. These are very different notions of distance—but the first is required by the definition of differential privacy, and the second is required as a condition on the input to the Gaussian mechanism.

The ℓ annotation on matrix types in Duet enables specifying the desired notion of distance between *rows*. The distance between two matrices is always equal to the sum of the distances between rows. The distance metric for the element datatype τ determines the distance between two corresponding elements, and the row metric ℓ specifies how to combine elementwise distances to determine the distance between two rows.

Figure 6 presents the complete set of distance metrics for matrices, as well as real numbers and the domain data for elements of the $\mathbb{D}$ type, which is a copy of $\mathbb{R}$ but with a discrete distance metric. Many combinations are possible, including the following common ones:

Ex. 1: $|X - X'|_{\mathbb{M}^{U}_{L\infty}[m,n]\ \mathbb{D}} = \sum_i \max_j |X_{i,j} - X'_{i,j}|_{\mathbb{D}}$

Distance is the *number of rows on which X and X' differ*; commonly used to describe neighboring input datasets.

Ex. 2: $|X - X'|_{\mathbb{M}^{U}_{L1}[m,n]\ \mathbb{R}} = \sum_i \sum_j |X_{i,j} - X'_{i,j}|_{\mathbb{R}}$

Distance is the *sum of elementwise differences.*

Ex. 3: $|X - X'|_{\mathbb{M}^{U}_{L2}[m,n]\ \mathbb{R}} = \sum_i \sqrt{\sum_j |X_{i,j} - X'_{i,j}|^2_{\mathbb{R}}}$

Distance is *sum of the $L2$ norm of the differences between corresponding rows.*

Ex. 4: : $|X - X'|_{\mathbb{M}^{U}_{L2}[1,n]\ \mathbb{R}} = \sqrt{\sum_j |X_{1,j} - X'_{1,j}|^2_{\mathbb{R}}}$

Represents a vector; distance is $L2$ sensitivity for vectors, as required by the Gaussian mechanism.

4.2 Vector-Valued Privacy Mechanisms

Both the Laplace and Gaussian mechanisms are capable of operating directly over vectors; the Laplace mechanism adds noise calibrated to the $L1$ sensitivity of the vector, while the Gaussian mechanism uses its $L2$ sensitivity. With the addition of matrices to Duet, we can introduce typing rules for these vector-valued mechanisms, using single-row matrices to represent vectors. We present the typing rule for MGauss in Figure 8; the rule for MLaplace is similar. We also introduce a typing rule for the exponential mechanism, which picks one element out of an input vector based on a sensitive scoring function (Figure 8, rule Exponential).

4.3 Gradients and Clipping

To prove that noisy gradient descent satisfies differential privacy, we bound the sensitivity of the gradient computation. The gradients for many kinds of convex loss functions satisfy a Lipschitz property, which ensures that if each sample in $X = (x_1, \ldots, x_n)$ has bounded $L2$ norm (i.e. $\|x_i\|_2 \leq 1$), then for all models θ and labelings y, the gradient $\nabla(\theta; X, y)$ has $L2$ sensitivity bounded by 1. Duet encodes this property in the Lipschitz Gradient typing rule in Figure 8.

To ensure that each sample has bounded $L2$ norm, $L\nabla$ requires input of type $\mathbb{M}^{L2}_{L\infty}[m,n]\ \mathbb{D}$. We obtain such a matrix by *clipping*, a common operation in differentially private machine learning. Clipping scales each row of a matrix to ensure its c norm (for $c \in \{L\infty, L1, L2\}$) is less than 1:

$$\text{clip}^{c}\ x_i \triangleq \begin{cases} \frac{x_i}{\|x_i\|_c} & \text{if} \quad \|x_i\|_c > 1 \\ x_i & \text{if} \quad \|x_i\|_c \leq 1 \end{cases}$$

The clipping process is encoded in Duet in the Clip rule (Figure 8), which introduces a new bound on the c norm of its output. We are now ready to implement the clip-matrix, zeros, and noisy-grad helpers for our Duet implementation of gradient descent.

```
clip-matrix Xj ≜ mmap X {x_i ⇒ clip^{L2} x_i}
zeros n ≜ mcreate_{L∞}[1,n]{i,j ⇒ 0.0}

noisy-grad θ X y ε δ ≜
  let s = ℝ[1.0]/real (rows X) in
  let z = zeros (cols X) in
  let gs = mmap-row X,y {x_i,y_i ⇒ L∇^{GR}_{L2}[θ;x_i,y_i]} in
  let g = mfold-row gs on z {x_1,x_2 ⇒ x_1 + x_2} in
  let g_s = mmap g {x ⇒ s * x} in
  mgauss[s,ε,δ] <X,y> {g_s}
```

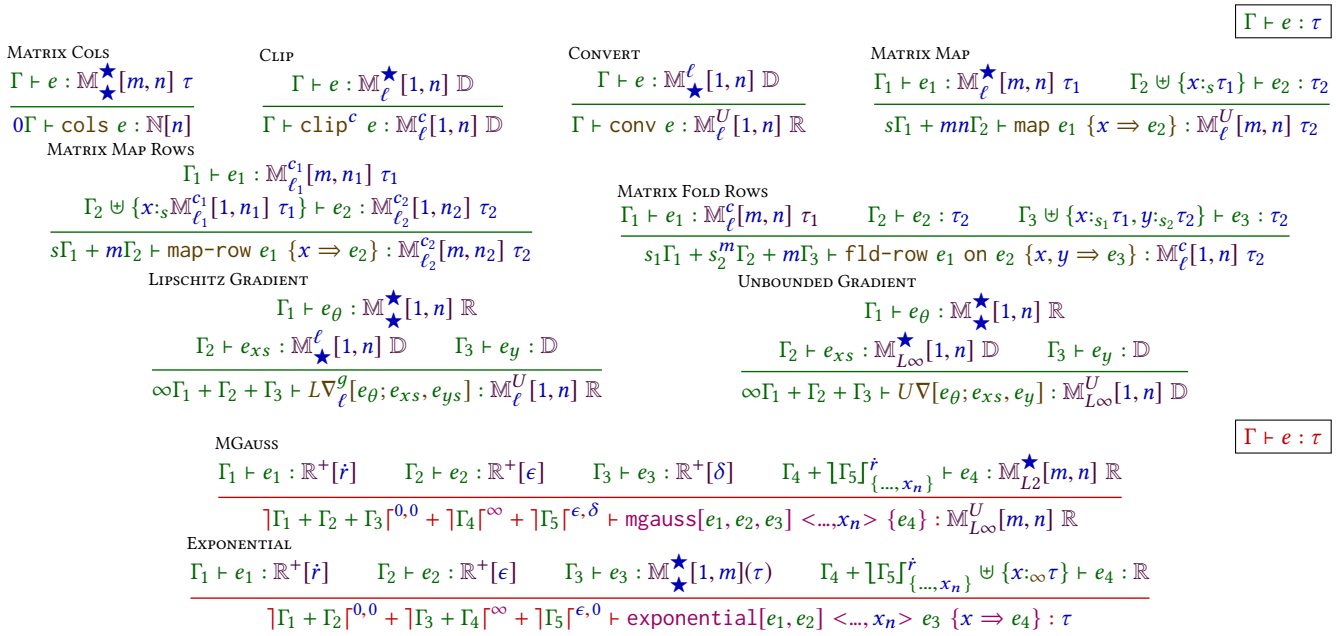

Figure 8. Matrix Typing Rules

The type of sensitivity lets embedded in the privacy language, as used above and in Figure 1, are interpreted via inlining, not encoded using return and bind. This is to avoid destroying the sensitivity of the variables referenced in the RHS. We can similarly use `mmap` to implement matrix scaling and elementwise arithmetic operators (e.g. $+$, $-$, $\cdot$, and $\times$). With these helpers, Figure 1 gives a complete, (ϵ, δ)-differentially private implementation of gradient descent in DUET.

5 Variants of Differential Privacy

The DUET language and type system are designed for easy extension to new variants of differential privacy. New variants can be supported by making three simple changes:

- Modify the *privacy cost* syntax p to describe the privacy parameters associated with the new definition.
- Modify the sum operator $\_+\_$ to reflect sequential composition in the new definition.
- Modify the typing for basic mechanisms (e.g. `gauss`) to reflect corresponding mechanisms in the new definition.

The soundness proof for the new type system will also be nearly identical to the existing proof. The only cases that require updating are the ones for RETURN, BIND, and the basic mechanisms. Importantly, the $p\lambda$ and application cases do *not* change.

Figure 9 presents the changes needed to extend DUET to three recent variants of differential privacy: Rényi differential privacy (RDP) [31], zero-concentrated differential privacy (zCDP) [13], and truncated concentrated differential privacy (tCDP) [12]. Each one has different privacy parameters and a different form of sequential composition, summarized in Figure 9. The basic mechanism for RDP and zCDP is the Gaussian mechanism; tCDP uses a novel *sinh-normal* mechanism [12] which decays more quickly in its tails.

All three provide asymptotically tight bounds on privacy cost under composition, while at the same time eliminating the "catastrophic" privacy failure that can occur with probability δ under (ϵ, δ)-differential privacy. In each of these variants, the "LOOP" rule for (ϵ, δ)-differential privacy is replaced by a simpler (yet still optimal) rule that merely iterates the sequential composition privacy bounds.

The use of these more advanced variants of differential privacy can yield dramatically more accurate algorithms, as demonstrated by our experimental results in Section 7. Easy extension to new variants is a major strength of DUET's design; it means that as new variants are developed, they can immediately be applied to existing programs to derive improved privacy bounds.

6 Case Studies

In this section, we study DUET implementations of two additional variants of gradient descent, a DUET implementation of the private Frank-Wolfe algorithm, and a DUET implementation of optimized local hashing (an algorithm for local differential privacy). In all three cases, DUET matches or improves upon the best previously published privacy bound, all of which were derived by manual proof.

6.1 Gradient Descent

We can extend the simple version of gradient descent from Figure 1 in various ways to more closely match the variants of the algorithm widely used in practice.

Variant	**Privacy Cost**	**Sequential Composition**	k**-Loop Composition**	**Basic Mechanism**
RDP [31]	$\alpha, \epsilon \mid \infty$	$(\alpha, \epsilon_1) + (\alpha, \epsilon_2) \triangleq (\alpha, \epsilon_1 + \epsilon_2)$	$(\alpha, k\epsilon)$	`gauss[`s, α, ϵ`] {`e`}`
zCDP [13]	$\rho \mid \infty$	$\rho_1 + \rho_2 \triangleq \rho_1 + \rho_2$	$k\rho$	`gauss[`s, ρ`] {`e`}`
tCDP [12]	$\rho, \omega \mid \infty$	$(\rho_1, \omega_1) + (\rho_2, \omega_2) \triangleq (\rho_1 + \rho_2, \min(\omega_1, \omega_2))$	$(k\rho, \omega)$	`sinh-normal[`s, ρ, ω`] {`e`}`

Figure 9. Duet Extensions for Recent Variants of Differential Privacy

Minibatching. Bassily et al. [11] present an algorithm for differentially private stochastic gradient descent. Their approach samples a single random example from the training to compute the gradient in each iteration, and leverages the idea of *privacy amplification* to improve privacy cost. The privacy amplification lemma states that if mechanism $\mathcal{M}(D)$ provides (ϵ, δ)-differential privacy for the datset D of size n, then running $\mathcal{M}$ on uniformly random γn entries of D (for $\gamma \leq 1$) provides $(2\gamma\epsilon, \gamma\delta)$-differential privacy [11, 47] (this bound is loose, but used here for readability).

We encode the privacy amplification lemma in Duet using a new language construct called `sample`. The expression

$$\texttt{sample}\ b\ \texttt{on}\ X, y\ \{X', y' \Rightarrow e\}$$

samples b corresponding rows from X and y, and runs e with access to the sampled rows. We extend the type system with a rule that encodes the privacy bounds from the privacy amplification lemma:

$$\textsc{Sample}\quad \frac{\begin{array}{l}\Delta, \Gamma_1 \vdash e_1 : \mathbb{N}[m_2] \quad m_2 \leq m_1 \quad \dot{r}_\epsilon = 2m_2/m_1 \quad \dot{r}_\delta = m_2/m_1 \\ \Gamma_2 \supseteq \{x_1 :_{(\dot{r}_\epsilon \epsilon_1, \dot{r}_\delta \delta_1)} \mathbb{M}^{c_1}_{\ell_1}[m_1, n_1]\ \tau_1, x_2 :_{(\dot{r}_\epsilon \epsilon_2, \dot{r}_\delta \delta_2)} \mathbb{M}^{c_2}_{\ell_2}[m_1, n_2]\ \tau_2\} \\ \Delta, \Gamma_3 \uplus \{y_1 :_{(\epsilon_1, \delta_1)} \mathbb{M}^{c_1}_{\ell_1}[m_2, n_1]\ \tau_1, y_2 :_{(\epsilon_2, \delta_2)} \mathbb{M}^{c_2}_{\ell_2}[m_2, n_2]\ \tau_2\} \vdash e_2 : \tau_3\end{array}}{\rceil \Gamma_1 \lceil^{0,0} + \Gamma_2 + \Gamma_3 \vdash \texttt{sample}\ e_1\ \texttt{on}\ x_1, x_2\ \{y_1, y_2 \Rightarrow e_2\} : \tau_3}$$

Similar privacy amplification lemmas exist for RDP [47] and tCDP [12]. Privacy amplification is not possible under zCDP. We can use the `sample` construct to implement minibatching SGD in Duet:

```
minibatch-gradient-descent X y k b η ϵ δ ≜
  let X₁ = clip-matrix X in
  loop [δ] k on zeros (cols X₁) <X₁, y> {t, θ ⇒
    sample b on X₁, y {X₁', y' ⇒
      g_p ← noisy-grad θ X₁' y' ϵ δ ; return θ − η · g_p }}
```

Under (ϵ, δ)-differential privacy with privacy amplification, Duet derives the same privacy bounds for this algorithm as the manual proof of Bassily et al. [11].

Gradient Clipping. We can also use Duet to implement stochastic gradient descent with gradient clipping, as commonly used for differentially private deep learning [2, 39]. This requires clipping the gradient (using `clip`) for each training example, then calling `conv` to transform the resulting matrix of `data` elements to a matrix of $\mathbb{R}$ elements. The `conv` construct is typed by the Convert rule, which encodes the fact that a `data` matrix with bounded norm can be converted to a $\mathbb{R}$ matrix with bounded *sensitivity*. This results in a sensitivity of $\frac{1}{n}$ for the averaged clipped gradient, even though no sensitivity bound is known for $U\nabla$ (as is commonly the case in deep learning). Finally, we modify the definition of SGD from above by *not* clipping the training examples, and instead using the gradient-clipping helper.

```
noisy-unbounded-grad θ X y ϵ δ ≜
  let s = ℝ[1.0]/real (rows X) in
  let z = zeros (cols X) in
  let gs = mmap-row X, y {x_i, y_i ⇒
      conv (clip^L2 (U∇[θ; x_i, y_i]))} in
  let g = mfold-row gs on z {x₁, x₂ ⇒ x₁ + x₂} in
  let g_s = mmap g {x ⇒ s ∗ x} in
  mgauss_L2[s, ϵ, δ] <X, y> {g_s}

unbounded-minibatch-gradient-descent X y k b η ϵ δ ≜
  let θ₀ = zeros (cols X) in
  loop [δ] k on θ₀ <X, y> {t, θ ⇒
    sample b on X, y {X', y' ⇒
      g_p ← noisy-unbounded-grad θ X' y' ϵ δ ;
      return θ − η · g_p } }
```

Using zero-concentrated differential privacy or Rényi differential privacy, Duet matches the manually-derived privacy bounds proven by Talwar et al. [2].

6.2 The Frank-Wolfe Algorithm

Unlike private gradient descent, the private Frank-Wolfe algorithm [44] has dimension-independent utility, making it useful for high-dimensional datasets. In each iteration, the algorithm takes a step of fixed size in a *single* dimension, using the exponential mechanism to choose the best direction based on the gradient. The sensitivity of each update is therefore dependent on the L_∞ norm of each sample, rather than the L_2 norm.

Our implementation uses the exponential mechanism to select the direction in which the gradient has its maximum value, then updates θ in only the selected dimension. To get the right sensitivity, we compute the gradient with $L\nabla^{LR}_{L\infty}$, which requires an $L\infty$ norm bound on its input and ensures bounded $L\infty$ sensitivity.

```
frank-wolfe X y k ϵ δ ≜
  let X₁ = clip-matrix_L∞ X in
  let d = cols X in
  let θ₀ = zeros d in
  let idxs = mcreate_L∞[1,2·d]{i,j ⇒ ⟨j mod d, sign(j − d)⟩} in
  loop [δ] k on θ₀ {t, θ ⇒
    let μ = 1.0/((real t) + 2.0) in
    let g = L∇^LR_L∞[θ; X₁, y] in
    ⟨i, s⟩ ← exponential[1/(rows X₁), ϵ] idxs {⟨i, s⟩ ⇒ s · g#[0, i]} ;
    let g_p = (zeros d)#[0, i ↦ s · 100] in
    return ((1.0 − μ) · θ) + (μ · g_p) }
```

Each iteration of the algorithm is $(\epsilon, 0)$-differentially private. Using RDP, zCDP, or tCDP, Duet derives *asymptotically*

better bounds than the previously published manual analysis [44]. This illustrates a strength of our approach: as new privacy definitions are developed, DUET can automatically derive new bounds on privacy cost for existing algorithms, without new manual proofs.

6.3 Optimized Local Hashing

Our final case study examines the settingof *local* differential privacy, in which each individual adds noise to his or her data *before* submitting it to a centralized aggregator. This setting contrasts with the *central* model of differential privacy, which we have targeted in the previous examples, in which a trusted aggregator collects (un-noised) sensitive data and adds noise to the *result of processing* the data. The local model eliminates the need for a trusted aggregator, and is the basis for Google's RAPPOR [21] and Apple's differential privacy system [1].

DUET applies equally well in this setting. As a representative example, we implement the *Optimized Local Hashing* (OLH) [46] algorithm, which generalizes the classic idea of randomized response [48]. The algorithm's input is a single element of a finite domain of size d (e.g. in RAPPOR, the URL of the user's homepage); its output is a (randomized) element of the same domain. Note that OLH does not require (ϵ, δ)-differential privacy or iteration, so it could also be implemented in *DFuzz*. Our DUET implementation requires a prime number p larger than d, in addition to d and the sensitive index x of the selected element. It leverages the `rand-resp` construct (typing rule given below).

$$\begin{array}{l}\texttt{OLH}\ d\ p\ x \triangleq \\ \quad a \leftarrow \texttt{rand-nat}(1, p)\ ; \\ \quad b \leftarrow \texttt{rand-nat}(0, p)\ ; \\ \quad \texttt{rand-resp}[d+1, \epsilon]\ \{((a \cdot x + b)\ \texttt{mod}\ p)\ \texttt{mod}\ (d+1)\}\end{array}$$

RAND-RESP

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{N}[n] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\epsilon] \qquad \Gamma_3 + \lfloor \Gamma_4 \rfloor^n_{\{\ldots, x_n\}} \vdash e_3 : \mathbb{N}}{\rceil \Gamma_1 + \Gamma_2 \lceil^{0,0} + \rceil \Gamma_3 \lceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\epsilon,0}_{\{\ldots, x_n\}} \vdash \texttt{rand-resp}[e_1, e_2] <\ldots, x_n> \{e_3\} : \mathbb{N}}$$

To improve utility, OLH encodes the input using a family of hash functions before invoking randomized response; the use of `mod` bounds sensitivity. For a complete analysis of OLH, see Blocki et al. [46]; DUET derives the same state-of-the-art privacy bounds as the manual privacy proof.

7 Implementation & Evaluation

This section describes our implementation of DUET, and our empirical evaluation of DUET's ability to produce accurate differentially private models. Our results demonstrate that the state-of-the-art privacy bounds derivable by DUET can result in huge gains in accuracy for a given level of privacy.

7.1 Implementation

We have implemented a prototype of a large subset of DUET in Haskell that includes a type inferencer and an interpreter and runs on all examples described in this paper. The interpreter recursively evaluates expressions in sensitivity and privacy languages. We use the *Hmatrix* library for matrix operations, making it fast enough to train models on real datasets in minutes. We use a multiply-with-carry pseudo-random generator and the interpreter is able to efficiently generate large amounts of random indices needed for mini-batching.

The type inferencer roughly follows the bottom-up approach of *DFuzz*'s implementation [15]. Type inferencing requires solving constraints involving expressions containing $\ln(\cdot)$ and $\sqrt{\cdot}$. *DFuzz* uses a satisfiability modulo theories (SMT) solver, but SMT solvers typically do not support operators like $\ln$ and $\sqrt{\cdot}$. DUET therefore implements a custom solver for inequalities between sensitivity and privacy cost expressions. The solver is based on a simple decidable (but incomplete) theory for these inequalities, and is sufficient to perform automatic inference on the case study programs in Section 6.

Our implementation is open source and freely available on GitHub at: https://github.com/uvm-plas/duet.

Dataset	# Samples	# Dim.	# Classes
Adult	45,220	104	2
KDDCup99	70,000	114	2
Facebook	40,949	54	2
Gisette	6,000	5,000	2

Table 2. Datasets, with number of samples, number of dimensions, and number of classes, for algorithm utility evaluation.

7.2 Evaluation of Private Gradient Descent and Private Frank-Wolfe

Our evaluation studies the accuracy of the models produced by the DUET implementations of private gradient descent and private Frank-Wolfe in Section 6. We evaluate both algorithms on 4 datasets. Details about the datasets—including the number of samples and dimension in each—can be found in Table 2.

We ran both algorithms on each dataset with per-iteration $\epsilon_i \in \{0.0001, 0.001, 0.01, 0.1\}$ and then used DUET to derive the corresponding total privacy cost. We fixed $\delta = \frac{1}{n^2}$, where n is the size of the dataset. For private gradient descent, we set $\eta = 1.0$, and for private Frank-Wolfe we set the size of each corner $c = 100$.

We randomly shuffled each dataset, then chose 80% of the dataset as training data and reserved 20% for testing. We ran each training algorithm 5 times on the training data, and take the average testing error over all 5, to account for the randomness in the training process.

We present the results in Figures 10 (private SGD) and 11. Both algorithms are capable of generating accurate models at reasonable values of ϵ. The results demonstrate the huge advantages afforded by recently developed variants of differential privacy—recent variants with tighter composition bounds yield good accuracy with *orders of magnitude* smaller values for ϵ.

8 Related Work

The initial definition of differential privacy is due to Cynthia Dwork [17–19]. The book by Dwork and Roth [20] provides an excellent overview of techniques in differential privacy.

Systems for Differential Privacy. Several systems exist for enforcing differential privacy on database-style queries. The first of these, Privacy Integrated Queries (PINQ) [30], answers database queries and implements the Laplace mechanism with a measure of global sensitivity. The calculus used in PINQ can be viewed as a privacy type system for such queries.

Other systems include FLEX [26], for SQL queries; Airavat [37], for MapReduce programs; DJoin [33] for queries over distributed datasets; and GUPT [32], which implements the Sample & Aggregate framework for Python programs.

Differentially private machine learning. A large body of work exists on differentially private convex empirical risk minimization (ERM), from which we draw the algorithms considered in this paper. The first algorithms proposed were output perturbation and objective perturbation [14]. The private SGD algorithm was first proposed by Song et al. [42], with improved risk bounds due to Bassily et al. [10]. More recently, an improved versions of output perturbation [51] has been proposed. The high-dimensional setting has also been studied [27, 41], and the private Frank-Wolfe algorithm we consider in this paper is due to Talwar et al. [43].

In the non-convex case, achieving differential privacy is more challenging. The first approach for deep learning was proposed by Shokri and Shmatikov [39], and employed a private SGD algorithm in combination with clamping of the gradient. More recently, Talwar et al. [2] improved on this result by introducing the *moments accountant*, which yields tighter bounds on the privacy loss. As we have shown, Duet is capable of reproducing both results automatically.

Language-based approaches. Language-based approaches for differential privacy fall into two categories: approaches based on type systems, and those based on relational verification. Barthe et al. [7] provide a survey. The type-system based approaches are most related to our work, but relational verification approaches have also received considerable attention in recent years [4, 6, 8, 9, 38].

Approaches based on type systems began with *Fuzz* [36], which uses linear types to track sensitivity. *DFuzz* [15, 24] adds dependent types to allow for more expressive sensitivity annotations. Winograd-Cort et al. [49] add a runtime component to *Fuzz* to provide (ϵ, δ)-differential privacy for iterative algorithms. Our work builds on *Fuzz* and *DFuzz* to provide a static guarantees for variants other than $(\epsilon, 0)$-differential privacy.

Recent work has made considerable progress in proving differential privacy for low-level mechanisms. The approach used in HOARe^2 [5] uses refinement types; LightDP [52] has a relational type system; and Albarghouthi and Hsu [3] build on LightDP to prove more expressive algorithms. All three are capable of proving (ϵ, δ)-differentially privacy for algorithms, but do not focus on automation. These tools are complementary to our work: they would enable mechanizing the privacy proofs for the built-in mechanisms Duet provides.

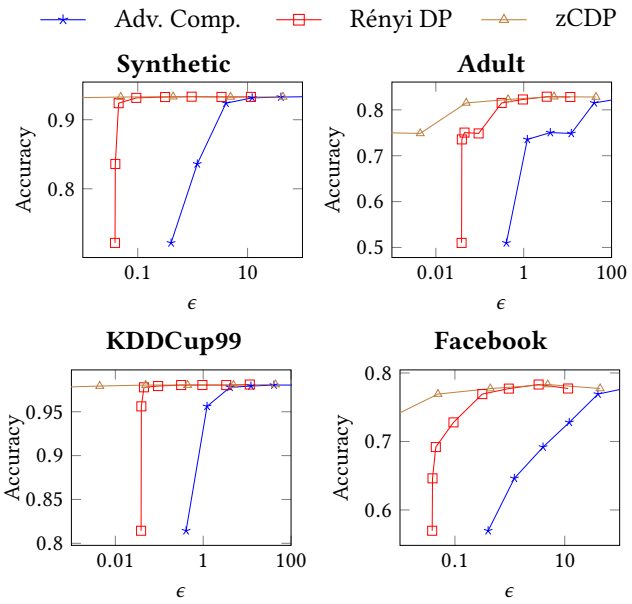

Figure 10. Results for Private Gradient Descent

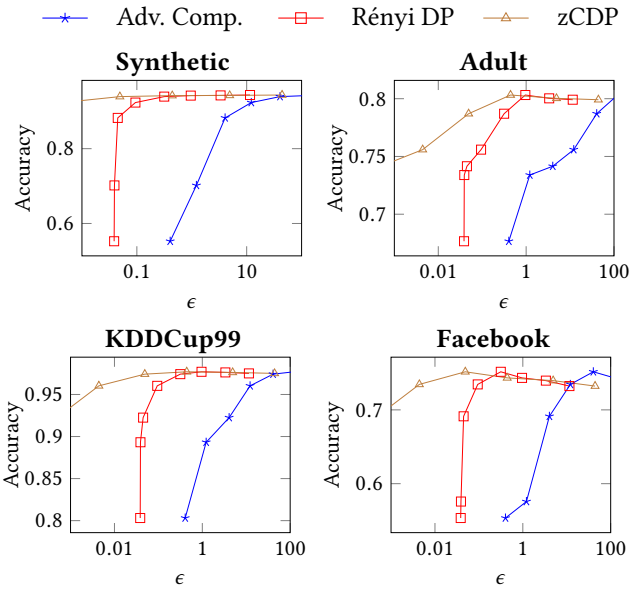

Figure 11. Results for Private Frank-Wolfe

9 Conclusion

We have presented Duet, a a language and type system for expressing and statically verifying privacy-preserving

programs. Unlike previous work, Duet is agnostic to the underlying privacy definition, and requires only that it support sequential composition and post-processing. We have extended Duet to support several recent variants of differential privacy, and our case studies demonstrate that Duet derives state-of-the-art privacy bounds for a number of useful machine learning algorithms. We have implemented a prototype of Duet, and our experimental results demonstrate the benefits of flexibility in privacy definition.

Acknowledgments

The authors would like to thank Arthur Azevedo de Amorim, Justin Hsu, and Om Thakkar for their insightful comments and helpful discussions. This work was supported by the Center for Long-Term Cybersecurity, and DARPA & SPAWAR under contract N66001-15-C-4066. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright notation thereon. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] Apple previews iOS 10, the biggest iOS release ever. http://www.apple.com/newsroom/2016/06/apple-previews-ios-10-biggest-ios-release-ever.html. (????).

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 308–318.

[3] Aws Albarghouthi and Justin Hsu. 2017. Synthesizing coupling proofs of differential privacy. *Proceedings of the ACM on Programming Languages* 2, POPL (2017), 58.

[4] Gilles Barthe, Noémie Fong, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2016. Advanced probabilistic couplings for differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 55–67.

[5] Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. 2015. Higher-order approximate relational refinement types for mechanism design and differential privacy. In *ACM SIGPLAN Notices*, Vol. 50. ACM, 55–68.

[6] Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2016. Proving differential privacy via probabilistic couplings. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*. ACM, 749–758.

[7] Gilles Barthe, Marco Gaboardi, Justin Hsu, and Benjamin Pierce. 2016. Programming language techniques for differential privacy. *ACM SIGLOG News* 3, 1 (2016), 34–53.

[8] Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella-Béguelin. 2013. Probabilistic relational reasoning for differential privacy. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 35, 3 (2013), 9.

[9] Gilles Barthe and Federico Olmedo. 2013. Beyond differential privacy: Composition theorems and relational logic for f-divergences between probabilistic programs. In *International Colloquium on Automata, Languages, and Programming*. Springer, 49–60.

[10] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*. IEEE, 464–473.

[11] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*. IEEE, 464–473.

[12] Mark Bun, Cynthia Dwork, Guy N Rothblum, and Thomas Steinke. 2018. Composable and versatile privacy via truncated CDP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 74–86.

[13] Mark Bun and Thomas Steinke. 2016. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*. Springer, 635–658.

[14] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. 2011. Differentially private empirical risk minimization. *Journal of Machine Learning Research* 12, Mar (2011), 1069–1109.

[15] Arthur Azevedo De Amorim, Marco Gaboardi, Emilio Jesús Gallego Arias, and Justin Hsu. 2014. Really Natural Linear Indexed Type Checking. In *Proceedings of the 26nd 2014 International Symposium on Implementation and Application of Functional Languages*. ACM, 5.

[16] Arthur Azevedo de Amorim, Marco Gaboardi, Justin Hsu, and Shinya Katsumata. 2018. Metric Semantics for Probabilistic Relational Reasoning. *CoRR* abs/1807.05091 (2018). arXiv:1807.05091 http://arxiv.org/abs/1807.05091

[17] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*, Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener (Eds.). Lecture Notes in Computer Science, Vol. 4052. Springer Berlin Heidelberg, 1–12. https://doi.org/10.1007/11787006_1

[18] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*. Springer, 1–19.

[19] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*. Springer, 265–284.

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

[21] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. ACM, 1054–1067.

[22] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 1322–1333. https://doi.org/10.1145/2810103.2813677

[23] Arik Friedman, Shlomo Berkovsky, and Mohamed Ali Kaafar. 2016. A differential privacy framework for matrix factorization recommender systems. *User Modeling and User-Adapted Interaction* 26, 5 (2016), 425–458.

[24] Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C Pierce. 2013. Linear dependent types for differential privacy. In *ACM SIGPLAN Notices*, Vol. 48. ACM, 357–370.

[25] Samuel Haney, Ashwin Machanavajjhala, John M Abowd, Matthew Graham, Mark Kutzbach, and Lars Vilhuber. 2017. Utility cost of formal privacy for releasing national employer-employee statistics. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 1339–1354.

[26] Noah M. Johnson, Joseph P. Near, and Dawn Xiaodong Song. 2017. Towards Practical Differential Privacy for SQL Queries. *CoRR* abs/1706.09479 (2017). http://arxiv.org/abs/1706.09479

[27] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. 2012. Private convex empirical risk minimization and high-dimensional regression. *Journal of Machine Learning Research* 1 (2012), 41.

[28] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. 2008. Privacy: Theory meets practice on the map. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*. IEEE Computer Society, 277–286.

[29] Frank McSherry and Kunal Talwar. 2007. Mechanism design via differential privacy. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*. IEEE, 94–103.

[30] Frank D McSherry. 2009. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. ACM, 19–30.

[31] Ilya Mironov. 2017. Renyi differential privacy. In *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*. IEEE, 263–275.

[32] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. 2012. GUPT: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. ACM, 349–360.

[33] Arjun Narayan and Andreas Haeberlen. 2012. DJoin: differentially private join queries over distributed databases. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. 149–162.

[34] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755* (2016).

[35] Davide Proserpio, Sharon Goldberg, and Frank McSherry. 2014. Calibrating data to sensitivity in private data analysis: A platform for differentially-private analysis of weighted datasets. *PVLDB* 7, 8 (2014), 637–648.

[36] Jason Reed and Benjamin C Pierce. 2010. Distance makes the types grow stronger: a calculus for differential privacy. *ACM Sigplan Notices* 45, 9 (2010), 157–168.

[37] Indrajit Roy, Srinath TV Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. 2010. Airavat: Security and Privacy for MapReduce.. In *NSDI*, Vol. 10. 297–312.

[38] Tetsuya Sato. 2016. Approximate relational Hoare logic for continuous random samplings. *Electronic Notes in Theoretical Computer Science* 325 (2016), 277–298.

[39] R. Shokri and V. Shmatikov. 2015. Privacy-preserving deep learning. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 909–910. https://doi.org/10.1109/ALLERTON.2015.7447103

[40] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*. 3–18. https://doi.org/10.1109/SP.2017.41

[41] Adam Smith and Abhradeep Thakurta. 2013. Differentially Private Feature Selection via Stability Arguments, and the Robustness of the Lasso. In *COLT*.

[42] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. 2013. Stochastic gradient descent with differentially private updates. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 245–248.

[43] Kunal Talwar, Abhradeep Thakurta, and Li Zhang. 2014. Private Empirical Risk Minimization Beyond the Worst Case: The Effect of the Constraint Set Geometry. *CoRR* abs/1411.5417 (2014).

[44] Kunal Talwar, Abhradeep Guha Thakurta, and Li Zhang. 2015. Nearly optimal private lasso. In *Advances in Neural Information Processing Systems*. 3025–3033.

[45] Philip Wadler. 1990. Linear types can change the world. In *IFIP TC*, Vol. 2. 347–359.

[46] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally Differentially Private Protocols for Frequency Estimation. In *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017.*, Engin Kirda and Thomas Ristenpart (Eds.). USENIX Association, 729–745. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/wang-tianhao

[47] Yu-Xiang Wang, Borja Balle, and Shiva Kasiviswanathan. 2018. Subsampled Rényi Differential Privacy and Analytical Moments Accountant. *CoRR* abs/1808.00087 (2018). arXiv:1808.00087 http://arxiv.org/abs/1808.00087

[48] Stanley L Warner. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *J. Amer. Statist. Assoc.* 60, 309 (1965), 63–69.

[49] Daniel Winograd-Cort, Andreas Haeberlen, Aaron Roth, and Benjamin C Pierce. 2017. A framework for adaptive differential privacy. *Proceedings of the ACM on Programming Languages* 1, ICFP (2017), 10.

[50] X. Wu, M. Fredrikson, S. Jha, and J. F. Naughton. 2016. A Methodology for Formalizing Model-Inversion Attacks. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*. 355–370. https://doi.org/10.1109/CSF.2016.32

[51] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey Naughton. 2017. Bolt-on Differential Privacy for Scalable Stochastic Gradient Descent-based Analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD '17)*. ACM, New York, NY, USA, 1307–1322. https://doi.org/10.1145/3035918.3064047

[52] Danfeng Zhang and Daniel Kifer. 2017. LightDP: Towards automating differential privacy proofs. In *ACM SIGPLAN Notices*, Vol. 52. ACM, 888–901.

A Full Type System and Formalism

Figure 12 shows the full type system. It includes a type-level language of non-negative real-valued symbolic expressions η, in particular including type-level variables β. Type-level variables β (ranging over non-negative reals, not types) are quantified in privacy function types. The function and application expression forms for privacy lambda explicitly introduce and instantiate these quantified type-level variables—they are not inferred through unification in our implementation.

Figure 13 shows type-level metafunctions used in the typing rules. Each of the operations are over types, and return a triple which encodes the resulting sensitivity "cost" of the operation in the sensitivity language. E.g., $s_1, s_2, \tau = \tau_1 + \tau_2$ means that when you add an expression at type τ_1 to an expression at type τ_2, the resulting type is τ_3, the sensitivity of the left argument should be scaled by s_1, and the sensitivity of the second argument should be scaled by s_2. The mod rules is special in that the sensitivities returned are interpreted as max bounds on the arguments, not scaling factors. The main purpose of these metafunctions are to support a form of ad-hoc polymorphism in the type system. E.g., adding two naturals returns a natural, adding two reals returns a real, and adding two statically known numbers returns another statically known number—known to be the sum of the arguments.

Rules for multiplication and mod are particularly important as a non-infinity sensitivity can be given the left operand if the right operand is known statically. Multiplication of a non-statically known value to the inverse of a statically known value is commonly used in our examples to scale a result, resulting in a lower scaled sensitivity cost.

Figure 14 shows the full language definition for sensitivity and privacy languages. From this core language we derive helpers and typecheck all case studies mentioned in the main body of the paper. Included in the sensitivity language are standard linear logic connectives, as described by Fuzz [36]. We also include basic language features like conditionals, and loops—both with static and non-static loop bounds. Note that privacy lambdas include an explicit list of quantified type-level variables β which range over non-negative real-valued expressions. `conv` is a conversion operation that matrices containing data clipped to some norm ℓ to matrices of real numbers, but with sensitivity ℓ.

Figure 15 shows arithmetic metafunctions for sensitivities and privacy annotations. Note that privacy annotations support $+$ but not $\cdot$. The "ceiling" operator is defined in this figure in four forms: sensitivity-to-sensitivity, privacy-to-sensitivity, privacy-to-privacy and sensitivity-to-privacy. We write $\approx$ instead of $=$ because our implementation uses an incomplete (but well-defined) decision procedure for deciding when two real-valued expressions are equal.

Figure 16 shows the kinding system which establishes well-formedness of type variables, symbolic non-negative real-valued expressions, and types. The context Δ is used to track kinds of type-level variables, which are either $\mathbb{N}$ or $\mathbb{R}^+$. We implement a solver for determining when two symbolic real expressions are equal or one is less-than-or-equal to another, and we use the type information in the solver, in addition to merely establishing type well-formedness.

Figure 17 shows the typing rules for the fragment of the sensitivity language that concerns literals and operations over basic primitive types. These typing rules use type-level metafunctions defined in Figure 13.

Figure 18 shows the typing rules for matrix operations, including clipping, gradients, and aggregate operations like maps and folds. We include both unary and binary variants of two different types of maps—element-wise and row-wise.

Figure 19 shows the typing rules for control flow—e.g., loops and ifs—and compound types in the sensitivity language. Compound types include sums, multiplicative products, additive products, sensitivity functions and privacy functions.

Figure 20 shows the typing rules for the entire privacy language. The running theme in each of these rules is to clip contexts up to infinity in the conclusion of rules where no privacy guarantee can be made, and to assume a clipped context up to some fixed value in the assumption of rules where the privacy guarantee requires a bound on sensitivity or privacy on an argument, written in curly brackets. Note that many rules take an explicit list of variables to be considered for the purpose of the assumed bound, written in ascii angle brackets.

Figures 21 and 22 define the metric spaces used to interpret types. We call the model for types a "domain", because we extend the usual notion of metric space with an additional operation which captures how to interpret the norm of values in the underlying set. The distance metric and the norm metric may not be related, and their interpretations do not inter-depend on each other in the semantics.

Figure 23 shows the semantics of types and typing judgments in terms of domains, metrics and norms defined in Figures 21 and 22. Our soundness result is that for well-typed terms there exists an interpretation for terms which inhabits the interpretation of a typing context.

Each of the figures mentioned above are displayed next. After these figures, we present key theorems from the literature which are used in our proof of type soundness, after which we present key cases of the proof. After presenting key proof cases, we re-presented rules which change when adapting the type system to Rényi differential privacy, zero-concentrated differential privacy, and truncated-concentrated differential privacy.

$$
\begin{array}{rll}
n \in \mathbb{N} & & \textit{natural numbers} \\
r \in \mathbb{R} & & \textit{real numbers} \\
r^+ \in \mathbb{R}^+ & & \textit{non-negative real numbers} \\
\beta \in \text{rvar} & & \textit{real-expression variable} \\
x \in \text{var} & & \textit{variables} \\
\kappa \in \text{kind} & ::= \mathbb{N} \mid \mathbb{R}^+ & \textit{real-expression kinds} \\
\Delta \in \text{kcxt} & \triangleq \text{rvar} \rightharpoonup \text{kind} & \textit{kind context} \\
& ::= \{\beta{:}\kappa, ..., \beta{:}\kappa\} & \\
\eta \in \text{rexp} & ::= \beta \mid n \mid r^+ & \textit{var, nat and real} \\
& \mid \eta \sqcup \eta \mid \eta \sqcap \eta & \textit{max and min} \\
& \mid \eta + \eta \mid \eta \cdot \eta & \textit{plus and times} \\
& \mid 1/\eta \mid \sqrt{\eta} \mid \ln \eta & \textit{inverse, root and log} \\
s \in \text{sens} & ::= \eta \mid \infty & \textit{sensitivity} \\
p \in \text{priv} & ::= \eta_\epsilon, \eta_\delta \mid \infty & \textit{privacy cost} \\
\ell \in \text{norm} & ::= L1 \mid L2 \mid L\infty & \textit{norms} \\
c \in \text{clip} & ::= \ell \mid U & \textit{clipping} \\
\tau \in \text{type} & ::= \mathbb{N}[\eta] \mid \mathbb{R}^+[\eta] \mid \mathbb{N} \mid \mathbb{R} \mid \texttt{data} & \textit{numeric types} \\
& \mid \texttt{idx}[\eta] \mid \texttt{matrix}^c_\ell[\eta, \eta]\ \tau & \textit{matrix types} \\
& \mid \tau \uplus \tau \mid \tau \times \tau \mid \tau\ \&\ \tau & \textit{compound types} \\
& \mid \tau \multimap_s \tau \mid \forall\ [\beta{:}\kappa, ..., \beta{:}\kappa]\ (\tau@p, ..., \tau@p) \multimap^* \tau & \textit{function types}
\end{array}
$$

Figure 12. Types and Kinds (shared)

$$
\begin{array}{rl}
\_ \sqcup \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N}[\eta_1] \sqcup \mathbb{N}[\eta_2] & \triangleq \langle 0, 0, \mathbb{N}[\eta_1 \sqcup \eta_2] \rangle \\
\mathbb{R}^+[\eta_1] \sqcup \mathbb{R}^+[\eta_2] & \triangleq \langle 0, 0, \mathbb{R}^+[\eta_1 \sqcup \eta_2] \rangle \\
\mathbb{N} \sqcup \mathbb{N} & \triangleq \langle 1, 1, \mathbb{N} \rangle \\
\mathbb{R} \sqcup \mathbb{R} & \triangleq \langle 1, 1, \mathbb{R} \rangle
\end{array}
\qquad
\begin{array}{rl}
\_ \sqcap \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N}[\eta_1] \sqcap \mathbb{N}[\eta_2] & \triangleq \langle 0, 0, \mathbb{N}[\eta_1 \sqcap \eta_2] \rangle \\
\mathbb{R}^+[\eta_1] \sqcap \mathbb{R}^+[\eta_2] & \triangleq \langle 0, 0, \mathbb{R}^+[\eta_1 \sqcap \eta_2] \rangle \\
\mathbb{N} \sqcap \mathbb{N} & \triangleq \langle 1, 1, \mathbb{N} \rangle \\
\mathbb{R} \sqcap \mathbb{R} & \triangleq \langle 1, 1, \mathbb{R} \rangle
\end{array}
$$

$$
\begin{array}{rl}
\_ + \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N}[\eta_1] + \mathbb{N}[\eta_2] & \triangleq \langle 0, 0, \mathbb{N}[\eta_1 + \eta_2] \rangle \\
\mathbb{R}^+[\eta_1] + \mathbb{R}^+[\eta_2] & \triangleq \langle 0, 0, \mathbb{R}^+[\eta_1 + \eta_2] \rangle \\
\mathbb{N} + \mathbb{N} & \triangleq \langle 1, 1, \mathbb{N} \rangle \\
\mathbb{R} + \mathbb{R} & \triangleq \langle 1, 1, \mathbb{R} \rangle
\end{array}
\qquad
\begin{array}{rl}
\_ \cdot \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N}[\eta_1] \cdot \mathbb{N}[\eta_2] & \triangleq \langle 0\ , 0\ , \mathbb{N}[\eta_1 \cdot \eta_2] \rangle \\
\mathbb{R}^+[\eta_1] \cdot \mathbb{R}^+[\eta_2] & \triangleq \langle 0\ , 0\ , \mathbb{R}^+[\eta_1 \cdot \eta_2] \rangle \\
\mathbb{N}[\eta_1] \cdot \mathbb{N} & \triangleq \langle 0\ , \eta_1, \mathbb{N} \rangle \\
\mathbb{R}^+[\eta_1] \cdot \mathbb{R} & \triangleq \langle 0\ , \eta_1, \mathbb{R} \rangle \\
\mathbb{N} \cdot \mathbb{N}[\eta_2] & \triangleq \langle \eta_2, 0\ , \mathbb{N} \rangle \\
\mathbb{R} \cdot \mathbb{R}^+[\eta_2] & \triangleq \langle \eta_2, 0\ , \mathbb{R} \rangle \\
\mathbb{N} \cdot \mathbb{N} & \triangleq \langle \infty, \infty, \mathbb{N} \rangle \\
\mathbb{R} \cdot \mathbb{R} & \triangleq \langle \infty, \infty, \mathbb{R} \rangle
\end{array}
\qquad
\begin{array}{rl}
1/\_ & \in \tau \rightharpoonup \text{sens} \times \tau \\
1/\mathbb{R}^+[\eta] & \triangleq \langle 0\ , \mathbb{R}^+[1/\eta] \rangle \\
1/\mathbb{R} & \triangleq \langle \infty, \mathbb{R} \rangle
\end{array}
$$

$$
\begin{array}{rl}
\sqrt{\_} & \in \tau \rightharpoonup \text{sens} \times \tau \\
\sqrt{\mathbb{R}^+[\eta]} & \triangleq \langle 0\ , \mathbb{R}^+[\sqrt{\eta}] \rangle \\
\sqrt{\mathbb{R}} & \triangleq \langle \infty, \mathbb{R} \rangle
\end{array}
\qquad
\begin{array}{rl}
\ln \_ & \in \tau \rightharpoonup \text{sens} \times \tau \\
\ln \mathbb{R}^+[\eta] & \triangleq \langle 0\ , \mathbb{R}^+[\ln \eta] \rangle \\
\ln \mathbb{R} & \triangleq \langle \infty, \mathbb{R} \rangle
\end{array}
$$

$$
\begin{array}{rl}
\_ \bmod \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N} \bmod \mathbb{N}[\eta_2] & \triangleq \langle \eta_2, 0\ , \mathbb{N} \rangle \\
\texttt{idx}[\eta_1] \bmod \texttt{idx}[\eta_2] & \triangleq \langle \eta_2, \eta_1, \texttt{idx}[\eta_1 \sqcap \eta_2] \rangle \\
\mathbb{N} \bmod \mathbb{N} & \triangleq \langle \infty, \infty, \mathbb{N} \rangle
\end{array}
$$

$$
\begin{array}{rl}
\_ - \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N} - \mathbb{N} & \triangleq \langle 1, 1, \mathbb{N} \rangle \\
\mathbb{R} - \mathbb{R} & \triangleq \langle 1, 1, \mathbb{R} \rangle
\end{array}
\qquad
\begin{array}{rl}
\_ \dot{-} \_ & \in \tau \times \tau \rightharpoonup \text{sens} \times \text{sens} \times \tau \\
\mathbb{N} \dot{-} \mathbb{N} & \triangleq \langle 1, 1, \mathbb{N} \rangle \\
\mathbb{R} \dot{-} \mathbb{R} & \triangleq \langle 1, 1, \mathbb{N} \rangle
\end{array}
$$

Figure 13. Type Metafunctions

$g \in \text{grad}$	$::= LS \mid LR \mid GR \mid HSVM$	*gradients*[1]
$e \in \text{exp}$	$::= \mathbb{N}[n] \mid \mathbb{N}[r^+] \mid \texttt{dyn}\ e \mid n \mid r \mid \texttt{real}\ e$	*naturals and reals*
	$\mid e \sqcup e \mid e \sqcap e \mid e + e \mid e \cdot e \mid 1/e$	*arithmetic*
	$\mid \sqrt{e} \mid \ln e \mid e\ \texttt{mod}\ e \mid e - e \mid e \doteq e$	*arithmetic*
	$\mid \texttt{mcreate}_\ell[e, e]\{x, x \Rightarrow e\} \mid e\#e$	*matrix creation and index*
	$\mid \texttt{rows}\ e \mid \texttt{cols}\ e$	*matrix transpose and lengths*
	$\mid \texttt{clip}^\ell\ e \mid \texttt{conv}\ e$	*clipping and conversion*
	$\mid L\nabla^g_\ell[e; e, e] \mid U\nabla[e; e, e]$	*gradient*
	$\mid \texttt{mmap}\ e\ \{x \Rightarrow e\}$	*matrix map*
	$\mid \texttt{mmap}\ e, e\ \{x, x \Rightarrow e\}$	*matrix binary map*
	$\mid \texttt{mmap-row}\ e\ \{x \Rightarrow e\}$	*matrix map rows*
	$\mid \texttt{mmap-row}\ e, e\ \{x, x \Rightarrow e\}$	*matrix binary map rows*
	$\mid \texttt{mfold-row}\ e\ \texttt{on}\ e\ \{x, x \Rightarrow e\}$	*fold row*
	$\mid \texttt{if}\ e\ \{e\}\ \{e\}$	*conditional*
	$\mid \texttt{sloop}\ e\ \texttt{on}\ e\ \{x, x \Rightarrow e\}$	*static finite iteration*
	$\mid \texttt{loop}\ e\ \texttt{on}\ e\ \{x, x \Rightarrow e\}$	*finite iteration*
	$\mid x \mid \texttt{let}\ x = e\ \texttt{in}\ e \mid \lambda\ x{:}\tau \Rightarrow e \mid e\ e$	*variables and functions*
	$\mid p\lambda\ [\beta{:}\kappa, ..., \beta{:}\kappa]\ (x{:}\tau, ..., x{:}\tau) \Rightarrow e$	*privacy function*
	$\mid \texttt{inl}[\tau]\ e \mid \texttt{inr}[\tau]\ e \mid \texttt{case}\ e\ \{x \Rightarrow e\}\ \{x \Rightarrow e\}$	*sums*
	$\mid \langle e, e \rangle \mid \texttt{let}\ x, x = e\ \texttt{in}\ e$	*mult. products*
	$\mid \langle e, e \rangle \mid \texttt{prl}\ e \mid \texttt{prr}\ e$	*add. products*
$\Gamma_s \in \text{tcxt}_s$	$\triangleq \text{var} \rightharpoonup \text{sens} \times \text{type}$	*type contexts*
	$::= \{x{:}_s\tau, ..., x{:}_s\tau\}$	

$e \in \text{exp}$	$::= \texttt{return}\ e \mid x \leftarrow e\ ;\ e \mid e[\eta, ..., \eta](x, ..., x)$	*ret, bind and apply*
	$\mid \texttt{sloop}[e]\ e\ \texttt{on}\ e\ \texttt{<}x, ..., x\texttt{>}\ \{x, x \Rightarrow e\}$	*finite iteration*
	$\mid \texttt{gauss}[e, e, e]\ \texttt{<}x, ..., x\texttt{>}\ \{e\}$	*gaussian noise*
	$\mid \texttt{mgauss}[e, e, e]\ \texttt{<}x, ..., x\texttt{>}\ \{e\}$	*gaussian noise for matrices*
	$\mid \texttt{laplace}[e, e]\ \texttt{<}x, ..., x\texttt{>}\ \{e\}$	*laplacean noise*
	$\mid \texttt{exponential}[e, e]\ e\ \texttt{<}x, ..., x\texttt{>}\ \{x \Rightarrow e\}$	*exponential mechanism*
	$\mid \texttt{rand-resp}[e, e]\ \texttt{<}x, ..., x\texttt{>}\ \{e\}$	*randomized response*
	$\mid \texttt{sample}\ e\ \texttt{on}\ x, x\ \{x, x \Rightarrow e\}$	*sampling*
	$\mid \texttt{rand-nat}(e, e)$	*random natural*
$\Gamma_p \in \text{tcxt}_p$	$\triangleq \text{var} \rightharpoonup \text{priv} \times \text{type}$	*type contexts*
	$::= \{x{:}_p\tau, ..., x{:}_p\tau\}$	

Figure 14. Sensitivity and Privacy Languages

$$
\begin{aligned}
\_ + \_ &\in (\text{sens} \times \text{sens} \to \text{sens}) \\
&\uplus (\text{tcxt}_s \times \text{tcxt}_s \to \text{tcxt}_s) \\
\eta_1 + \eta_2 &\triangleq \eta_1 + \eta_2 \\
\infty + s_2 &\triangleq \infty \\
s_1 + \infty &\triangleq \infty \\
(\Gamma_1 + \Gamma_2)(x) &\triangleq \Gamma_1(x) + \Gamma_2(x)
\end{aligned}
$$

$$
\begin{aligned}
\_ \cdot \_ &\in (\text{sens} \times \text{sens} \to \text{sens}) \\
&\uplus (\text{tcxt}_s \times \text{tcxt}_s \to \text{tcxt}_s) \\
\eta_1 \cdot \eta_2 &\triangleq \eta_1 \eta_2 \\
\infty \cdot s_2 &\triangleq \begin{cases} 0 & \text{if} \quad s_2 \approx 0 \\ \infty & \text{if} \quad s_2 \not\approx 0 \end{cases} \\
s_1 \cdot \infty &\triangleq \begin{cases} 0 & \text{if} \quad s_1 \approx 0 \\ \infty & \text{if} \quad s_1 \not\approx 0 \end{cases} \\
(\Gamma_1 \cdot \Gamma_2)(x) &\triangleq \Gamma_1(x) \cdot \Gamma_2(x)
\end{aligned}
$$

$$
\begin{aligned}
\_ + \_ &\in (\text{priv} \times \text{priv} \to \text{priv}) \\
&\uplus (\text{tcxt}_p \times \text{tcxt}_p \to \text{tcxt}_p) \\
(\eta_{\epsilon_1}, \eta_{\delta_1}) + (\eta_{\epsilon_2}, \eta_{\delta_2}) &\triangleq \eta_{\epsilon_1} + \eta_{\epsilon_2}, \eta_{\delta_1} + \eta_{\delta_2} \\
\infty + p_2 &\triangleq \infty \\
p_1 + \infty &\triangleq \infty \\
(\Gamma_1 + \Gamma_2)(x) &\triangleq \Gamma_1(x) + \Gamma_2(x)
\end{aligned}
$$

$$
\begin{aligned}
\rceil \_ \lceil - &\in (\text{sens} \times \text{sens} \to \text{sens}) \\
&\uplus (\text{tcxt}_s \times \text{sens} \to \text{tcxt}_s) \\
\rceil \eta \lceil^s &\triangleq \begin{cases} 0 & \text{if} \quad \eta \approx 0 \\ s & \text{if} \quad \eta \not\approx 0 \end{cases} \\
\rceil \infty \lceil^s &\triangleq s \\
\rceil \Gamma \lceil^s(x) &\triangleq \rceil \Gamma(x) \lceil^s
\end{aligned}
$$

$$
\begin{aligned}
\rceil \_ \lceil - &\in (\text{priv} \times \text{sens} \to \text{sens}) \\
&\uplus (\text{tcxt}_p \times \text{sens} \to \text{tcxt}_s) \\
\rceil \eta_\epsilon, \eta_\delta \lceil^s &\triangleq \begin{cases} 0 & \text{if} \quad \eta_\epsilon \approx 0 \wedge \eta_\delta \approx 0 \\ s & \text{if} \quad \eta_\epsilon \not\approx 0 \vee \eta_\delta \not\approx 0 \end{cases} \\
\rceil \infty \lceil^s &\triangleq s \\
\rceil \Gamma \lceil^s(x) &\triangleq \rceil \Gamma(x) \lceil^s
\end{aligned}
$$

$$
\begin{aligned}
\rceil \_ \lceil - &\in (\text{priv} \times \text{priv} \to \text{priv}) \\
&\uplus (\text{tcxt}_p \times \text{priv} \to \text{tcxt}_p) \\
\rceil \eta_\epsilon, \eta_\delta \lceil^p &\triangleq \begin{cases} 0, 0 & \text{if} \quad \eta_\epsilon \approx 0 \wedge \eta_\delta \approx 0 \\ p & \text{if} \quad \eta_\epsilon \not\approx 0 \vee \eta_\delta \not\approx 0 \end{cases} \\
\rceil \infty \lceil^p &\triangleq p \\
\rceil \Gamma \lceil^p(x) &\triangleq \rceil \Gamma(x) \lceil^p
\end{aligned}
$$

$$
\begin{aligned}
\rceil \_ \lceil - &\in (\text{sens} \times \text{priv} \to \text{priv}) \\
&\uplus (\text{tcxt}_s \times \text{priv} \to \text{tcxt}_p) \\
\rceil \eta \lceil^p &\triangleq \begin{cases} 0, 0 & \text{if} \quad \eta \approx 0 \\ p & \text{if} \quad \eta \not\approx 0 \end{cases} \\
\rceil \infty \lceil^p &\triangleq p \\
\rceil \Gamma \lceil^p(x) &\triangleq \rceil \Gamma(x) \lceil^p
\end{aligned}
$$

Figure 15. Sensitivity and Privacy Metafunctions

$\boxed{\Delta \vdash \eta : \kappa}$

$$\textsc{RVar}\ \frac{\beta{:}\kappa \in \Delta}{\Delta \vdash \beta : \kappa} \qquad \textsc{Nat}\ \frac{}{\Delta \vdash n : \mathbb{N}} \qquad \textsc{Real}\ \frac{}{\Delta \vdash r^+ : \mathbb{R}^+}$$

$$\textsc{Max}\ \frac{\Delta \vdash \eta_1 : \kappa \qquad \Delta \vdash \eta_2 : \kappa}{\Delta \vdash \eta_1 \sqcup \eta_2 : \kappa} \qquad \textsc{Min}\ \frac{\Delta \vdash \eta_1 : \kappa \qquad \Delta \vdash \eta_2 : \kappa}{\Delta \vdash \eta_1 \sqcap \eta_2 : \kappa}$$

$$\textsc{Plus}\ \frac{\Delta \vdash \eta_1 : \kappa \qquad \Delta \vdash \eta_2 : \kappa}{\Delta \vdash \eta_1 + \eta_2 : \kappa} \qquad \textsc{Times}\ \frac{\Delta \vdash \eta_1 : \kappa \qquad \Delta \vdash \eta_2 : \kappa}{\Delta \vdash \eta_1 \cdot \eta_2 : \kappa}$$

$$\textsc{Div}\ \frac{\Delta \vdash \eta_1 : \mathbb{R}^+ \qquad \Delta \vdash \eta_2 : \mathbb{R}^+}{\Delta \vdash \eta_1 / \eta_2 : \mathbb{R}^+} \qquad \textsc{Root}\ \frac{\Delta \vdash \eta : \mathbb{R}^+}{\Delta \vdash \sqrt{\eta} : \mathbb{R}^+} \qquad \textsc{Log}\ \frac{\Delta \vdash \eta : \mathbb{R}^+}{\Delta \vdash \ln \eta : \mathbb{R}^+}$$

$$\textsc{Weaken}\ \frac{\Delta \vdash \eta : \mathbb{N}}{\Delta \vdash \eta : \mathbb{R}^+}$$

$\boxed{\Delta \vdash s}$

$$\textsc{Sens}\ \frac{\Delta \vdash \eta : \mathbb{R}^+}{\Delta \vdash \eta} \qquad \textsc{Inf}\ \frac{}{\Delta \vdash \infty}$$

$\boxed{\Delta \vdash p}$

$$\textsc{Sens}\ \frac{\Delta \vdash \eta_\epsilon : \mathbb{R}^+ \qquad \Delta \vdash \eta_\delta : \mathbb{R}^+}{\Delta \vdash \eta_\epsilon, \eta_\delta} \qquad \textsc{Inf}\ \frac{}{\Delta \vdash \infty}$$

$\boxed{\Delta \vdash \tau}$

$$\textsc{Static Nat}\ \frac{\Delta \vdash \eta : \mathbb{N}}{\Delta \vdash \mathbb{N}[\eta]} \qquad \textsc{Static Real}\ \frac{\Delta \vdash \eta : \mathbb{R}^+}{\Delta \vdash \mathbb{R}^+[\eta]} \qquad \textsc{Nat}\ \frac{}{\Delta \vdash \mathbb{N}} \qquad \textsc{Real}\ \frac{}{\Delta \vdash \mathbb{R}} \qquad \textsc{Data}\ \frac{}{\Delta \vdash \texttt{data}}$$

$$\textsc{Index}\ \frac{\Delta \vdash \eta : \mathbb{N}}{\Delta \vdash \texttt{idx}[\eta]} \qquad \textsc{Matrix}\ \frac{\Delta \vdash \eta_m : \mathbb{N} \qquad \Delta \vdash \eta_n : \mathbb{N} \qquad \Delta \vdash \tau}{\Delta \vdash \texttt{matrix}^c_\ell[\eta_m, \eta_n]\ \tau}$$

$$+\text{-F}\ \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 + \tau_2} \qquad \times\text{-F}\ \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 \times \tau_2} \qquad \&\text{-F}\ \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1\ \&\ \tau_2} \qquad \multimap\text{-F}\ \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2 \qquad \Delta \vdash s}{\Delta \vdash \tau_1 \multimap_s \tau_2}$$

$$\multimap^*\text{-F}\ \frac{\Delta' = \Delta \uplus \{\beta_1{:}\kappa_1, \ldots, \beta_n{:}\kappa_n\} \qquad \Delta' \vdash \tau_i\ (\forall i) \qquad \Delta' \vdash p_i\ (\forall i) \qquad \Delta' \vdash \tau}{\Delta \vdash \forall\ [\beta_1{:}\kappa_1, \ldots, \beta_n{:}\kappa_n]\ (\tau_1@p_1, \ldots, \tau_n@p_n) \multimap^* \tau}$$

Figure 16. Kinding and Type Formation

$\boxed{\Delta, \Gamma \vdash e : \tau}$

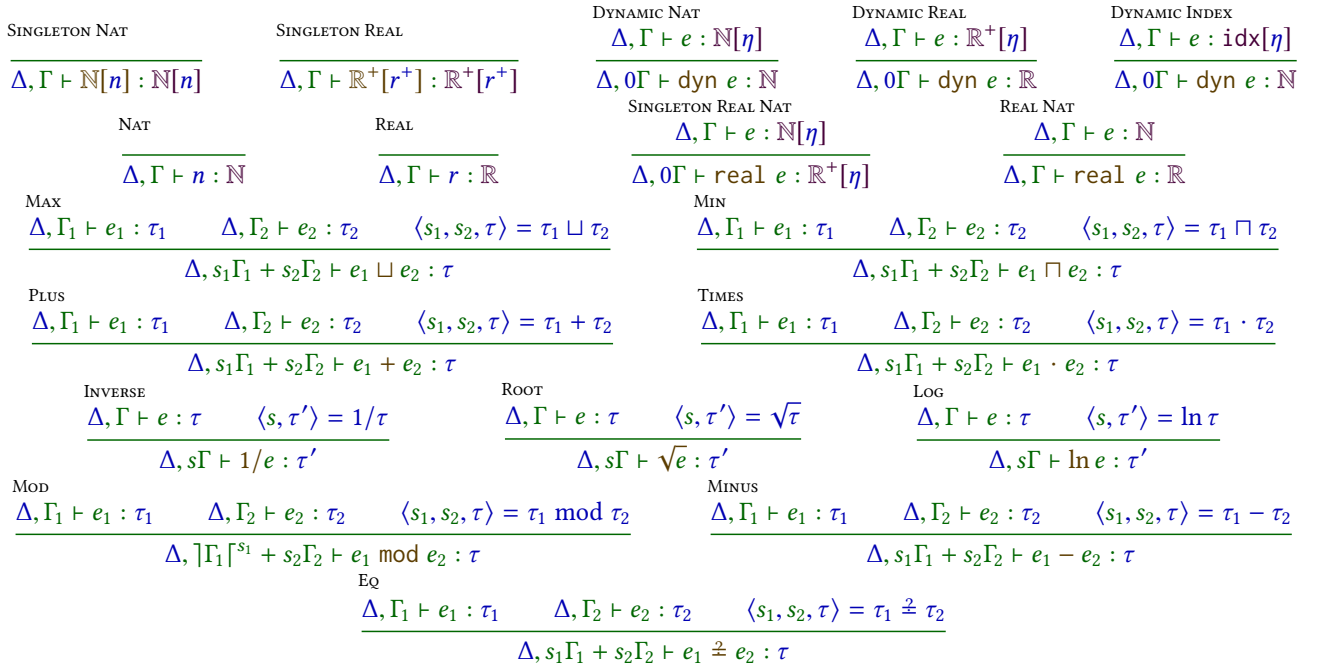

Figure 17. Sensitivity Typing—Literals and Arithmetic

$\boxed{\Delta, \Gamma \vdash e : \tau}$

MATRIX CREATION

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{N}[\eta_m] \qquad \Delta, \Gamma_2 \vdash e_2 : \mathbb{N}[\eta_n] \qquad \Delta, \Gamma_3 \uplus \{x_1 :_\infty \mathtt{idx}[\eta_m], x_2 :_\infty \mathtt{idx}[\eta_n]\} \vdash e_3 : \tau}{\Delta, 0(\Gamma_1 + \Gamma_2) + \eta_m \eta_n \Gamma_3 : \mathtt{mcreate}_\ell[e_1, e_2]\{x_1, x_2 \Rightarrow e_3\} : \mathtt{matrix}^U_\ell[\eta_m, \eta_n]\ \tau}$$

MATRIX INDEX

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^c_\ell[\eta_m, \eta_n]\ \tau \qquad \Delta, \Gamma_2 \vdash e_2 : \mathtt{idx}[\eta_m]}{\Delta, \Gamma_1 + 0\Gamma_2 \vdash e_1 \# e_2 : \mathtt{matrix}^c_\ell[1, \eta_n]\ \tau}$$

MATRIX ROWS

$$\frac{\Delta, \Gamma \vdash e : \mathtt{matrix}^\star_\star[\eta_m, \eta_n]\ \tau}{\Delta, 0\Gamma \vdash \mathtt{rows}\ e : \mathbb{N}[\eta_m]}$$

MATRIX COLS

$$\frac{\Delta, \Gamma \vdash e : \mathtt{matrix}^\star_\star[\eta_m, \eta_n]\ \tau}{\Delta, 0\Gamma \vdash \mathtt{cols}\ e : \mathbb{N}[\eta_n]}$$

CLIP

(this could work for R, but we never need it at that type for examples)

$$\frac{\Delta, \Gamma \vdash e : \mathtt{matrix}^\star_\ell[1, \eta_n]\ \mathtt{data}}{\Delta, \Gamma \vdash \mathtt{clip}^c\ e : \mathtt{matrix}^c_\ell[1, \eta_n]\ \mathtt{data}}$$

CONVERT

$$\frac{\Delta, \Gamma \vdash e : \mathtt{matrix}^\ell_\star[1, \eta_n]\ \mathtt{data}}{\Delta, \Gamma \vdash \mathtt{conv}\ e : \mathtt{matrix}^U_\ell[1, \eta_n]\ \mathbb{R}}$$

LIPSCHITZ GRADIENT

$$\frac{\Delta, \Gamma_1 \vdash e_\theta : \mathtt{matrix}^\star_\star[1, \eta_n]\ \mathbb{R} \qquad \Delta, \Gamma_2 \vdash e_{xs} : \mathtt{matrix}^\ell_\star[1, \eta_n]\ \mathtt{data} \qquad \Delta, \Gamma_3 \vdash e_y : \mathtt{data}}{\Delta, \infty\Gamma_1 + \Gamma_2 + \Gamma_3 \vdash L\nabla^g_\ell[e_\theta; e_{xs}, e_{ys}] : \mathtt{matrix}^U_\ell[1, \eta_n]\ \mathbb{R}}$$

UNBOUNDED GRADIENT

$$\frac{\Delta, \Gamma_1 \vdash e_\theta : \mathtt{matrix}^\star_\star[1, \eta_n]\ \mathbb{R} \qquad \Delta, \Gamma_2 \vdash e_{xs} : \mathtt{matrix}^\star_{L\infty}[1, \eta_n]\ \mathtt{data} \qquad \Delta, \Gamma_3 \vdash e_y : \mathtt{data}}{\Delta, \infty\Gamma_1 + \Gamma_2 + \Gamma_3 \vdash U\nabla[e_\theta; e_{xs}, e_y] : \mathtt{matrix}^U_{L\infty}[1, \eta_n]\ \mathtt{data}}$$

MATRIX MAP

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^\star_\ell[\eta_m, \eta_n]\ \tau_1 \qquad \Delta, \Gamma_2 \uplus \{x :_s \tau_1\} \vdash e_2 : \tau_2}{\Delta, s\Gamma_1 + \eta_m \eta_n \Gamma_2 \vdash \mathtt{mmap}\ e_1\ \{x \Rightarrow e_2\} : \mathtt{matrix}^U_\ell[\eta_m, \eta_n]\ \tau_2}$$

MATRIX BINARY MAP

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^\star_\ell[\eta_m, \eta_n]\ \tau_1 \qquad \Delta, \Gamma_2 \vdash e_2 : \mathtt{matrix}^\star_\ell[\eta_m, \eta_n]\ \tau_2 \qquad \Delta, \Gamma_3 \uplus \{x_1 :_{s_1} \tau_1, x_2 :_{s_2} \tau_2\} \vdash e_3 : \tau_3}{\Delta, s_1\Gamma_1 + s_2\Gamma_2 + \eta_m \eta_n \Gamma_3 \vdash \mathtt{mmap}\ e_1, e_2\ \{x_1, x_2 \Rightarrow e_3\} : \mathtt{matrix}^U_\ell[\eta_m, \eta_n]\ \tau_3}$$

MATRIX MAP ROWS

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^{c_1}_{\ell_1}[\eta_m, \eta_{n_1}]\ \tau_1 \qquad \Delta, \Gamma_2 \uplus \{x :_s \mathtt{matrix}^{c_1}_{\ell_1}[1, \eta_{n_1}]\ \tau_1\} \vdash e_2 : \mathtt{matrix}^{c_2}_{\ell_2}[1, \eta_{n_2}]\ \tau_2}{\Delta, s\Gamma_1 + \eta_m \Gamma_2 \vdash \mathtt{mmap\text{-}row}\ e_1\ \{x \Rightarrow e_2\} : \mathtt{matrix}^{c_2}_{\ell_2}[\eta_m, \eta_{n_2}]\ \tau_2}$$

MATRIX BINARY MAP ROWS

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^{c_1}_{\ell_1}[\eta_m, \eta_{n_1}]\ \tau_1 \qquad \Delta, \Gamma_2 \vdash e_2 : \mathtt{matrix}^{c_2}_{\ell_2}[\eta_m, \eta_{n_2}]\ \tau_2 \qquad \Delta, \Gamma_3 \uplus \{x_1 :_{s_1} \mathtt{matrix}^{c_1}_{\ell_1}[1, \eta_{n_1}]\ \tau_1, x_2 :_{s_2} \mathtt{matrix}^{c_2}_{\ell_2}[1, \eta_{n_2}]\ \tau_2\} \vdash e_3 : \mathtt{matrix}^{c_3}_{\ell_3}[1, \eta_{n_3}]\ \tau_3}{\Delta, s_1\Gamma_1 + s_2\Gamma_2 + \eta_m\Gamma_3 \vdash \mathtt{mmap\text{-}row}\ e_1, e_2\ \{x_1, x_2 \Rightarrow e_3\} : \mathtt{matrix}^{c_3}_{\ell_3}[\eta_m, \eta_{n_3}]\ \tau_3}$$

MATRIX FOLD ROWS

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathtt{matrix}^c_\ell[\eta_m, \eta_n]\ \tau_1 \qquad \Delta, \Gamma_2 \vdash e_2 : \tau_2 \qquad \Delta, \Gamma_3 \uplus \{x :_{s_1} \tau_1, y :_{s_2} \tau_2\} \vdash e_3 : \tau_2}{\Delta, s_1\Gamma_1 + s_2^{\eta_m}\Gamma_2 + \eta_m\Gamma_3 \vdash \mathtt{mfold\text{-}row}\ e_1\ \mathtt{on}\ e_2\ \{x, y \Rightarrow e_3\} : \mathtt{matrix}^c_\ell[1, \eta_n]\ \tau_2}$$

Figure 18. Sensitivity Typing—Matrices

$\boxed{\Delta, \Gamma \vdash e : \tau}$

$$\textsc{If}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{N} \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 \vdash e_3 : \tau}{\Delta, \infty\Gamma_1 + \Gamma_2 + \Gamma_3 \vdash \texttt{if}\ e_1\ \{e_2\}\ \{e_3\} : \tau}$$

$$\textsc{Static Loop}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{N}[\eta] \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 \uplus \{x_1 :_\infty \texttt{idx}[\eta], x_2 :_s \tau\} \vdash e_3 : \tau}{\Delta, 0\Gamma_1 + s^\eta \Gamma_2 + \eta\Gamma_3 \vdash \texttt{sloop}\ e_1\ \texttt{on}\ e_2\ \{x_1, x_2 \Rightarrow e_3\} : \tau}$$

$$\textsc{Loop}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{N} \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_1 \tau\} \vdash e_3 : \tau}{\Delta, \Gamma_1 + \Gamma_2 + \infty\Gamma_3 \vdash \texttt{loop}\ e_1\ \texttt{on}\ e_2\ \{x_1, x_2 \Rightarrow e_3\} : \tau}$$

$$\textsc{Var}\ \frac{}{\Delta, \Gamma \uplus \{x :_1 \tau\} \vdash x : \tau}$$

$$\textsc{Let}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \qquad \Delta, \Gamma_2 \uplus \{x :_s \tau_1\} \vdash e_2 : \tau_2}{\Delta, s\Gamma_1 + \Gamma_2 \vdash \texttt{let}\ x = e_1\ \texttt{in}\ e_2 : \tau_2}$$

$$\multimap\text{-I}\ \frac{\Delta \vdash \tau_1 \qquad \Delta, \Gamma \uplus \{x :_s \tau_1\} \vdash e : \tau_2}{\Delta, \Gamma \vdash (\lambda\ x{:}\tau_1 \Rightarrow e) : \tau_1 \multimap_s \tau_2}$$

$$\multimap\text{-E}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \multimap_s \tau_2 \qquad \Delta, \Gamma_2 \vdash e_2 : \tau_1}{\Delta, \Gamma_1 + s\Gamma_2 \vdash (e_1\ e_2) : \tau_2}$$

$$\multimap^*\text{-I}\ \frac{\Delta' = \Delta \uplus \{\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n\} \qquad \Delta' \vdash \tau_i\ (\forall i) \qquad \Delta', \Gamma \uplus \{x_1, :_{p_1} \tau_1, ..., x_n{:}_{p_n}\tau_n\} \vdash e : \tau}{\Delta, \rceil\Gamma\lceil^\infty \vdash (p\lambda\ [\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n]\ (x_1{:}\tau_1, ..., x_n{:}\tau_n) \Rightarrow e) : \forall\ [\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n]\ (\tau_1@p_1, ..., \tau_n@p_n) \multimap^* \tau}$$

$$\uplus\text{-I-1}\ \frac{\Delta \vdash \tau_2 \qquad \Delta, \Gamma \vdash e : \tau_1}{\Delta, \Gamma \vdash \texttt{inl}[\tau_2]\ e : \tau_1 \uplus \tau_2} \qquad \uplus\text{-I-2}\ \frac{\Delta \vdash \tau_1 \qquad \Delta, \Gamma \vdash e : \tau_2}{\Delta, \Gamma \vdash \texttt{inr}[\tau_1]\ e : \tau_1 \uplus \tau_2}$$

$$\uplus\text{-E}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \uplus \tau_2 \qquad \Delta, \Gamma_2 \uplus \{x_1 :_s \tau_1\} \vdash e_2 : \tau_3 \qquad \Delta, \Gamma_2 \uplus \{x_2 :_s \tau_2\} \vdash e_3 : \tau_3}{\Delta, s\Gamma_1 + \Gamma_2 \vdash \texttt{case}\ e_1\ \{x_1 \Rightarrow e_2\}\ \{x_2 \Rightarrow e_3\} : \tau_3}$$

$$\&\text{-I}\ \frac{\Delta, \Gamma \vdash e_1 : \tau_1 \qquad \Delta, \Gamma \vdash e_2 : \tau_2}{\Delta, \Gamma \vdash \langle e_1, e_2 \rangle : \tau_1\ \&\ \tau_2}$$

$$\&\text{-E-1}\ \frac{\Delta, \Gamma \vdash e : \tau_1\ \&\ \tau_2}{\Delta, \Gamma \vdash \texttt{prl}\ e : \tau_1} \qquad \&\text{-E-2}\ \frac{\Delta, \Gamma \vdash e : \tau_1\ \&\ \tau_2}{\Delta, \Gamma \vdash \texttt{prr}\ e : \tau_2}$$

$$\times\text{-I}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \qquad \Delta, \Gamma_2 \vdash e_2 : \tau_2}{\Delta, \Gamma_1 + \Gamma_2 \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2}$$

$$\times\text{-E}\ \frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \times \tau_2 \qquad \Delta, \Gamma_2 \uplus \{x_1 :_s \tau_1, x_2 :_s \tau_2\} \vdash e_2 : \tau_3}{\Delta, s\Gamma_1 + \Gamma_2 \vdash \texttt{let}\ x_1, x_2 = e_1\ \texttt{in}\ e_2 : \tau_3}$$

Figure 19. Sensitivity Typing—Iteration and Connectives

$\boxed{\Delta, \Gamma \vdash e : \tau}$

RETURN

$$\frac{\Delta, \Gamma \vdash e : \tau}{\Delta, \lceil \Gamma \rceil^{\infty} \vdash \texttt{return}\ e : \tau}$$

BIND

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \tau_1 \qquad \Delta, \Gamma_2 \uplus \{x :_{\infty} \tau_1\} \vdash e_2 : \tau_2}{\Delta, \Gamma_1 + \Gamma_2 \vdash x \leftarrow e_1 \ ; \ e_2 : \tau_2}$$

$\multimap^*$-E

$$\frac{\Delta \vdash \eta_i : \kappa_i \ (\forall i) \qquad \Delta, \Gamma \vdash e : \forall\ [\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n]\ (\tau_1 @ p_1, ..., \tau_n @ p_n) \multimap^* \tau}{\Delta, \lceil \Gamma \rceil^{\infty} + \{x_1 :_{p_1} \tau_1, ..., x_n :_{p_n} \tau_n\} \vdash e[\eta_1, ..., \eta_n](x_1, ..., x_n) : \tau[\eta_i / \beta_i]}$$

STATIC LOOP *(ADVANCED COMPOSITION)*

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_{\delta'}] \qquad \Delta, \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_n] \qquad \Delta, \Gamma_3 \vdash e_3 : \tau \qquad \Delta, \Gamma_4 + \lfloor \Gamma_5 \rfloor^{\eta_\epsilon, \eta_\delta}_{\{x'_1, ..., x'_n\}} \uplus \{x_1 :_{\infty} \mathbb{N}, x_2 :_{\infty} \tau\} \vdash e_4 : \tau}{\Delta, \lceil \Gamma_1 + \Gamma_2 \rceil^{0,0} + \lceil \Gamma_3 \rceil^{\infty} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rfloor^{2 \cdot \eta_\epsilon \sqrt{2 \cdot \eta_n \ln(1/\eta_{\delta'})}, \eta_{\delta'} + \eta_n \eta_\delta}_{\{x'_1, ..., x'_n\}} \vdash \texttt{loop}[e_1]\ e_2\ \texttt{on}\ e_3\ \texttt{<}x'_1, ..., x'_n\texttt{>}\ \{x_1, x_2 \Rightarrow e_4\} : \tau}$$

GAUSS

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\delta] \qquad \Gamma_4 + \lfloor \Gamma_5 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \vdash e_4 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rfloor^{\eta_\epsilon, \eta_\delta}_{\{x_1, ..., x_n\}} \vdash \texttt{gauss}[e_1, e_2, e_3]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_4\} : \mathbb{R}}$$

MGAUSS

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\delta] \qquad \Gamma_4 + \lfloor \Gamma_5 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \vdash e_4 : \texttt{matrix}^{\bigstar}_{L2}[\eta_m, \eta_n]\ \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rfloor^{\eta_\epsilon, \eta_\delta}_{\{x_1, ..., x_n\}} \vdash \texttt{mgauss}[e_1, e_2, e_3]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_4\} : \texttt{matrix}^{U}_{L\infty}[\eta_m, \eta_n]\ \mathbb{R}}$$

LAPLACE

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_3 + \lfloor \Gamma_4 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \vdash e_3 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{0,0} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\eta_\epsilon, 0}_{\{x_1, ..., x_n\}} \vdash \texttt{laplace}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_3\} : \mathbb{R}}$$

EXPONENTIAL

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_3 \vdash e_3 : \texttt{matrix}^{\bigstar}_{\bigstar}[1, \eta_m](\tau) \qquad \Gamma_4 + \lfloor \Gamma_5 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \uplus \{x :_{\infty} \tau\} \vdash e_4 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{0,0} + \lceil \Gamma_3 + \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rfloor^{\eta_\epsilon, 0}_{\{x_1, ..., x_n\}} \vdash \texttt{exponential}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ e_3\ \{x \Rightarrow e_4\} : \tau}$$

RAND-RESP

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{N}[\eta_n] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_3 + \lfloor \Gamma_4 \rfloor^{\eta_n}_{\{x_1, ..., x_n\}} \vdash e_3 : \mathbb{N}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{0,0} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\eta_\epsilon, 0}_{\{x_1, ..., x_n\}} \vdash \texttt{rand-resp}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_3\} : \mathbb{N}}$$

SAMPLE

$$\frac{\begin{array}{l}\Delta, \Gamma_1 \vdash e_1 : \mathbb{N}[\eta_{m_2}] \qquad \eta_{m_2} \le \eta_{m_1} \qquad \eta_{\epsilon'} = 2 \cdot \eta_{m_2} \cdot 1/\eta_{m_1} \qquad \eta_{\delta'} = \eta_{m_2} \cdot 1/\eta_{m_1} \\ \Gamma_2 \supseteq \{x_1 :_{(\eta_{\epsilon'} \eta_{\epsilon_1}, \eta_{\delta'} \eta_{\delta_1})} \mathbb{M}^{c_1}_{\ell_1}[m_1, n_1]\ \tau_1, x_2 :_{(\eta_{\epsilon'} \eta_{\epsilon_2}, \eta_{\delta'} \eta_{\delta_2})} \mathbb{M}^{c_2}_{\ell_2}[m_1, n_2]\ \tau_2\} \\ \Delta, \Gamma_3 \uplus \{y_1 :_{(\eta_{\epsilon_1}, \eta_{\delta_1})} \mathbb{M}^{c_1}_{\ell_1}[m_2, n_1]\ \tau_1, y_2 :_{(\eta_{\epsilon_2}, \eta_{\delta_2})} \mathbb{M}^{c_2}_{\ell_2}[m_2, n_2]\ \tau_2\} \vdash e_2 : \tau_3\end{array}}{\lceil \Gamma_1 \rceil^{0,0} + \Gamma_2 + \Gamma_3 \vdash \texttt{sample}\ e_1\ \texttt{on}\ x_1, x_2\ \{x_3, x_4 \Rightarrow e_2\} : \tau_3}$$

RAND-NAT

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{N} \qquad \Gamma_2 \vdash e_2 : \mathbb{N}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{\infty} \vdash \texttt{rand-nat}(e_1, e_2) : \mathbb{N}}$$

Figure 20. Privacy Type Systems

$$\nu, \epsilon, \delta \in \mathbb{R}^+_\infty \quad ::= r^+ \mid \infty$$
$$D \in \mathrm{dom} \triangleq (\|D\| \in \mathrm{set}) \times (|\_|_D \in \|D\| \to \mathbb{R}^+_\infty) \times (|\_ - \_|_D \in \|D\| \times \|D\| \to \mathbb{R}^+_\infty)$$

$$|\_|^{L\infty}_D \in \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X|^{L\infty}_D \triangleq \sum_i \bigsqcup_j |X_{i,j}|_D$$
$$|\_|^{L1}_D \in \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X|^{L1}_D \triangleq \sum_i \sum_j |X_{i,j}|_D$$
$$|\_|^{L2}_D \in \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X|^{L2}_D \triangleq \sum_i \sqrt{\sum_j |X_{i,j}|^2_D}$$

$$|\_ - \_|^{L\infty}_D \in \mathrm{matrix}[m,n](\|D\|) \times \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X - Y|^{L\infty}_D \triangleq \sum_i \bigsqcup_j |X_{i,j} - Y_{i,j}|_D$$
$$|\_ - \_|^{L1}_D \in \mathrm{matrix}[m,n](\|D\|) \times \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X - Y|^{L1}_D \triangleq \sum_i \sum_j |X_{i,j} - Y_{i,j}|_D$$
$$|\_ - \_|^{L2}_D \in \mathrm{matrix}[m,n](\|D\|) \times \mathrm{matrix}[m,n](\|D\|) \to \mathbb{R}^+_\infty \qquad |X - Y|^{L2}_D \triangleq \sum_i \sqrt{\sum_j |X_{i,j} - Y_{i,j}|^2_D}$$

$$\begin{aligned}
\|\hat{\mathbb{N}}[n]\| &\triangleq \{n\} \\
\|\hat{\mathbb{R}}^+[r^+]\| &\triangleq \{r^+\} \\
\|\hat{\mathbb{N}}\| &\triangleq \mathbb{N} \\
\|\hat{\mathbb{R}}\| &\triangleq \mathbb{R} \\
\|\mathrm{data}\| &\triangleq \mathbb{R} \\
\|\mathrm{idx}[n]\| &\triangleq \{m \in \mathbb{N} \mid m < n\} \\
\|\widehat{\mathrm{matrix}}^U_\bigstar[m,n](D)\| &\triangleq \mathrm{matrix}[m,n](\|D\|) \\
\|\widehat{\mathrm{matrix}}^\ell_\bigstar[m,n](D)\| &\triangleq \left\{X \in \mathrm{matrix}[m,n](\|D\|) \;\middle|\; |X|^\ell_D \le 1\right\} \\
\|D_1 \hat{\uplus} D_2\| &\triangleq \|D_1\| \uplus \|D_2\| \\
\|D_1 \hat{\times} D_2\| &\triangleq \|D_1\| \times \|D_2\| \\
\|D_1 \mathbin{\hat{\&}} D_2\| &\triangleq \|D_1\| \times \|D_2\| \\
\|D_1 \mathbin{\widehat{\multimap}_\nu} D_2\| &\triangleq \left\{f \in \|D_1\| \to \|D_2\| \;\middle|\; |x-y|_{D_1} \le 1 \Rightarrow |f(x) - f(y)|_{D_2} \le \nu\right\} \\
\|(D_1@(\epsilon_1,\delta_1), \ldots, D_n@(\epsilon_n,\delta_n)) \mathbin{\widehat{\multimap}^*} X\| &\triangleq \left\{f \in \|D_1\| \times \cdots \times \|D_n\| \to \mathcal{D}(X) \;\middle|\; |x_i - y|_{D_i} \le 1 \Rightarrow \right. \\
&\qquad \left. \Pr[f(x_1,\ldots,x_i,\ldots,x_n) = d] \le e^{\epsilon_i} \Pr[f(x_1,\ldots,y,\ldots,x_n) = d] + \delta_i\right\} \\
\|\hat{\forall}\ [\beta_1, \ldots, \beta_n \in \mathbb{R}^+_\infty]\ D_{[\beta_i]}\| &\triangleq \forall (\beta_1 \ldots \beta_n \in \mathbb{R}^+_\infty).\ \|D_{[\beta_i]}\|
\end{aligned}$$

Figure 21. Metrics (1)

$$
\begin{aligned}
|n|_{\hat{\mathbb{N}}[n]} &\triangleq 0 \\
|r^+|_{\hat{\mathbb{R}}^+[r^+]} &\triangleq 0 \\
|n|_{\hat{\mathbb{N}}} &\triangleq n \\
|r|_{\hat{\mathbb{R}}} &\triangleq |r| \\
|r|_{\text{data}} &\triangleq |r| \\
|m|_{\text{idx}[n]} &\triangleq m \\
|X|_{\widehat{\text{matrix}}^{\star}_{\ell}[m,n](D)} &\triangleq |X|^{\ell}_{D} \\
|x_1 \in \|D_1\|\|_{D_1 \hat{\uplus} D_2} &\triangleq |x_1|_{D_1} \\
|x_2 \in \|D_2\|\|_{D_1 \hat{\uplus} D_2} &\triangleq |x_2|_{D_2} \\
|x_1, x_2|_{D_1 \hat{\times} D_2} &\triangleq |x_1|_{D_1} + |x_2|_{D_2} \\
|x_1, x_2|_{D_1 \hat{\&} D_2} &\triangleq |x_1|_{D_1} \sqcup |x_2|_{D_2} \\
|f|_{D_1 \widehat{\multimap}_{\nu} D_2} &\triangleq \infty \\
|f|_{(D_1@(\epsilon_1,\delta_2),\ldots,D_n@(\epsilon_n,\delta_n))\widehat{\multimap}^* X} &\triangleq \infty \\
|x|_{\hat{\forall}\ [\beta_1,\ldots,\beta_n \in \mathbb{R}^+_\infty]\ D_{[\beta_i]}} &\triangleq \bigsqcup_{\nu_i \in \mathbb{R}^+_\infty} |x_{[\nu_i/\beta_i]}|_{D_{[\nu_i/\beta_i]}}
\end{aligned}
$$

$$
\begin{aligned}
|n - n|_{\hat{\mathbb{N}}[n]} &\triangleq 0 \\
|r^+ - r^+|_{\hat{\mathbb{R}}^+[r^+]} &\triangleq 0 \\
|n_1 - n_2|_{\hat{\mathbb{N}}} &\triangleq |n_1 - n_2| \\
|r_1 - r_2|_{\hat{\mathbb{R}}} &\triangleq |r_1 - r_2| \\
|r_1 - r_2|_{\text{data}} &\triangleq \begin{cases} 0 & \text{if} \quad r_1 = r_2 \\ 1 & \text{if} \quad r_1 \neq r_2 \end{cases} \\
|m_1 - m_2|_{\text{idx}[n]} &\triangleq |m_1 - m_2| \\
|X_1 - X_2|_{\widehat{\text{matrix}}^{\star}_{\ell}[m,n](D)} &\triangleq |X_1 - X_2|^{\ell}_{D} \\
|x_1 - x_2|_{D_1 \hat{\uplus} D_2} &\triangleq \begin{cases} |x_1 - x_2|_{D_1} & \text{if} \quad \{x_1, x_2\} \subseteq D_1 \\ |x_1 - x_2|_{D_2} & \text{if} \quad \{x_1, x_2\} \subseteq D_2 \\ \infty & \text{if} \quad (x_1 \in D_1 \wedge x_2 \in D_2) \vee (x_1 \in D_2 \wedge x_2 \in D_1) \end{cases} \\
|\langle x_1, y_1\rangle - \langle x_2, y_2\rangle|_{D_1 \hat{\times} D_2} &\triangleq |x_1 - x_2|_{D_1} + |y_1 - y_2|_{D_2} \\
|\langle x_1, y_1\rangle - \langle x_2, y_2\rangle|_{D_1 \hat{\&} D_2} &\triangleq |x_1 - x_2|_{D_1} \sqcup |y_1 - y_2|_{D_2} \\
|f_1 - f_2|_{D_1 \widehat{\multimap}_{\nu} D_2} &\triangleq \begin{cases} 0 & \text{if} \quad f_1 = f_2 \\ \infty & \text{if} \quad f_1 \neq f_2 \end{cases} \\
|f_1 - f_2|_{[D_1@(\epsilon_1,\delta_2),\ldots,D_n@(\epsilon_n,\delta_n)]\widehat{\multimap}^* D} &\triangleq \begin{cases} 0 & \text{if} \quad f_1 = f_2 \\ \infty & \text{if} \quad f_1 \neq f_2 \end{cases} \\
|x_1 - x_2|_{\hat{\forall}\ [\beta_1,\ldots,\beta_n \in \mathbb{R}^+_\infty]\ D_{[\beta_i]}} &\triangleq \bigsqcup_{\nu_i \in \mathbb{R}^+_\infty} |x_{1[\nu_i/\beta_i]} - x_{2[\nu_i/\beta_i]}|_{D_{[\nu_i/\beta_i]}}
\end{aligned}
$$

Figure 22. Metrics (2)

$[\![\_]\!]\_ \in \text{rexp} \times \text{kenv} \to \mathbb{R}^+_\infty$

$$
\begin{aligned}
[\![\beta]\!]^\gamma &\triangleq \gamma(\beta) \\
[\![n]\!]^\gamma &\triangleq n \\
[\![r^+]\!]^\gamma &\triangleq r^+ \\
[\![\eta_1 \sqcup \eta_2]\!]^\gamma &\triangleq [\![\eta_1]\!]^\gamma \sqcup [\![\eta_2]\!]^\gamma \\
[\![\eta_1 \sqcap \eta_2]\!]^\gamma &\triangleq [\![\eta_1]\!]^\gamma \sqcap [\![\eta_2]\!]^\gamma \\
[\![\eta_1 + \eta_2]\!]^\gamma &\triangleq [\![\eta_1]\!]^\gamma + [\![\eta_2]\!]^\gamma \\
[\![\eta_1 \cdot \eta_2]\!]^\gamma &\triangleq [\![\eta_1]\!]^\gamma \cdot [\![\eta_2]\!]^\gamma \\
[\![1/\eta]\!]^\gamma &\triangleq 1/[\![\eta]\!]^\gamma \\
[\![\sqrt{\eta}]\!]^\gamma &\triangleq \sqrt{[\![\eta]\!]^\gamma} \\
[\![\ln \eta]\!]^\gamma &\triangleq \ln [\![\eta]\!]^\gamma
\end{aligned}
$$

$[\![\_]\!]\_ \in \text{sens} \times \text{kenv} \to \mathbb{R}^+_\infty$

$$
\begin{aligned}
[\![\eta]\!]^\gamma &\triangleq [\![\eta]\!]^\gamma \\
[\![\infty]\!]^\gamma &\triangleq \infty
\end{aligned}
$$

$[\![\_]\!]\_ \in \text{priv} \times \text{kenv} \to \mathbb{R}^+_\infty \times \mathbb{R}^+_\infty$

$$
\begin{aligned}
[\![\eta_\epsilon, \eta_\delta]\!]^\gamma &\triangleq [\![\eta_\epsilon]\!]^\gamma, [\![\eta_\delta]\!]^\gamma \\
[\![\infty]\!]^\gamma &\triangleq \infty, 0
\end{aligned}
$$

$[\![\_]\!]\_ \in \text{type} \times \text{kenv} \to \text{dom}$

$$
\begin{aligned}
[\![\mathbb{N}[\eta]]\!]^\gamma &\triangleq \hat{\mathbb{N}}[[\![\eta]\!]^\gamma] \\
[\![\mathbb{R}^+[\eta]]\!]^\gamma &\triangleq \hat{\mathbb{R}}^+[[\![\eta]\!]^\gamma] \\
[\![\mathbb{N}]\!]^\gamma &\triangleq \hat{\mathbb{N}} \\
[\![\mathbb{R}]\!]^\gamma &\triangleq \hat{\mathbb{R}} \\
[\![\texttt{data}]\!]^\gamma &\triangleq \text{data} \\
[\![\texttt{idx}[\eta]]\!]^\gamma &\triangleq \text{idx}[[\![\eta]\!]^\gamma] \\
[\![\texttt{matrix}^c_\ell[\eta_m, \eta_m]\ \tau]\!]^\gamma &\triangleq \widehat{\text{matrix}}^c_\ell[[\![\eta_m]\!]^\gamma, [\![\eta_n]\!]^\gamma]([\![\tau]\!]^\gamma) \\
[\![\tau_1 \uplus \tau_2]\!]^\gamma &\triangleq [\![\tau_1]\!]^\gamma \hat{\uplus} [\![\tau_2]\!]^\gamma \\
[\![\tau_1 \times \tau_2]\!]^\gamma &\triangleq [\![\tau_1]\!]^\gamma \hat{\times} [\![\tau_2]\!]^\gamma \\
[\![\tau_1 \,\&\, \tau_2]\!]^\gamma &\triangleq [\![\tau_1]\!]^\gamma \,\hat{\&}\, [\![\tau_2]\!]^\gamma \\
[\![\tau_1 \multimap_s \tau_2]\!]^\gamma &\triangleq [\![\tau_1]\!]^\gamma \,\widehat{\multimap}_{[\![s]\!]^\gamma}\, [\![\tau_2]\!]^\gamma \\
[\![\forall\ [\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n]\ (\tau_1@p_1, ..., \tau_n@p_n) \multimap^*\ \tau]\!]^\gamma &\triangleq \hat{\forall}\ [\beta_1, ..., \beta_n \in \mathbb{R}^+_\infty]\ ([\![\tau_1]\!]^{\gamma'}@[\![p_1]\!]^{\gamma'}, ..., [\![\tau_n]\!]^{\gamma'}@[\![p_n]\!]^{\gamma'})\ \widehat{\multimap}^*\ \|[\![\tau]\!]^{\gamma'}\| \\
&\text{where } \gamma' = \gamma \uplus \{\beta_1, ..., \beta_n\}
\end{aligned}
$$

$[\![\_, \_ \vdash \_]\!] \in \text{kcxt} \times \text{tcxt} \times \text{type} \to \text{dom}$

$$
\begin{aligned}
&[\![\overbrace{\{\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n\}}^{\Delta}, \overbrace{\{x_1{:}_{s_1}\tau_1, ..., x_n{:}_{s_n}\tau_n\}}^{\Gamma} \vdash \tau]\!] \triangleq \hat{\forall}\ [\beta_1, ..., \beta_n \in \mathbb{R}^+_\infty]\ ([\![\tau_1]\!]^\gamma \,\widehat{\multimap}_{[\![s_1]\!]^\gamma}\, \cdots([\![\tau_n]\!]^\gamma \,\widehat{\multimap}_{[\![s_n]\!]^\gamma}\, [\![\tau]\!]^\gamma)) \\
&\text{where } \gamma = \{\beta_1, ..., \beta_n\}
\end{aligned}
$$

$[\![\_, \_ \vdash \_]\!] \in \text{kcxt} \times \text{tcxt} \times \text{type} \to \text{dom}$

$$
\begin{aligned}
&[\![\overbrace{\{\beta_1{:}\kappa_1, ..., \beta_n{:}\kappa_n\}}^{\Delta}, \overbrace{\{x_1{:}_{p_1}\tau_1, ..., x_n{:}_{p_n}\tau_n\}}^{\Gamma} \vdash \tau]\!] \triangleq \hat{\forall}\ [\beta_1, ..., \beta_n \in \mathbb{R}^+_\infty]\ ([\![\tau_1]\!]^\gamma@[\![p_1]\!]^\gamma, ..., [\![\tau_n]\!]^\gamma@[\![p_n]\!]^\gamma)\ \widehat{\multimap}^*\ \|[\![\tau]\!]^\gamma\| \\
&\text{where } \gamma = \{\beta_1, ..., \beta_n\}
\end{aligned}
$$

Figure 23. Semantics of Typing

B Theorems & Lemmas

Theorem B.1 (Gaussian mechanism). *If* $|f(\gamma) - f(\gamma[x_i \mapsto d])| \leq r$, *then for* $f' = \lambda\gamma.\ f(\gamma) + \mathcal{N}(0, \sigma^2)$ *and* $\sigma^2 = \frac{2\log(1.25/\delta)r^2}{\epsilon^2}$, *and all* $\epsilon_i, \delta_i > 0$:

$$\Pr[f'(\gamma) = d] \leq e^{\epsilon_i}\Pr[f'(\gamma[x_i \mapsto d']) = d] + \delta_i$$

Proof. By the assumption, $\|f(\gamma) - f(\gamma[x_i \mapsto d])\|_2 \leq r$. The conclusion follows by Dwork and Roth [20], Theorem A.1. □

Theorem B.2 (Advanced composition). *If:*

- $\Pr[f(\gamma) = d] \leq e^{\epsilon_i}\Pr[f(\gamma[x_i \mapsto d']) = d] + \delta_i$
- $\delta' > 0$
- $\hat{\epsilon} = e^{2\epsilon_i}\sqrt{2n\log(1/\delta')}$
- $\hat{\delta} = \delta' + n\delta_i$

Then:

$$\Pr[\text{iter}(k, f(\gamma), \iota) = d''] \leq$$
$$e^{\hat{\epsilon}}\Pr[\text{iter}(k, f(\gamma[x_i \mapsto d']), \iota) = d''] + \hat{\delta}$$

Where iter *is defined as:*

$$\text{iter}(0, f, d) \triangleq d$$
$$\text{iter}(n, f, d) \triangleq \text{iter}(n-1, f, f(d))$$

Proof. By its definition, $\text{iter}(n, f, d)$ is an instance of n-fold adaptive composition of the (ϵ_i, δ_i)-differentially private mechanism f. The conclusion follows from Dwork and Roth [20], Theorem 3.20. □

Theorem B.3 (Sequential Composition). *If:*

$$|\gamma[x_i] - d| \leq 1 \implies$$
$$\Pr[f_1(\gamma) = d'] \leq e^{\epsilon_i}\Pr[f_1(\gamma[x_i \mapsto d]) = d'] + \delta_i$$

and:

$$|\gamma[x_i] - d| \leq 1 \implies$$
$$\Pr[f_2(\gamma[x \mapsto d']) = d''] \leq$$
$$e^{\epsilon'_i}\Pr[f_2(\gamma[x \mapsto d', x_i \mapsto d]) = d''] + \delta'_i$$

then:

$$|\gamma[x_i] - d| \leq 1 \implies$$
$$\Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d''] \leq$$
$$e^{\epsilon_i + \epsilon'_i}\Pr[f_1(\gamma[x_i \mapsto d]) = d', f_2(\gamma[x \mapsto d', x_i \mapsto d]) = d''] + \delta_i + \delta'_i$$

Proof. The conclusion follows from Dwork and Roth [20], Theorem B.1. □

C Proofs

We define the semantics of each language simultaneously with its proof of correctness. That is, we give a denotation for well typed terms that inhabits the interpretation of the typing judgment. We simplify the proof slightly to not consider symbolic real expressions in singleton and matrix types—the denotation for those expressions is straightforward as elements of real numbers extended with infinity, so we just consider this set in the proof below. (In this section, we use the same variable names at different colors to indicate distinct metavariables, e.g., Γ and Γ are not the same.)

Theorem C.1 (Soundness). *There exists an interpretation of well typed terms* $\Gamma \vdash e : \tau$ *and* $\Gamma \vdash e : \tau$, *notated* $[\![e]\!]$ *and* $[\![e]\!]$, *such that* $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$ *and* $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$.

Proof. We construct the interpretaion by induction on typing derivations $\Gamma \vdash e : \tau$ and $\Gamma \vdash e : \tau$ which are defined mutually inductively.

We show key cases for the proof in each language. For the sensitivity language $\Gamma \vdash e : \tau$, there is only one interesting case that is not already considered in prior work (Fuzz and DFuzz): the introduction rule for the privacy lambda.

$$\boxed{[\![\_]\!] \in \{e \in \text{exp} \mid \Gamma \vdash e : \tau\} \to [\![\Gamma \vdash \tau]\!]}$$

- Case $\overbrace{\lceil\Gamma\rceil^{\infty}}^{\Gamma} \vdash (p\lambda\ (\ldots,x_n{:}\tau_n) \Rightarrow e) : (\ldots,\tau_n@p_n) \multimap^* \tau$:
  By inversion:
  $$\Gamma \uplus \{x_1, :_{p_1} \tau_1, \ldots, x_n :_{p_n} \tau_n\} \vdash e : \tau$$
  By induction hypothesis (IH):
  $$[\![e]\!] = f \in [\![\Gamma \uplus \{x_1, :_{p_1} \tau_1, \ldots, x_n :_{p_n} \tau_n\} \vdash \tau]\!]$$
  Define:
  $$[\![p\lambda\ (\ldots,x_n{:}\tau_n) \Rightarrow e]\!] \triangleq f' \quad \textit{where}$$
  $$f'(\gamma) = \lambda\gamma'.\ f'(\gamma \uplus \gamma')$$
  To show $f' \in [\![\Gamma \vdash (\ldots,\tau_n@p_n) \multimap^* \tau]\!]$, i.e.:
  (1) $\forall \gamma \in [\![\overbrace{\lceil x'_i :_{\dot{r}'_i} \tau'_i \rceil}^{\Gamma}]\!], d \in [\![\tau_i]\!].$
  $|\gamma[x_i] - d|_{[\![\tau'_i]\!]} \leq 1$
  $\implies |f'(\gamma) - f'(\gamma[x_i \mapsto d])|_{[\![(\ldots,\tau_n@p_n) \multimap^* \tau]\!]} \leq \dot{r}'_i$
  and
  (2) $\forall \gamma \in [\![\Gamma]\!], \gamma' \in [\![\{x_i, :_{\epsilon_i,\delta_i} \tau_i\}]\!], d \in [\![\tau_i]\!].$
  $|\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1$
  $\implies \Pr[f'(\gamma)(\gamma') = d] \leq$
  $e^{\epsilon_i}\Pr[f'(\gamma)(\gamma'[x_i \mapsto d]) = d] + \delta_i$

* (1) Subcase: $\dot{r}'_i = \infty$
  The property holds trivially when $\dot{r}'_i = \infty$.
* (1) Subcase: $\dot{r}'_i = 0$
  It must be that $\{x'_i :_{0,0} \tau'_i\} \in \Gamma$.
  By IH, f is constant w.r.t. x'_i i.e. $\Pr[f'(\gamma)(\gamma') = d] = \Pr[f'(\gamma)(\gamma'[x'_i \mapsto d]) = d]$
  Therefore $|f(\gamma) - f(\gamma[x'_i \mapsto d])|_{[\![\tau'_i]\!]} = 0$.
* (2) follows immediately from IH and the definition of $[\![(\ldots,\tau_n@p_n) \multimap^* \tau]\!]$

For the privacy language $\Gamma \vdash \tau$ we repeat the Return and Bind rules (from Fuzz and DFuzz) in addition to our novel rules for privacy lambda application, mgauss, and loop. The rest of the rules in the privacy language are straightforward adaptations of these proofs.

$$\boxed{[\![\_]\!] \in \{e \in \exp \mid \Gamma \vdash e : \tau\} \to [\![\Gamma \vdash \tau]\!]}$$

- Case: $\overbrace{\rceil \Gamma \lceil^{\infty}}^{\Gamma} \vdash \texttt{return}\ e : \tau$
  By inversion:
  $$\Gamma \vdash e : \tau$$
  By Induction Hypothesis (IH):
  $$[\![e]\!] = f \in [\![\Gamma \vdash \tau]\!]$$
  Define:
  $$[\![\texttt{return}\ e]\!] \triangleq f' \quad \textit{where}$$
  $$\Pr[f'(\gamma) = d] \triangleq \begin{cases} 1 & \textit{if} \quad d = f(\gamma) \\ 0 & \textit{if} \quad d \neq f(\gamma) \end{cases}$$
  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:
  $$\forall \gamma \in [\![\overbrace{x_i :_{\epsilon_i,\, \delta_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1 \implies \Pr[f'(\gamma) = d] \leq e^{\epsilon_i} \Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta_i$$
  * Subcase: $\epsilon_i, \delta_i = \infty, 0$
    The property holds trivially when $\epsilon_i, \delta_i = \infty, 0$
  * Subcase: $\epsilon_i, \delta_i = 0, 0$
    It must be that $\{x'_i :_0 \tau'_i\} \in \Gamma$.
    By IH, f is constant w.r.t. x_i i.e.
    $|f(\gamma) - f(\gamma[x_i \mapsto d])|_{[\![\tau_i]\!]} = 0$
    Therefore $\Pr[f'(\gamma) = d] = \Pr[f'(\gamma[x_i \mapsto d]) = d]$
- Case: $\overbrace{\rceil \Gamma_1 \lceil^{\infty} + \{x_1 :_{p_1} \tau_1, \ldots, x_n :_{p_n} \tau_n\}}^{\Gamma} \vdash e(x_1,\ldots,x_n) : \tau$
  By inversion:
  $$\Gamma \vdash e : (\tau_1@p_1, \ldots, \tau_n@p_n) \multimap^* \tau$$
  By Induction Hypothesis (IH):
  $$[\![e]\!] = f \in [\![\Gamma \vdash (\tau_1@p_1, \ldots, \tau_n@p_n) \multimap^* \tau]\!]$$
  Define:
  $$[\![e(x_1,\ldots,x_n)]\!] \triangleq f' \quad \textit{where}$$
  $$\Pr[f'(\gamma) = d] \triangleq \Pr[f(\gamma)(\gamma[x_1], \ldots, \gamma[x_n]) = d]$$
  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:
  $$\forall \gamma \in [\![\overbrace{x_i :_{\epsilon_i,\, \delta_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1 \implies \Pr[f'(\gamma) = d] \leq e^{\epsilon_i} \Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta_i$$
  By definition of $+$ for privacy contexts:
  $\epsilon_i, \delta_i = \epsilon'_i + \epsilon''_i, \delta'_i + \delta''_i$ for
  $\{x_i :_{\epsilon'_i, \delta'_i} \tau_i\} \in \rceil \Gamma_1 \lceil^{\infty}$ and $\epsilon''_i, \delta''_i = [\![p_i]\!]$.
  * Subcase $\epsilon'_i, \delta'_i = \infty, 0$:
    $\epsilon_i, \delta_i = \infty, 0$; the property holds trivially
  * Subcase $\epsilon'_i, \delta'_i = 0, 0$:
    $\epsilon_i, \delta_i = \epsilon''_i, \delta''_i$; the property follows by IH and definition of $\multimap^*$ instiated to index i.
- Case $\overbrace{\Gamma_1 + \Gamma_2}^{\Gamma} \vdash x \leftarrow e_1 \,;\, e_2 : \tau_2$
  By inversion:
  $$\Gamma_1 \vdash e_1 : \tau_1$$
  $$\Gamma_2 \uplus \{x :_{\infty} \tau_1\} \vdash e_2 : \tau_2$$
  By Induction Hypothesis (IH):
  (1) $[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \tau_1]\!]$
  (2) $[\![e_2]\!] = f_2 \in [\![\Gamma_2 \uplus \{x :_{\infty} \tau_1\} \vdash \tau_2]\!]$
  Define:
  $$[\![x \leftarrow e_1; e_2]\!] \triangleq f' \quad \textit{where}$$
  $$\Pr[f'(\gamma) = d] \triangleq \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d]$$
  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:
  $$\forall \gamma \in [\![\overbrace{x_i :_{\epsilon_i,\, \delta_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1 \implies \Pr[f'(\gamma) = d] \leq e^{\epsilon_i} \Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta_i$$
  By definition of $+$ for privacy contexts:
  $\epsilon_i, \delta_i = \epsilon'_i + \epsilon''_i, \delta'_i + \delta''_i$ for
  $\{x_i :_{\epsilon'_i, \delta'_i} \tau\} \in \Gamma_1$ and $\{x_i :_{\epsilon''_i, \delta''_i} \tau\} \in \Gamma_2$.
  Property holds via IH.1, IH.2 and theorem B.3 instantiated with (ϵ'_i, δ'_i) and $(\epsilon''_i, \delta''_i)$.
- Case:
  $$\overbrace{\rceil \Gamma_1 + \Gamma_2 + \Gamma_3 \lceil^{0,0} + \rceil \Gamma_4 \lceil^{\infty} + \lfloor \Gamma_5 \rceil^{\epsilon,\delta}_{\{x_1,\ldots,x_n\}}}^{\Gamma} \vdash \texttt{mgauss}[e_1, e_2, e_3]\ \langle x_1, \ldots, x_n \rangle\ \{e_4\} : \mathbb{M}^U_{L\infty}[m, n]\ \mathbb{R}$$
  By inversion:
  $$\Gamma_1 \vdash e_1 : \mathbb{R}^+[\dot{r}]$$
  $$\Gamma_2 \vdash e_2 : \mathbb{R}^+[\epsilon]$$
  $$\Gamma_3 \vdash e_3 : \mathbb{R}^+[\delta]$$
  $$\Gamma_4 + \lfloor \Gamma_5 \rceil^{\dot{r}}_{\{x_1,\ldots,x_n\}} \vdash e_4 : \mathbb{M}^{\bigstar}_{L2}[m, n]\ \mathbb{R}$$

By Induction Hypothesis (IH):

$[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \mathbb{R}^+[\dot{r}]]\!]$
$[\![e_2]\!] = f_2 \in [\![\Gamma_2 \vdash \mathbb{R}^+[\epsilon]]\!]$
$[\![e_3]\!] = f_3 \in [\![\Gamma_3 \vdash \mathbb{R}^+[\delta]]\!]$
$[\![e_4]\!] = f_4 \in [\![\Gamma_4 + \lfloor\Gamma_5\rceil^{\dot{r}}_{\{x_1,\ldots,x_n\}} \vdash \mathbb{M}^{\bigstar}_{L2}[m,n]\ \mathbb{R}]\!]$

Define:

$[\![\texttt{mgauss}[e_1,e_2,e_3]\ \texttt{<}x_1,\ldots,x_n\texttt{>}\ \{e_4\}]\!] \triangleq f'$ *where*
$\Pr[f'(\gamma) = d] \triangleq \Pr[f_4(\gamma) + \mathcal{N}(0,\sigma^2) = d]$
$\sigma^2 = \frac{2\log(1.25/\delta)r^2}{\epsilon^2}$
$r = f_1(\gamma)$
$\epsilon = f_2(\gamma)$
$\delta = f_3(\gamma)$

To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

$\forall \gamma \in [\![\overset{\Gamma}{\overbrace{x_i :_{\epsilon_i,\delta_i} \tau_i}}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \le 1$
$\implies \Pr[f'(\gamma) = d] \le e^{\epsilon_i}\Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta_i$

By IH, we have that $\sigma^2 = \frac{2\log(1.25/\delta)\dot{r}^2}{\epsilon^2}$
By definition of $+$ for privacy contexts:
$\epsilon_i, \delta_i = \epsilon_i' + \epsilon_i'' + \epsilon_i''' + \epsilon_i'^4 + \epsilon_i'^5, \delta_i' + \delta_i'' + \delta_i''' + \delta_i'^4 + \delta_i'^5$
for
$\{x_i :_{\epsilon_i',\delta_i'} \tau\} \in \rceil\Gamma_1\lceil^{0,0}$
$\{x_i :_{\epsilon_i'',\delta_i''} \tau\} \in \rceil\Gamma_2\lceil^{0,0}$
$\{x_i :_{\epsilon_i''',\delta_i'''} \tau\} \in \rceil\Gamma_3\lceil^{0,0}$
$\{x_i :_{\epsilon_i'^4,\delta_i'^4} \tau\} \in \rceil\Gamma_4\lceil^{\infty}$
$\{x_i :_{\epsilon_i'^5,\delta_i'^5} \tau\} \in \lfloor\Gamma_5\rceil^{\epsilon,\delta}_{\{x_1,\ldots,x_n\}}$
Each of $\epsilon_i', \epsilon_i'', \epsilon_i''', \delta_i', \delta_i'', \delta_i'''$ must be 0.

* Subcase $\epsilon_i'^4, \delta_i'^4 = \infty, 0$:
$\epsilon_i, \delta_i = \infty, 0$; the property holds trivially
* Subcase $\epsilon'^4, \delta'^4 = 0, 0$:
$\epsilon_i, \delta_i = \epsilon_i'^5, \delta_i'^5 = \epsilon, \delta$
By IH, $|f_4(\gamma) - f_4(\gamma[x_i \mapsto d])|_{[\![\mathbb{R}]\!]} \le r$
The property follows from Theorem B.1.

- Case:

$$\overset{\Gamma}{\overbrace{\rceil\Gamma_1\lceil^{0,0} + \rceil\Gamma_2\lceil^{0,0} + \rceil\Gamma_3\lceil^{\infty} + \rceil\Gamma_4\lceil^{\infty} + \lfloor\Gamma_5\rceil^{2\epsilon\sqrt{2n\ln(\frac{1}{\delta'})},\delta'+n\delta}_{\{\ldots,x_n'\}}}} \vdash \texttt{loop}[e_1]\ e_2\ \texttt{on}\ e_3\ \texttt{<}\ldots,x_n'\texttt{>}\ \{x_1,x_2 \Rightarrow e_4\} : \tau$$

By inversion:

$\Gamma_1 \vdash e_1 : \mathbb{R}^+[\delta']$
$\Gamma_2 \vdash e_2 : \mathbb{R}^+[n]$
$\Gamma_3 \vdash e_3 : \tau$
$\Gamma_4 + \lfloor\Gamma_5\rceil^{\epsilon,\delta}_{\{x_1',\ldots,x_n'\}} \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_\infty \tau\} \vdash e_4 : \tau$

By Induction Hypothesis (IH):

$[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \mathbb{R}^+[\delta']]\!]$
$[\![e_2]\!] = f_2 \in [\![\Gamma_2 \vdash \mathbb{R}^+[n]]\!]$
$[\![e_3]\!] = f_3 \in [\![\Gamma_3 \vdash e_3 : \tau]\!]$
$[\![e_4]\!] = f_4 \in [\![\Gamma_4 + \lfloor\Gamma_5\rceil^{\epsilon,\delta}_{\{\ldots,x_n'\}} \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_\infty \tau\} \vdash \tau]\!]$

Define:

$[\![\texttt{loop}[e_1]\ e_2\ \texttt{on}\ e_3\ \texttt{<}\ldots,x_n'\texttt{>}\{x_1,x_2 \Rightarrow e_4\}]\!] \triangleq f'$ *where*
$\Pr[f'(\gamma) = d] \triangleq$
$\Pr[\text{iter}(f_2(\gamma), f_3(\gamma)$
$,(\lambda x i.\, f_4(\gamma[x_1 \mapsto i][x_2 \mapsto x]))) = d]$

To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

$\forall \gamma \in [\![\overset{\Gamma}{\overbrace{x_i :_{\epsilon_i,\delta_i} \tau_i}}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \le 1$
$\implies \Pr[f'(\gamma) = d] \le e^{\epsilon_i}\Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta_i$

By definition of $+$ for privacy contexts:
$\epsilon_i, \delta_i = \epsilon_i' + \epsilon_i'' + \epsilon_i''' + \epsilon_i'^4 + \epsilon_i'^5, \delta_i' + \delta_i'' + \delta_i''' + \delta_i'^4 + \delta_i'^5$
for
$\{x_i :_{\epsilon_i',\delta_i'} \tau\} \in \rceil\Gamma_1\lceil^{0,0}$
$\{x_i :_{\epsilon_i'',\delta_i''} \tau\} \in \rceil\Gamma_2\lceil^{0,0}$
$\{x_i :_{\epsilon_i''',\delta_i'''} \tau\} \in \rceil\Gamma_3\lceil^{\infty}$
$\{x_i :_{\epsilon_i'^4,\delta_i'^4} \tau\} \in \rceil\Gamma_4\lceil^{\infty}$
$\{x_i :_{\epsilon_i'^5,\delta_i'^5} \tau\} \in \lfloor\Gamma_5\rceil^{\epsilon,\delta}_{\{x_1,\ldots,x_n\}}$
Each of $\epsilon_i', \epsilon_i'', \delta_i', \delta_i''$ must be 0.

* Subcase $\epsilon_i''', \delta_i''' = \infty, 0$ or $\epsilon_i'^4, \delta_i'^4 = \infty, 0$:
$\epsilon_i, \delta_i = \infty, 0$; the property holds trivially
* Subcase $\epsilon_i''', \delta_i''' = 0, 0$ and $\epsilon'^4, \delta'^4 = 0, 0$:
$\epsilon_i, \delta_i = \epsilon_i'^5, \delta_i'^5 = 2\epsilon\sqrt{2n\ln(1/\delta')}, \delta' + n\delta$
By IH: $\Pr[f'(\gamma) = d] \le e^{\epsilon}\Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta$
The property holds by Theorem B.2:
$\Pr[f'(\gamma) = d] \le e^{2\epsilon\sqrt{2n\log(1/\delta')}}\Pr[f'(\gamma[x_i \mapsto d]) = d] + \delta$

□

D Rényi Differential Privacy

Theorem D.1 (Gaussian mechanism (Rényi Differential Privacy)). *If* $|f(\gamma) - f(\gamma[x_i \mapsto d])| \le r$, *then for* $f' = \lambda\gamma.\ f(\gamma) + \mathcal{N}(0,\sigma^2)$ *and* $\sigma^2 = \alpha r^2/(2\epsilon)$, *and all* $\alpha, \epsilon > 0$:

$$D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d'])) \le \epsilon$$

Proof. By Mironov [31], Proposition 7, we have that:

$$D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d'])) \le \alpha r^2/(2\sigma^2) = \epsilon$$

□

Theorem D.2 (Adaptive sequential composition (Rényi differential privacy)). *If:*

$$|\gamma[x_i] - d| \le 1 \implies$$
$$D_\alpha(f_1(\gamma) \| f_1(\gamma[x_i \mapsto d])) \le \epsilon_i$$

and:

$$|\gamma[x_i] - d| \le 1 \implies$$
$$D_\alpha(f_2(\gamma[x \mapsto d']) \| f_2(\gamma[x \mapsto d', x_i \mapsto d])) \le \epsilon_i'$$

and:

$$\Pr[f_3(\gamma) = d''] = \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d'']$$

then:

$$|\gamma[x_i] - d| \leq 1 \implies$$
$$D_\alpha(f_3(\gamma) \| f_3(\gamma[x_i \mapsto d])) \leq \epsilon_i + \epsilon_i'$$

Proof. The result follows directly from Mironov [31], Proposition 1, setting $f = f_1$ and $g = f_2$. □

Theorem D.3 (Soundness). *There exists an interpretation of well typed terms $\Gamma \vdash e : \tau$ and $\Gamma \vdash e : \tau$, notated $[\![e]\!]$ and $[\![e]\!]$, such that $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$ and $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$.*

Proof. We include the cases that are different from the proof for (ϵ, δ)-differential privacy: Bind and Gauss.

$$\boxed{[\![\_]\!] \in \{e \in \exp \mid \Gamma \vdash e : \tau\} \to [\![\Gamma \vdash \tau]\!]}$$

- Case $\overbrace{\Gamma_1 + \Gamma_2}^{\Gamma} \vdash x \leftarrow e_1 \,;\, e_2 : \tau_2$
  By inversion:

$$\Gamma_1 \vdash e_1 : \tau_1$$
$$\Gamma_2 \uplus \{x{:}_\infty \tau_1\} \vdash e_2 : \tau_2$$

  By Induction Hypothesis (IH):

  (1) $[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \tau_1]\!]$
  (2) $[\![e_2]\!] = f_2 \in [\![\Gamma_2 \uplus \{x{:}_\infty \tau_1\} \vdash \tau_2]\!]$

  Define:

$$[\![x \leftarrow e_1; e_2]\!] \triangleq f' \quad \textit{where}$$
$$\Pr[f'(\gamma) = d] \triangleq \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d]$$

  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

$$\forall \gamma \in [\![\overbrace{x_i{:}_{\alpha_i,\, \epsilon_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1$$
$$\implies D_{\alpha_i}(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \leq \epsilon_i$$

  By definition of $+$ for privacy contexts:
  $\epsilon_i = \epsilon_i' + \epsilon_i''$ for
  $\{x_i{:}_{\alpha_i, \epsilon_i'} \tau\} \in \Gamma_1$ and $\{x_i{:}_{\alpha_i, \epsilon_i''} \tau\} \in \Gamma_2$.
  Property holds via IH.1, IH.2 and theorem D.2 instantiated with (α_i, ϵ_i') and (α_i, ϵ_i'').
- Case:

$$\overbrace{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rfloor^{\alpha, \epsilon}_{\{x_1, \ldots, x_n\}}}^{\Gamma}$$
$$\vdash \texttt{mgauss}[e_1, e_2, e_3]\ \texttt{<}x_1, \ldots, x_n\texttt{>}\ \{e_4\} : \mathbb{M}^U_{L\infty}[m, n]\ \mathbb{R}$$

  By inversion:

$$\Gamma_1 \vdash e_1 : \mathbb{R}^+[\dot{r}]$$
$$\Gamma_2 \vdash e_2 : \mathbb{R}^+[\alpha]$$
$$\Gamma_3 \vdash e_3 : \mathbb{R}^+[\epsilon]$$
$$\Gamma_4 + \lfloor \Gamma_5 \rfloor^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \mathbb{M}^\star_{L2}[m, n]\ \mathbb{R}$$

  By Induction Hypothesis (IH):

$$[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \mathbb{R}^+[\dot{r}]]\!]$$
$$[\![e_2]\!] = f_2 \in [\![\Gamma_2 \vdash \mathbb{R}^+[\alpha]]\!]$$
$$[\![e_3]\!] = f_3 \in [\![\Gamma_3 \vdash \mathbb{R}^+[\epsilon]]\!]$$
$$[\![e_4]\!] = f_4 \in [\![\Gamma_4 + \lfloor \Gamma_5 \rfloor^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash \mathbb{M}^\star_{L2}[m, n]\ \mathbb{R}]\!]$$

  Define:

$$[\![\texttt{mgauss}[e_1, e_2, e_3]\ \texttt{<}x_1, \ldots, x_n\texttt{>}\ \{e_4\}]\!] \triangleq f' \quad \textit{where}$$
$$\Pr[f'(\gamma) = d] \triangleq \Pr[f_4(\gamma) + \mathcal{N}(0, \sigma^2) = d]$$
$$\sigma^2 = \frac{r^2 \alpha}{2\epsilon}$$
$$r = f_1(\gamma)$$
$$\alpha = f_2(\gamma)$$
$$\epsilon = f_3(\gamma)$$

  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

$$\forall \gamma \in [\![\overbrace{x_i{:}_{\alpha_i,\, \epsilon_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1$$
$$\implies D_{\alpha_i}(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \leq \epsilon_i$$

  By IH, we have that $\sigma^2 = \frac{r^2 \alpha}{2\epsilon}$
  By definition of $+$ for privacy contexts:
  $\epsilon_i = \epsilon_i' + \epsilon_i'' + \epsilon_i''' + \epsilon_i'^4 + \epsilon_i'^5$ for
  $\{x_i{:}_{\alpha_i, \epsilon_i'} \tau\} \in \lceil \Gamma_1 \rceil^{0,0}$
  $\{x_i{:}_{\alpha_i, \epsilon_i''} \tau\} \in \lceil \Gamma_2 \rceil^{0,0}$
  $\{x_i{:}_{\alpha_i, \epsilon_i'''} \tau\} \in \lceil \Gamma_3 \rceil^{0,0}$
  $\{x_i{:}_{\alpha_i, \epsilon_i'^4} \tau\} \in \lceil \Gamma_4 \rceil^{\infty}$
  $\{x_i{:}_{\alpha_i, \epsilon_i'^5} \tau\} \in \lfloor \Gamma_5 \rfloor^{\alpha, \epsilon}_{\{x_1, \ldots, x_n\}}$
  Each of ϵ_i', ϵ_i'', ϵ_i''' must be 0.

  * Subcase $\epsilon_i'^4 = \infty$:
    $\epsilon_i = \infty$; the property holds trivially
  * Subcase $\epsilon'^4 = 0$:
    $\alpha_i, \epsilon_i = \alpha_i, \epsilon_i'^5 = \alpha, \epsilon$
    By IH, $|f_4(\gamma) - f_4(\gamma[x_i \mapsto d])|_{[\![\mathbb{R}]\!]} \leq r$
    The property follows from Theorem D.1.

□

E Zero-Concentrated Differential Privacy

Theorem E.1 (Gaussian mechanism (Zero-Concentrated Differential Privacy)). *If $|f(\gamma) - f(\gamma[x_i \mapsto d])| \leq r$, then for $f' = \lambda\gamma.\ f(\gamma) + \mathcal{N}(0, \sigma^2)$ and $\sigma^2 = r^2/(2\rho)$, and all $\rho > 0$ and $\alpha \geq 1$:*

$$D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d'])) \leq \alpha\rho$$

$$\boxed{\Delta, \Gamma \vdash e : \tau}$$

STATIC LOOP

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_n] \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 + \lfloor \Gamma_4 \rceil^{\eta_\alpha, \eta_\epsilon}_{\{x'_1, \ldots, x'_n\}} \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_\infty \tau\} \vdash e_3 : \tau}{\Delta, \lceil \Gamma_1 + \Gamma_1 \rceil^{0,0} + \lceil \Gamma_2 \rceil^{\infty} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rceil^{\eta_\alpha, \eta_n \cdot \eta_\epsilon}_{\{x'_1, \ldots, x'_n\}} \vdash \texttt{loop}\ e_1\ \texttt{on}\ e_2\ \texttt{<}x'_1, \ldots, x'_n\texttt{>}\ \{x_1, x_2 \Rightarrow e_3\} : \tau}$$

GAUSS

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\alpha] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_4 + \lfloor \Gamma_5 \rceil^{\eta_s}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rceil^{\eta_\alpha, \eta_\epsilon}_{\{x_1, \ldots, x_n\}} \vdash \texttt{gauss}[e_1, e_2, e_3]\ \texttt{<}x_1, \ldots, x_n\texttt{>}\ \{e_4\} : \mathbb{R}}$$

MGAUSS

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\alpha] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\epsilon] \qquad \Gamma_4 + \lfloor \Gamma_5 \rceil^{\eta_s}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \texttt{matrix}^{\bigstar}_{L2}[\eta_m, \eta_n]\ \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rceil^{\eta_\alpha, \eta_\epsilon}_{\{x_1, \ldots, x_n\}} \vdash \texttt{mgauss}[e_1, e_2, e_3]\ \texttt{<}x_1, \ldots, x_n\texttt{>}\ \{e_4\} : \texttt{matrix}^{U}_{L\infty}[\eta_m, \eta_n]\ \mathbb{R}}$$

Figure 24. Privacy Type System Modifications, Rényi Differential Privacy

Proof. By Bun and Steinke [13], Lemma 2.4, we have that:

$$D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d'])) \leq \alpha r^2/(2\sigma^2) = \alpha\rho$$

□

Theorem E.2 (Adaptive sequential composition (Zero-Concentrated differential privacy)). *For all $\alpha \geq 1$, if:*

$$|\gamma[x_i] - d| \leq 1 \implies D_\alpha(f_1(\gamma) \| f_1(\gamma[x_i \mapsto d])) \leq \alpha\rho_i$$

and:

$$|\gamma[x_i] - d| \leq 1 \implies D_\alpha(f_2(\gamma[x \mapsto d']) \| f_2(\gamma[x \mapsto d', x_i \mapsto d])) \leq \alpha\rho'_i$$

and:

$$\Pr[f_3(\gamma) = d''] = \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d'']$$

then:

$$|\gamma[x_i] - d| \leq 1 \implies D_\alpha(f_3(\gamma) \| f_3(\gamma[x_i \mapsto d])) \leq \alpha(\rho_i + \rho'_i)$$

Proof. The result follows directly from Bun and Steinke [13], Lemma 2.3, setting $M = f_1$ and $M' = f_2$. □

Theorem E.3 (Soundness). *There exists an interpretation of well typed terms $\Gamma \vdash e : \tau$ and $\Gamma \vdash e : \tau$, notated $[\![e]\!]$ and $[\![e]\!]$, such that $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$ and $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$.*

Proof. We include the cases that are different from the proof for (ϵ, δ)-differential privacy: BIND and GAUSS.

$$\boxed{[\![\_]\!] \in \{e \in \exp \mid \Gamma \vdash e : \tau\} \to [\![\Gamma \vdash \tau]\!]}$$

- Case $\overbrace{\Gamma_1 + \Gamma_2}^{\Gamma} \vdash x \leftarrow e_1 ; e_2 : \tau_2$

  By inversion:

  $$\Gamma_1 \vdash e_1 : \tau_1$$
  $$\Gamma_2 \uplus \{x :_\infty \tau_1\} \vdash e_2 : \tau_2$$

  By Induction Hypothesis (IH):

  (1) $[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \tau_1]\!]$
  (2) $[\![e_2]\!] = f_2 \in [\![\Gamma_2 \uplus \{x :_\infty \tau_1\} \vdash \tau_2]\!]$

  Define:

  $[\![x \leftarrow e_1; e_2]\!] \triangleq f'$ *where*
  $\Pr[f'(\gamma) = d] \triangleq \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d]$

  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

  $$\forall \gamma \in [\![\overbrace{x_i :_{\alpha_i, \epsilon_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1 \implies \forall \alpha.\ D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \leq \rho_i$$

  By definition of $+$ for privacy contexts:
  $\rho_i = \rho'_i + \rho''_i$ for
  $\{x_i :_{\rho'_i} \tau\} \in \Gamma_1$ and $\{x_i :_{\rho''_i} \tau\} \in \Gamma_2$.
  Property holds via IH.1, IH.2 and theorem E.2 instantiated with ρ'_i and ρ''_i.
- Case:

  $$\overbrace{\lceil \Gamma_1 + \Gamma_2 \rceil^{0} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rceil^{\rho}_{\{x_1, \ldots, x_n\}}}^{\Gamma} \vdash \texttt{mgauss}[e_1, e_2]\ \texttt{<}x_1, \ldots, x_n\texttt{>}\ \{e_3\} : \mathbb{M}^U_{L\infty}[m, n]\ \mathbb{R}$$

  By inversion:

  $$\Gamma_1 \vdash e_1 : \mathbb{R}^+[\dot{r}]$$
  $$\Gamma_2 \vdash e_2 : \mathbb{R}^+[\rho]$$
  $$\Gamma_3 + \lfloor \Gamma_4 \rceil^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash e_3 : \mathbb{M}^{\bigstar}_{L2}[m, n]\ \mathbb{R}$$

  By Induction Hypothesis (IH):

  $$[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \mathbb{R}^+[\dot{r}]]\!]$$
  $$[\![e_2]\!] = f_2 \in [\![\Gamma_2 \vdash \mathbb{R}^+[\rho]]\!]$$
  $$[\![e_3]\!] = f_3 \in [\![\Gamma_3 + \lfloor \Gamma_4 \rceil^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash \mathbb{M}^{\bigstar}_{L2}[m, n]\ \mathbb{R}]\!]$$

$\boxed{\Delta, \Gamma \vdash e : \tau}$

Static Loop

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_n] \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 + \lfloor \Gamma_4 \rfloor^{\eta_\rho}_{\{x'_1, ..., x'_n\}} \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_\infty \tau\} \vdash e_3 : \tau}{\Delta, \lceil \Gamma_1 + \Gamma_1 \rceil^{0,0} + \lceil \Gamma_2 \rceil^{\infty} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\eta_n \cdot \eta_\rho}_{\{x'_1, ..., x'_n\}} \vdash \texttt{loop}\ e_1\ \texttt{on}\ e_2\ \texttt{<}x'_1, ..., x'_n\texttt{>}\ \{x_1, x_2 \Rightarrow e_3\} : \tau}$$

Gauss

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\rho] \qquad \Gamma_3 + \lfloor \Gamma_4 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \vdash e_3 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{0} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\eta_\rho}_{\{x_1, ..., x_n\}} \vdash \texttt{gauss}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_3\} : \mathbb{R}}$$

MGauss

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\rho] \qquad \Gamma_3 + \lfloor \Gamma_4 \rfloor^{\eta_s}_{\{x_1, ..., x_n\}} \vdash e_3 : \texttt{matrix}^{\bigstar}_{L2}[\eta_m, \eta_n]\ \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 \rceil^{0} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rfloor^{\eta_\rho}_{\{x_1, ..., x_n\}} \vdash \texttt{mgauss}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_3\} : \texttt{matrix}^{U}_{L\infty}[\eta_m, \eta_n]\ \mathbb{R}}$$

Figure 25. Privacy Type System Modifications, Zero-Concentrated Differential Privacy

Define:

$$[\![\texttt{mgauss}[e_1, e_2]\ \texttt{<}x_1, ..., x_n\texttt{>}\ \{e_3\}]\!] \triangleq f' \quad \textit{where}$$
$$\Pr[f'(\gamma) = d] \triangleq \Pr[f_3(\gamma) + \mathcal{N}(0, \sigma^2) = d]$$
$$\sigma^2 = \frac{r^2}{2\rho}$$
$$r = f_1(\gamma)$$
$$\rho = f_2(\gamma)$$

To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

$$\forall \gamma \in [\![\overbrace{x_i :_{\rho_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \le 1 \implies \forall \alpha.\ D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \le \rho_i$$

By IH, we have that $\sigma^2 = \frac{r^2}{2\rho}$
By definition of $+$ for privacy contexts:
$\rho_i = \rho'_i + \rho''_i + \rho'''_i + \rho'^4_i$ for
$\{x_i :_{\rho'_i} \tau\} \in \lceil \Gamma_1 \rceil^{0,0}$
$\{x_i :_{\rho''_i} \tau\} \in \lceil \Gamma_2 \rceil^{0,0}$
$\{x_i :_{\rho'''_i} \tau\} \in \lceil \Gamma_3 \rceil^{\infty}$
$\{x_i :_{\rho'^4_i} \tau\} \in \lfloor \Gamma_4 \rfloor^{\rho}_{\{x_1, ..., x_n\}}$
Each of ρ'_i, ρ''_i must be 0.

* Subcase $\rho'''_i = \infty$:
  $\rho_i = \infty$; the property holds trivially
* Subcase $\rho'''_i = 0$:
  $\rho_i = \rho'^4_i = \rho$
  By IH, $|f_3(\gamma) - f_3(\gamma[x_i \mapsto d])|_{[\![\mathbb{R}]\!]} \le r$
  The property follows from Theorem E.1.

□

F Truncated Concentrated Differential Privacy

Theorem F.1 (Sinh-normal mechanism (Truncated Concentrated Differential Privacy)). *If* $|f(\gamma) - f(\gamma[x_i \mapsto d])| \le r$, $\rho > 0$, *and* $\omega > 1/\sqrt{\rho}$, *then for*

$$f' = \lambda \gamma.\ f(\gamma) + \omega r\ \text{arsinh}(\frac{1}{\omega r} \mathcal{N}(0, \sigma^2))$$

and $\sigma^2 = r^2/(2\rho)$, *we have:*

$$\forall \alpha \in (1, \omega).\ D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d'])) \le \alpha\rho$$

Proof. The result follows directly from Bun et al.[12], Proposition 4. □

Theorem F.2 (Adaptive sequential composition (Truncated Concentrated Differential Privacy)). *If:*

$$|\gamma[x_i] - d| \le 1 \implies \forall \alpha \in (1, \omega_i).\ D_\alpha(f_1(\gamma) \| f_1(\gamma[x_i \mapsto d])) \le \alpha\rho_i$$

and:

$$|\gamma[x_i] - d| \le 1 \implies \forall \alpha \in (1, \omega'_i).\ D_\alpha(f_2(\gamma[x \mapsto d']) \| f_2(\gamma[x \mapsto d', x_i \mapsto d])) \le \alpha\rho'_i$$

and:

$$\Pr[f_3(\gamma) = d''] = \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d'']$$

then:

$$|\gamma[x_i] - d| \le 1 \implies \forall \alpha \in (1, \omega_i \sqcap \omega'_i).\ D_\alpha(f_3(\gamma) \| f_3(\gamma[x_i \mapsto d])) \le \alpha(\rho_i + \rho'_i)$$

Proof. The result follows directly from Bun et al. [12], Lemma 2, setting $M = f_1$ and $M' = f_2$. □

Theorem F.3 (Soundness). *There exists an interpretation of well typed terms* $\Gamma \vdash e : \tau$ *and* $\Gamma \vdash e : \tau$, *notated* $[\![e]\!]$ *and* $[\![e]\!]$, *such that* $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$ *and* $[\![e]\!] \in [\![\Gamma \vdash \tau]\!]$.

$\boxed{\Delta, \Gamma \vdash e : \tau}$

STATIC LOOP

$$\frac{\Delta, \Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_n] \qquad \Delta, \Gamma_2 \vdash e_2 : \tau \qquad \Delta, \Gamma_3 + \lfloor \Gamma_4 \rceil^{\eta_\rho, \eta_\omega}_{\{x'_1, \ldots, x'_n\}} \uplus \{x_1 :_\infty \mathbb{N}, x_2 :_\infty \tau\} \vdash e_3 : \tau}{\Delta, \lceil \Gamma_1 + \Gamma_1 \rceil^{0,0} + \lceil \Gamma_2 \rceil^{\infty} + \lceil \Gamma_3 \rceil^{\infty} + \lfloor \Gamma_4 \rceil^{\eta_n \cdot \eta_\rho, \eta_\omega}_{\{x'_1, \ldots, x'_n\}} \vdash \texttt{loop } e_1 \texttt{ on } e_2 \texttt{ <}x'_1, \ldots, x'_n\texttt{> } \{x_1, x_2 \Rightarrow e_3\} : \tau}$$

SINHNORMAL

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\rho] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\omega] \qquad \Gamma_4 + \lfloor \Gamma_5 \rceil^{\eta_s}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rceil^{\eta_\rho, \eta_\omega}_{\{x_1, \ldots, x_n\}} \vdash \texttt{sinh-normal}[e_1, e_2, e_3] \texttt{ <}x_1, \ldots, x_n\texttt{> } \{e_4\} : \mathbb{R}}$$

MSINHNORMAL

$$\frac{\Gamma_1 \vdash e_1 : \mathbb{R}^+[\eta_s] \qquad \Gamma_2 \vdash e_2 : \mathbb{R}^+[\eta_\rho] \qquad \Gamma_3 \vdash e_3 : \mathbb{R}^+[\eta_\omega] \qquad \Gamma_4 + \lfloor \Gamma_5 \rceil^{\eta_s}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \texttt{matrix}^{\bigstar}_{L2}[\eta_m, \eta_n]\ \mathbb{R}}{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rceil^{\eta_\rho, \eta_\omega}_{\{x_1, \ldots, x_n\}} \vdash \texttt{msinh-normal}[e_1, e_2, e_3] \texttt{ <}x_1, \ldots, x_n\texttt{> } \{e_4\} : \texttt{matrix}^{U}_{L\infty}[\eta_m, \eta_n]\ \mathbb{R}}$$

Figure 26. Privacy Type System Modifications, Truncated Concentrated Differential Privacy

Proof. We include the cases that are different from the proof for (ϵ, δ)-differential privacy: BIND and SINHNORMAL.

$\boxed{[\![\_]\!] \in \{e \in \text{exp} \mid \Gamma \vdash e : \tau\} \to [\![\Gamma \vdash \tau]\!]}$

- Case $\overbrace{\Gamma_1 + \Gamma_2}^{\Gamma} \vdash x \leftarrow e_1 \,;\, e_2 : \tau_2$
  By inversion:

  $\Gamma_1 \vdash e_1 : \tau_1$
  $\Gamma_2 \uplus \{x :_\infty \tau_1\} \vdash e_2 : \tau_2$

  By Induction Hypothesis (IH):

  (1) $[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \tau_1]\!]$
  (2) $[\![e_2]\!] = f_2 \in [\![\Gamma_2 \uplus \{x :_\infty \tau_1\} \vdash \tau_2]\!]$

  Define:

  $[\![x \leftarrow e_1; e_2]\!] \triangleq f'$ *where*
  $\Pr[f'(\gamma) = d] \triangleq \Pr[f_1(\gamma) = d', f_2(\gamma[x \mapsto d']) = d]$

  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

  $\forall \gamma \in [\![\overbrace{x_i :_{\rho_i,\, \omega_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1$
  $\implies \forall \alpha \in (1, \omega_i).\ D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \leq \alpha \rho_i$

  By definition of $+$ for privacy contexts:
  $\rho_i, \omega_i = \rho'_i + \rho''_i, \omega'_i \sqcap \omega''_i$ for
  $\{x_i :_{\rho_i, \omega'_i} \tau\} \in \Gamma_1$ and $\{x_i :_{\rho_i, \omega''_i} \tau\} \in \Gamma_2$.
  Property holds via IH.1, IH.2 and theorem F.2 instantiated with (ρ_i, ω'_i) and (ρ_i, ω''_i).
- Case:

  $\overbrace{\lceil \Gamma_1 + \Gamma_2 + \Gamma_3 \rceil^{0,0} + \lceil \Gamma_4 \rceil^{\infty} + \lfloor \Gamma_5 \rceil^{\rho, \omega}_{\{x_1, \ldots, x_n\}}}^{\Gamma}$
  $\vdash \texttt{msinh-normal}[e_1, e_2, e_3] \texttt{ <}x_1, \ldots, x_n\texttt{> } \{e_4\} : \mathbb{M}^U_{L\infty}[m, n]\ \mathbb{R}$

  By inversion:

  $\Gamma_1 \vdash e_1 : \mathbb{R}^+[\dot{r}]$
  $\Gamma_2 \vdash e_2 : \mathbb{R}^+[\rho]$
  $\Gamma_3 \vdash e_3 : \mathbb{R}^+[\omega]$
  $\Gamma_4 + \lfloor \Gamma_5 \rceil^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash e_4 : \mathbb{M}^{\bigstar}_{L2}[m, n]\ \mathbb{R}$

  By Induction Hypothesis (IH):

  $[\![e_1]\!] = f_1 \in [\![\Gamma_1 \vdash \mathbb{R}^+[\dot{r}]]\!]$
  $[\![e_2]\!] = f_2 \in [\![\Gamma_2 \vdash \mathbb{R}^+[\rho]]\!]$
  $[\![e_3]\!] = f_3 \in [\![\Gamma_3 \vdash \mathbb{R}^+[\omega]]\!]$
  $[\![e_4]\!] = f_4 \in [\![\Gamma_4 + \lfloor \Gamma_5 \rceil^{\dot{r}}_{\{x_1, \ldots, x_n\}} \vdash \mathbb{M}^{\bigstar}_{L2}[m, n]\ \mathbb{R}]\!]$

  Define:

  $[\![\texttt{msinh-normal}[e_1, e_2, e_3] \texttt{ <}x_1, \ldots, x_n\texttt{> } \{e_4\}]\!] \triangleq f'$ *where*
  $\Pr[f'(\gamma) = d] \triangleq \Pr[f_4(\gamma) + \omega r\ \text{arsinh}(\frac{1}{\omega r}\mathcal{N}(0, \sigma^2)) = d]$
  $\sigma^2 = \frac{r^2}{2\rho}$
  $r = f_1(\gamma)$
  $\rho = f_2(\gamma)$
  $\omega = f_3(\gamma)$

  To show $f' \in [\![\Gamma \vdash \tau]\!]$, i.e.:

  $\forall \gamma \in [\![\overbrace{x_i :_{\rho_i,\, \omega_i} \tau_i}^{\Gamma}]\!], d \in [\![\tau_i]\!].\ |\gamma[x_i] - d|_{[\![\tau_i]\!]} \leq 1$
  $\implies \forall \alpha \in (1, \omega_i).\ D_\alpha(f'(\gamma) \| f'(\gamma[x_i \mapsto d])) \leq \alpha \rho_i$

  By IH, we have that $\sigma^2 = \frac{r^2}{2\rho}$
  By definition of $+$ for privacy contexts:
  $\rho_i, \omega_i = \rho'_i + \rho''_i + \rho'''_i + \rho'^4_i + \rho'^5_i, \omega'_i + \omega''_i + \omega'''_i + \omega'^4_i + \omega'^5_i$
  for
  $\{x_i :_{\rho_i, \omega'_i} \tau\} \in \lceil \Gamma_1 \rceil^{0,0}$
  $\{x_i :_{\rho_i, \omega''_i} \tau\} \in \lceil \Gamma_2 \rceil^{0,0}$
  $\{x_i :_{\rho_i, \omega'''_i} \tau\} \in \lceil \Gamma_3 \rceil^{0,0}$
  $\{x_i :_{\rho_i, \omega'^4_i} \tau\} \in \lceil \Gamma_4 \rceil^{\infty}$
  $\{x_i :_{\rho_i, \omega'^5_i} \tau\} \in \lfloor \Gamma_5 \rceil^{\rho, \omega}_{\{x_1, \ldots, x_n\}}$
  Each of $\rho'_i, \rho''_i, \rho'''_i, \omega'_i, \omega''_i, \omega'''_i$ must be 0.

* Subcase $\rho_i'^4 = \infty$:
  $\rho_i = \infty$; the property holds trivially
* Subcase $\rho'^4 = 0$:
  $\rho_i, \omega_i = \rho_i, \omega_i'^5 = \rho, \omega$
  By IH, $|f_4(\gamma) - f_4(\gamma[x_i \mapsto d])|_{[\![\mathbb{R}]\!]} \leq r$
  The property follows from Theorem F.1.

□