Journal of Machine Learning Research 18 (2017) 1–5 Submitted 7/17; Published

# diffpriv: An R Package for Easy Differential Privacy

**Benjamin I. P. Rubinstein** BRUBINSTEIN@UNIMELB.EDU.AU
*School of Computing and Information Systems*
*The University of Melbourne, Parkville, VIC 3010, Australia*

**Francesco Aldà** FRANCESCO.ALDA@RUB.DE
*Horst Görtz Institute for IT Security and Faculty of Mathematics*
*Ruhr-Universität Bochum, D-44801 Bochum, Germany*

**Editor:** TBD

## Abstract

The R package diffpriv provides tools for statistics and machine learning under differential privacy. Suitable for releasing analyses on privacy-sensitive data to untrusted third parties, differential privacy has become the framework of choice for privacy-preserving learning. diffpriv delivers: (a) implementations of generic mechanisms for privatizing non-private target functions, including the Laplace, Gaussian, exponential and Bernstein mechanisms; (b) a recent sensitivity sampler due to **?** that empirically estimates the sensitivity of non-private targets—obviating mathematical analysis for exact sensitivity bounds needed for most generic mechanisms; (c) an extensible framework for implementing differentially-private mechanisms. Together, the components of diffpriv permit easy high-utility privatization of complex analyses, learners and even black-box software programs.

**Keywords:** differential privacy, empirical process theory, R, open-source software

## 1. Introduction

Differential privacy (**?**) has quickly become a key framework for semantic guarantees of data privacy when releasing analysis on privacy-sensitive data to untrusted third parties. The framework's popularity is in part due to a suite of generic mechanisms for privatizing non-private target functions of data *i.e.*, statistics, estimation procedures, and learners. Common to most generic mechanisms is the requirement that the non-private target's sensitivity to data set perturbation is known and bounded. Unfortunately, bounding sensitivity can be prohibitively complex for potential end users. This paper describes the diffpriv R package that implements generic mechanisms for differential privacy, along with our recent sensitivity sampler that replaces exact sensitivity bounds with empirical estimates (**?**). As a result, diffpriv privatizes a wide range of procedures under random differential privacy (**?**), automatically without mathematical analysis and in many cases achieving high utility. diffpriv v0.4.2 is available from https://brubinstein.github.io/diffpriv/ the project homepage and CRAN from https://cran.r-project.org/package=diffpriv under an open-source license.

## 2. Generic Mechanisms for Differential Privacy

Fundamental to differential privacy is a privacy-sensitive *data set* (or *database*) $D \in \mathcal{D}^n$ on *domain* $\mathcal{D}$. In diffpriv a data set can be a `list`, `matrix`, `data.frame` or `vector`. We

say that a pair of databases $D, D' \in \mathcal{D}^n$ is *neighboring* if they differ on exactly one record. While individual records should not be revealed, we aim to release aggregate information on $D$ with a mechanism. A *mechanism* $M : \mathcal{D}^n \to \mathcal{R}$ is a random-valued function of databases taking values in a response set $\mathcal{R}$; implemented in diffpriv as virtual (abstract) class `DPMech`.

**Definition 1 (?)** *For $\epsilon > 0$, mechanism $M : \mathcal{D}^n \to \mathcal{R}$ preserves* $\epsilon$-differential privacy *if for all neighboring pairs $D, D' \in \mathcal{D}^n$, measurable response sets $R \subseteq \mathcal{R}$, $\Pr(M(D) \in R) \leq \exp(\epsilon) \cdot \Pr(M(D') \in R)$. Alternatively for $\delta \in (0, 1)$, relaxed* $(\epsilon, \delta)$-differential privacy *holds if $\Pr(M(D) \in R) \leq \exp(\epsilon) \cdot \Pr(M(D') \in R) + \delta$.*

A mechanism preserves DP if its response distributions are close on neighboring pairs: an adversary cannot determine an unknown record from responses, even with knowledge of the remaining records. Privacy parameters $\epsilon$ $(\epsilon, \delta)$ are encapsulated in class `DPParamsEps` (`DPParamsDel`). `DPMech` generic method `releaseResponse(mechanism, privacyParams, X)` takes privacy parameters and a sensitive dataset, to private response. Most generic mechanisms in DP share a number of properties leveraged by the diffpriv package as follows.

**Privatizing a target function.** Many mechanisms $M : \mathcal{D}^n \to \mathcal{R}$ seek to privatize a non-private *target* function $f : \mathcal{D}^n \to \mathcal{B}$, with range $\mathcal{B}$ often coinciding with $\mathcal{R}$. Accordingly `DPMech` objects are initialized with `target` slot of type `function`. A mechanism's `releaseResponse()` method calls `target` to form privacy-preserving responses.

**Normed target range space.** Target $f$'s output space $\mathcal{B}$ is typically imbued with a norm, denoted $\|\cdot\|_{\mathcal{B}}$, needed for measuring the sensitivity of $f$'s outputs to input perturbation. diffpriv flexibly represents this norm within `DPMech` objects as described next.

**Sensitivity-induced privacy.** Many mechanisms achieve differential privacy by calibrating randomization to the sensitivity of target $f$. Insensitive targets need less response randomization. On a pair of neighboring databases $D, D' \in \mathcal{D}^n$ the *sensitivity* of $f$ is measured as $\Delta(D, D') = \|f(D) - f(D')\|_{\mathcal{B}}$. *Global sensitivity* is the largest such value $\overline{\Delta} = \sup_{D,D'} \|f(D) = f(D')\|_{\mathcal{B}}$ over all possible neighboring pairs. As we discuss in (**?**), a broad class of generic mechanisms, taking sensitivity $\Delta$ as a parameter, are *sensitivity-induced private*: for any neighboring pair $D, D' \in \mathcal{D}^n$ if $\Delta(D, D') \leq \Delta$ then the mechanism $M_\Delta$ run with parameter $\Delta$ achieves $\Pr(M_\Delta(D) \in R) \leq \exp(\epsilon) \cdot \Pr(M_\Delta(D') \in R)$ for all $R \subseteq \mathcal{R}$. When run with $\Delta = \overline{\Delta}$, this condition holds for all neighboring pairs: $M_{\overline{\Delta}}$ satisfies $\epsilon$-DP. Similarly for $(\epsilon, \delta)$-DP. In fact this is how differential privacy is typically proved for such generic mechanisms. `DPMech` objects can be initialized with a `sensitivity` argument, stored in an S4 slot of the same name. If the user provides a manually-derived global sensitivity bound $\overline{\Delta}$, then `releaseResponse()` responses preserve $\epsilon$- or $(\epsilon, \delta)$-DP (depending on the specific mechanism). We now demonstrate this use case.

### 2.1 Example: Laplace Mechanism Release of the Sample Mean

diffpriv implements Laplace (**?**), Gaussian (**?**), exponential (**?**), and Bernstein (**?**) mechanisms as `DPMech` subclasses `DPMechLaplace`, `DPMechGaussian`, `DPMechExponential`, and `DPMechBernstein`. An exponential example is included below. `DPMechLaplace` releases numeric vectors, adopting the $L_1$ norm (sum of absolutes) for $\|\cdot\|_{\mathcal{B}}$. The mechanism releases vectors in $\mathcal{R} = \mathcal{B} = \mathbb{R}^d$ by adding an i.i.d. sample of $d$ Laplace-distributed random variables

with means 0 and scales $\overline{\Delta}/\epsilon$ to $f(D)$ to achieve $\epsilon$-DP. `DPMechGaussian` also privatizes numeric responses but under $L_2$-sensitivity and weaker $(\epsilon, \delta)$-DP; `DPMechBernstein` leverages the Laplace mechanism to release multivariate real-valued functions.

We next demonstrate Laplace privatization of the sample mean on bounded data $\mathcal{D}^n = [0,1]^n$, for which $\mathcal{B}$ dimension `dims` is one. Global sensitivity is readily bounded as $1/n$.

```
library(diffpriv)
f <- function(X) mean(X) ## target function
n <- 100 ## dataset size
mechanism <- DPMechLaplace(target = f, sensitivity = 1/n, dims = 1)
D <- runif(n, min = 0, max = 1) ## the sensitive database in [0,1]^n
pparams <- DPParamsEps(epsilon = 1) ## desired privacy budget
r <- releaseResponse(mechanism, privacyParams = pparams, X = D)
cat("Private response r$response:", r$response,
  "\nNon-private response f(D):  ", f(D))

#> Private response r$response: 0.5244495
#> Non-private response f(D):   0.5343454
```

## 3. Sensitivity Sampling for Random Differential Privacy

When `target` global sensitivity is supplied as `sensitivity` within `DPMech` construction, responses are differentially private. Global sensitivity is known for *idealizations* of *e.g.*, coefficients for regularized logistic regression (**?**) and the SVM (**??**). In complex applications such as privatizing a software function, however, `target`'s global sensitivity may not be readily available. For such cases, diffpriv implements the sensitivity sampler of **?** which forms a high-probability estimate of `target` sensitivity by repeated probing of sensitivity on random neighboring database pairs, leveraging tools from empirical process theory. Like sensitivity estimates, resulting privacy holds with high probability.

**Definition 2 (?)** *A mechanism $M$ preserves* $(\epsilon, \gamma)$-random differential privacy *(with a corresponding form for $\epsilon, \delta, \gamma$) if* $\forall R \subseteq \mathcal{R}, \Pr(M(D) \in R) \leq \exp(\epsilon) \cdot \Pr(M(D') \in R)$ *holds with probability at least* $1 - \gamma$ *over random database pairs* $D, D'$.

While weaker than $\epsilon$-DP, RDP is arguably more natural than $(\epsilon, \delta)$-DP: The later safeguards all databases but not unlikely responses, while RDP protects against all responses but not pathological databases (as defined by the database sampling distribution). The sampling distribution can be anything meaningful *e.g.*, uniform, a Bayesian prior, a density from data privately fit by the Bernstein mechanism (**?**), etc.

The `DPMech` method `sensitivitySampler(object, oracle, n, m, gamma)` requires: a mechanism `object`; a function `oracle` which outputs i.i.d. random databases of given size `n` which should match the size of input data supplied later in calls to `releaseResponse()`; a sensitivity sample size `m`; and desired privacy confidence `gamma`. Either (but not both) of `m`, `gamma` can be omitted: the omitted resource will be optimized automatically. For example `m` taken small (few hundred) reflects limited sampling time; small `gamma` (*e.g.*, 0.05) prioritizes privacy. The sensitivity sampler calls appropriate `DPMech sensitivityNorm()` which implements $\Delta(D, D')$ for the mechanism's norm and stored `target`. New subclasses

of `DPMech` need only implement this method in order to take advantage of the sensitivity sampler. After `sensitivitySampler()`, subsequent `releaseResponse()` results have a privacy parameter slot of type `DPParamsGam` indicating response RDP.

### 3.1 Example: Frequent Characters with Sensitivity Sampling

All `DPMech` subclasses are sensitivity-induced private and can be sensitivity sampled. We demonstrate the exponential mechanism, which privately optimizes a given objective $s(r)$ of candidate response $r \in \mathcal{R}$. Its response distribution is proportional to $\exp(\epsilon \cdot s(r)/(2\Delta))$. Typically $s$ depends on input $D$, and so `DPMechExponential` is initialized with `target` that takes $D$ and outputs score function $s(r)$. That is, $\mathcal{B} = \mathbb{R}^{\mathcal{R}}$ is a real-valued function space on $\mathcal{R}$ and the class's `sensitivityNorm()` implements the sup-norm (*cf.* **?**). In practice, users supply `target` as an R closure as demonstrated below. Given `sensitivity` of `target`, the mechanism preserves $\epsilon$-DP; if `sensitivitySampler()` estimates sensitivity with some `gamma`, then confidence $\gamma =$`gamma` RDP is preserved.

We can apply these ideas to find the most frequent a–z character within the top-10 computer scientist names from Semantic Scholar, subject to individual privacy. Exponential privately maximizes total frequency. But without bounded name lengths, this function has unbounded global sensitivity. We therefore use the sensitivity sampler for $(1, 0.1)$-RDP, with an oracle that samples representative U.S. names with package `randomNames`.

```
library(randomNames) ## a package that generates representative random names
oracle <- function(n) randomNames(n)
D <- c("Michael Jordan", "Andrew Ng", "Andrew Zisserman","Christopher Manning",
       "Jitendra Malik", "Geoffrey Hinton", "Scott Shenker",
       "Bernhard Scholkopf", "Jon Kleinberg", "Judea Pearl")
n <- length(D)
f <- function(X) { function(r) sum(r == unlist(base::strsplit(X, ""))) }
rSet <- as.list(letters) ## the response set, letters a--z, must be a list
mechanism <- DPMechExponential(target = f, responseSet = rSet)
mechanism <- sensitivitySampler(mechanism, oracle = oracle, n = n, gamma = 0.1)
pparams <- DPParamsEps(epsilon = 1)
r <- releaseResponse(mechanism, privacyParams = pparams, X = D)
cat("Private response r$response: ", r$response,
  "\nNon-private f(D) maximizer:  ", letters[which.max(sapply(rSet, f(D)))])

#> Private response r$response:  e
#> Non-private f(D) maximizer:   e
```

## Acknowledgments

B. Rubinstein and F. Aldà acknowledge the support of the Australian Research Council (DE160100584) and the DFG Research Training Group GRK 1817/1 respectively.