

# fn cdp\_epsilon

Michael Shoemate

September 11, 2024

Proves soundness of `fn cdp_epsilon` in `cdp_epsilon.rs` at commit `0b8f4222` (outdated<sup>1</sup>). This proof is an adaptation of [subsection 2.3](#) of [\[CKS20\]](#).

## 1 Bound Derivation

**Definition 1.1.** (Privacy Loss Random Variable). Let  $M : \mathcal{X}^n \rightarrow \mathbb{Y}$  be a randomized algorithm. Let  $x, x' \in \mathcal{X}^n$  be neighboring inputs. Define  $f : \mathcal{Y} \rightarrow \mathbb{R}$  by  $f(y) = \log \left( \frac{\mathbb{P}[M(x)=y]}{\mathbb{P}[M(x')=y]} \right)$ . Let  $Z = f(M(x))$ , the privacy loss random variable, denoted  $Z \leftarrow \text{PrivLoss}(M(x)||M(x'))$ .

**Lemma 1.2.** [\[CKS20\]](#) Let  $\epsilon, \delta \geq 0$ . Let  $M : \mathcal{X}^n \rightarrow \mathcal{Y}$  be a randomized algorithm. Then  $M$  satisfies  $(\epsilon, \delta)$ -differential privacy if and only if

$$\delta \geq \mathbb{E}_{Z \leftarrow \text{PrivLoss}(M(x)||M(x'))} [\max(0, 1 - e^{\epsilon - Z})] \quad (1)$$

(2)

for all  $x, x' \in \mathcal{X}^n$  differing on a single element.

*Proof.* Fix neighboring inputs  $x, x' \in \mathcal{X}^n$ . Let  $f : \mathcal{Y} \rightarrow \mathbb{R}$  be as in [1.1](#). For notational simplicity, let  $Y = M(x)$ ,  $Y' = M(x')$ ,  $Z = f(Y)$  and  $Z' = -f(Y')$ . This is equivalent to  $Z \leftarrow \text{PrivLoss}(M(x)||M(x'))$ . Our first goal is to prove that

$$\sup_{E \subset \mathcal{Y}} \mathbb{P}[Y \in E] - e^\epsilon \mathbb{P}[Y' \in E] = \mathbb{E}[\max\{0, 1 - e^{\epsilon - Z}\}]. \quad (3)$$

For any  $E \subset \mathcal{Y}$ , we have

$$\mathbb{P}[Y' \in E] = \mathbb{E}[\mathbb{I}[Y' \in E]] = \mathbb{E}[\mathbb{I}[Y \in E] e^{-f(Y)}]. \quad (4)$$

This is because  $e^{-f(y)} = \frac{\mathbb{P}[Y=y]}{\mathbb{P}[Y'=y]}$ .

Thus, for all  $E \subset \mathcal{Y}$ , we have

$$\mathbb{P}[Y \in E] - e^\epsilon \mathbb{P}[Y' \in E] = \mathbb{E} \left[ \mathbb{I}[Y \in E] (1 - e^{\epsilon - f(Y)}) \right] \quad (5)$$

Now it is easy to identify the worst event as  $E = \{y \in \mathcal{Y} : 1 - e^{\epsilon - f(y)} > 0\}$ . Thus

$$\sup_{E \subset \mathcal{Y}} \mathbb{P}[Y \in E] - e^\epsilon \mathbb{P}[Y' \in E] = \mathbb{E} \left[ \mathbb{I}[1 - e^{\epsilon - f(Y)} > 0] (1 - e^{\epsilon - f(Y)}) \right] = \mathbb{E}[\max\{0, 1 - e^{\epsilon - Z}\}] \quad (6)$$

□

---

<sup>1</sup>See new changes with `git diff 0b8f4222..3687c79 rust/src/combinators/measure_cast/zCDP_to_approxDP/cdp_epsilon.rs`

**Theorem 1.3.** [CKS20] Let  $M : \mathcal{X}^n \rightarrow \mathcal{Y}$  be a randomized algorithm. Let  $\alpha \in (1, \infty)$  and  $\epsilon \geq 0$ . Suppose  $D_\alpha(M(x)||M(x')) \leq \tau$  for all  $x, x' \in \mathcal{X}^n$  differing in a single entry.<sup>2</sup> Then  $M$  is  $(\epsilon, \delta)$ -differentially private for

$$\delta = \frac{e^{(\alpha-1)(\tau-\epsilon)}}{\alpha-1} \left(1 - \frac{1}{\alpha}\right)^\alpha \quad (7)$$

*Proof.* Fix neighboring  $x, x' \in \mathcal{X}^n$  and let  $Z \leftarrow \text{PrivLoss}(M(x)||M(x'))$ . We have

$$\mathbb{E}[e^{(\alpha-1)Z}] = e^{(\alpha-1)D_\alpha(M(x)||M(x'))} \leq e^{(\alpha-1)\tau} \quad (8)$$

By 1.2, our goal is to prove that  $\delta \geq \mathbb{E}[\max\{0, 1 - e^{\epsilon-Z}\}]$ . Our approach is to pick  $c > 0$  such that  $\max\{0, 1 - e^{\epsilon-Z}\} \leq ce^{(\alpha-1)z}$  for all  $z \in \mathbb{R}$ . Then

$$\mathbb{E}[\max\{0, 1 - e^{\epsilon-Z}\}] \leq \mathbb{E}[ce^{(\alpha-1)z}] \leq ce^{(\alpha-1)\tau}. \quad (9)$$

We identify the smallest possible value of  $c$ :

$$c = \sup_{z \in \mathbb{R}} \frac{\max\{0, 1 - e^{\epsilon-z}\}}{e^{(\alpha-1)z}} = \sup_{z \in \mathbb{R}} e^{z-\alpha z} - e^{\epsilon-\alpha z} = \sup_{z \in \mathbb{R}} f(z) \quad (10)$$

where  $f(z) = e^{z-\alpha z} - e^{\epsilon-\alpha z}$ . We have

$$f'(z) = e^{z-\alpha z}(1-\alpha) - e^{\epsilon-\alpha z}(-\alpha) = e^{-\alpha z}(\alpha e^z - (\alpha-1)e^\epsilon) \quad (11)$$

Clearly  $f'(z) = 0 \iff e^z = \frac{\alpha-1}{\alpha}e^\epsilon \iff z = \epsilon - \log(1 - 1/\alpha)$ . Thus

$$c = f(\epsilon - \log(1 - 1/\alpha)) \quad (12)$$

$$= \left(\frac{\alpha}{\alpha-1}e^\epsilon\right)^{1-\alpha} - e^\epsilon \left(\frac{\alpha}{\alpha-1}e^\epsilon\right)^{-\alpha} \quad (13)$$

$$= \left(\frac{\alpha}{\alpha-1}e^\epsilon - e^\epsilon\right) \left(\frac{\alpha}{\alpha-1}e^{-\epsilon}\right)^\alpha \quad (14)$$

$$= \frac{e^\epsilon}{\alpha-1} \left(1 - \frac{1}{\alpha}\right)^\alpha e^{-\alpha\epsilon}. \quad (15)$$

Thus

$$\mathbb{E}[\max\{0, 1 - e^{\epsilon-Z}\}] \leq \frac{e^\epsilon}{\alpha-1} \left(1 - \frac{1}{\alpha}\right)^\alpha e^{-\alpha\epsilon} e^{(\alpha-1)\tau} = \frac{e^{(\alpha-1)(\tau-\epsilon)}}{\alpha-1} \left(1 - \frac{1}{\alpha}\right)^\alpha = \delta \quad (16)$$

□

**Corollary 1.** [CKS20] Let  $M : \mathcal{X}^n \rightarrow \mathcal{Y}$  be a randomized algorithm. Let  $\alpha \in (1, \infty)$  and  $\epsilon \geq 0$ . Suppose  $D_\alpha(M(x)||M(x')) \leq \tau$  for all  $x, x' \in \mathcal{X}^n$  differing in a single entry. Then  $M$  is  $(\epsilon, \delta)$ -differentially private for

$$\epsilon = \tau + \frac{\ln(1/\delta) + (\alpha-1)\ln(1-1/\alpha) - \ln(\alpha)}{\alpha-1} \quad (17)$$

*Proof.* This follows by rearranging 1.3. □

**Corollary 2.** Let  $M : \mathcal{X}^n \rightarrow \mathcal{Y}$  be a randomized algorithm satisfying  $\rho$ -concentrated differential privacy. Then  $M$  is  $(\epsilon, \delta)$ -differentially private for any  $0 < \delta \leq 1$  and

$$\epsilon = \inf_{\alpha \in (1, \infty)} \alpha\rho + \frac{\ln(1/\delta) + (\alpha-1)\ln(1-1/\alpha) - \ln(\alpha)}{\alpha-1} \quad (18)$$

*Proof.* This follows from 1 by taking the infimum over all divergence parameters  $\alpha$ . □

<sup>2</sup>This is the definition of  $(\alpha, \tau)$ -Rényi differential privacy.

## 1.1 Efficient computation of $\epsilon$

From 2, we have

$$\epsilon(\alpha) = \alpha\rho + \frac{\ln(1/\delta) + (\alpha - 1)\ln((\alpha - 1)/\alpha) - \ln(\alpha)}{\alpha - 1} \quad (19)$$

$$\epsilon'(\alpha) = \rho + \frac{\ln(\alpha\delta)}{(\alpha - 1)^2} \quad (20)$$

$$\epsilon''(\alpha) = \frac{2\alpha \ln(\alpha\delta) - \alpha + 1}{(\alpha - 1)^3\alpha} \quad (21)$$

Notice the curve is convex so long as

$$\delta < e^{1/2-1/(2\alpha)}/\alpha \quad (22)$$

Otherwise the curve is concave, with a non-negative derivative for any choice of  $\alpha$ :

$$\epsilon'(\alpha) = \rho + \frac{\ln(\alpha\delta)}{(\alpha - 1)^2} \geq \rho + \frac{\alpha - 1}{2\alpha(\alpha + 1)^2} \geq 0 \quad (23)$$

We can find the minimizer  $\alpha_*$  by conducting a binary search over the interval  $(1, \alpha_{max})$ , where  $\alpha_{max}$  is discovered via exponential search for a positive derivative.

## 2 Pseudocode

### Precondition

- Type Q must have trait Float.

### Implementation

```
1 def cdp_epsilon(rho: Q, delta: Q) -> Q:
2     if rho.is_sign_negative():
3         raise "rho must be non-negative"
4
5     if not delta.is_sign_positive():
6         raise "delta must be positive"
7
8     if rho.is_zero():
9         return 0
10
11     # checks if derivative is positive
12     def deriv_pos(a):
13         return rho > -log(a * delta) / (a - 1)**2
14
15     # find bounds
16     a_min = 1.01
17     a_max = 2
18     while not deriv_pos(a_max):
19         a_max *= 2
20
21     # optimize alpha
22     while True:
23         diff = a_max - a_min
24
25         a_mid = a_min + diff / _2
26
27         if a_mid == a_max or a_mid == a_min:
28             break
29
```

```

30     if deriv_pos(a_mid):
31         a_max = a_mid
32     else:
33         a_min = a_mid
34
35     # back out epsilon
36     a_m1 = a_max.inf_sub(_1)
37
38     numer = (a_m1.inf_div(a_max).inf_ln().inf_mul(a_m1)) \
39             .inf_sub(a_max.inf_ln()) \
40             .inf_add(delta.recip().inf_ln())
41
42     denom = a_max.neg_inf_sub(_1)
43
44     epsilon = a_max.inf_mul(rho).inf_add(numer.inf_div(denom))
45
46     return max(epsilon, 0)

```

### Postcondition

Either a valid epsilon is returned or an error is returned.

## 3 Proof

**Theorem 3.1.** For any possible setting of  $\rho$  and  $\delta$ , `cdp_epsilon` either returns an error, or an  $\epsilon$  such that any  $\rho$ -differentially private measurement is also  $(\epsilon, \delta)$ -differentially private.

*Proof.* The code always finds an  $\alpha_* \approx \mathbf{a\_max} \geq 1.01$ . Since  $\mathbf{a\_max} \in (1, \infty)$ , then by 2, any  $\rho$ -differentially private measurement is also  $(\epsilon(\mathbf{a\_max}), \delta)$ -differentially private. Define  $\epsilon_{cons}(\alpha)$  as a “conservative” function for computing  $\epsilon(\alpha)$ , where floating-point arithmetic is computed with conservative rounding such that  $\epsilon_{cons}(\alpha) \geq \epsilon(\alpha)$  for  $\forall \alpha \in (1, \infty)$ . Since  $\mathbf{epsilon} = \epsilon_{cons}(\mathbf{a\_max}) \geq \epsilon(\mathbf{a\_max})$ , then any  $(\epsilon(\mathbf{a\_max}), \delta)$ -differentially private measurement is also  $(\mathbf{epsilon}, \delta)$ -differentially private.  $\square$

## References

[CKS20] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. *CoRR*, abs/2004.00010, 2020.