

Generalized Resize Notes

This is a work in progress.

1 GOALS

The generalized resize component is a means of jointly achieving a few different goals:

1. guarantee known n ,
2. give users flexibility in how they trade off between bias and privacy usage, and
3. allow for a combination of c-stability and privacy amplification from subsampling.

2 ALGORITHM STATEMENT

The function will take the following inputs:

1. X : The private underlying data.
2. \tilde{n} : The size of the private underlying data.
3. n : The desired size of the new data.
4. p : The proportion of the underlying data that can be used to construct the new data.
Can be > 1 .
5. ...: Various arguments explaining imputation rules (not of interest for this doc)

Let $sample(Y, m)$ be a function that samples m elements from data set Y without replacement. Let $Aug(Y, m, \dots)$ be a function that imputes new elements independent of the data (using imputation parameters given by \dots) for a data set Y until it is of size m . The algorithm will look something like the following:

Algorithm 1 Generalized Resize: $\text{resize}(X, n, p, \text{neighboring}, \dots)$

```
1:  $c \leftarrow \lceil p \rceil$  ▷ sets c-stability property
2:  $s \leftarrow p/c$  ▷ sets subsampled_proportion property
3:  $X_c \leftarrow \bigcup_{i=1}^c X$  ▷ create new database of size  $c\tilde{n}$ , composed of  $c$  copies of  $X$ 
4: if neighboring == “replace one” then
5:    $m \leftarrow \lfloor s\tilde{n} \rfloor$  ▷ number of records that can be filled using subsampled private data
6: else if neighboring == “add/remove one” then
7:    $m \leftarrow \text{Binomial}(c\tilde{n}, s)$  ▷ number of records that can be filled using subsampled private data
8: end if
9:  $(\epsilon', \delta') \leftarrow \left( \log(1 + s(e^{c\epsilon} - 1)), s \left( \sum_{i=1}^{c-1} e^{i\epsilon} \right) \delta \right)$  ▷ privacy amplification via subsampling
10:  $X' \leftarrow \text{sample}(X_c, \max(m, n), \text{neighboring}) \cup (\text{Aug}(\emptyset, \max(0, n - m), \dots))$ 
11: return  $(X', \epsilon', \delta')$ 
```

The ϵ', δ' terms come from first applying the group privacy definition with group size c to the database X_c to get $(c\epsilon, \left(\sum_{i=1}^{c-1} e^{i\epsilon} \right) \delta)$ [Vad17] and then applying privacy amplification by subsampling results from Theorems 8 and 9 of [BBG18]. Note that, for the “replace one” definition, we could be using $\frac{m}{c\tilde{n}}$ instead of s in the privacy calculation. Using s gives us a very slightly worse privacy guarantee (the only difference is the $\lfloor \cdot \rfloor$ we used to get m), but is nice for consistency between the methods and not having to keep track of m as an extra property.

3 FUNCTIONAL PRIVACY PARAMETERS

We established in section 2 that a user asking for an (ϵ, δ) -DP guarantee will get an (ϵ', δ') -DP guarantee with respect to the original private data. What we’d really like, however, is for the user to ask for an (ϵ, δ) -DP guarantee and have the library come up with what we will call a *functional* (ϵ'', δ'') that will ensure (ϵ, δ) -DP on the original data. Any components that operate on the resized data will use the *functional* (ϵ'', δ'') internally instead of the parameters passed by the user.

Theorem 1. *A mechanism that respects*

$$\left(\frac{1}{c} \log \left(\frac{e^{\epsilon'} - 1}{s} + 1 \right), \frac{\delta'}{s \left(\sum_{i=1}^{c-1} e^{i\epsilon} \right)} \right) \text{-DP}$$

on the resized data respects (ϵ, δ) -DP on the true private data.

Proof. We know that an (ϵ, δ) -DP on the resized data corresponds to a

$$(\epsilon', \delta') = \left(\log(1 + s(e^{c\epsilon} - 1)), s \left(\sum_{i=1}^{c-1} e^{i\epsilon} \right) \delta \right)$$

guarantee on the private data. We just need to invert the function to find (ϵ, δ) in terms of (ϵ', δ') .

Let's start with ϵ, ϵ' :

$$\begin{aligned}\epsilon' &= \log(1 + s(e^{c\epsilon} - 1)) \\ e^{\epsilon'} &= 1 + s(e^{c\epsilon} - 1) \\ \frac{e^{\epsilon'} - 1}{s} &= e^{c\epsilon} - 1 \\ \frac{1}{c} \log\left(\frac{e^{\epsilon'} - 1}{s} + 1\right) &= \epsilon.\end{aligned}$$

We carry out a similar calculation for δ, δ' :

$$\begin{aligned}\delta' &= s \left(\sum_{i=1}^{c-1} e^{i\epsilon} \right) \delta \\ \frac{\delta'}{s \left(\sum_{i=1}^{c-1} e^{i\epsilon} \right)} &= \delta.\end{aligned}$$

□

So, in order for a mechanism to respect (ϵ, δ) -DP on the original data, it must respect $\left(\frac{1}{c} \log\left(\frac{e^{\epsilon}-1}{s} + 1\right), \frac{\delta}{s \left(\sum_{i=1}^{c-1} e^{i\epsilon}\right)}\right)$ -DP on the resized data.

We now present an mini-algorithm for finding the *functional* (ϵ, δ) .

Algorithm 2 Finding Functional (ϵ_f, δ_f) : `get_func_priv(p, ϵ, δ)`

```

1:  $c \leftarrow \lceil p \rceil$  ▷ sets c-stability property
2:  $s \leftarrow p/c$  ▷ sets subsampled_proportion property
3:  $\epsilon_f \leftarrow \frac{1}{c} \log\left(\frac{e^{\epsilon}-1}{s} + 1\right)$ 
4:  $\delta_f \leftarrow \frac{\delta}{s \left(\sum_{i=1}^{c-1} e^{i\epsilon}\right)}$ 
5: return  $(\epsilon_f, \delta_f)$ 
```

4 EXAMPLES

Let X be such that $\tilde{n} = |X| = 100$. We can look at a few examples of calls to the generalized resize function (which will return X') and check the behavior.

1. `resize(X, 150, 1, ...)`: X' will be made up of the 100 true elements of X and 50 imputed values. The functional privacy parameters are identical to the ones the user provides.
2. `resize(X, 100, 0.75, ...)`: X' will be made up of 75 true elements of X and 25 imputed values. The functional privacy parameters will benefit (lower noise) from amplification via subsampling.
3. `resize(X, 90, 1.5, ...)`: X' will be a random sample of $X \cup X$ of size 90. The functional privacy parameters will lead to greater noise than what the user provides, as they have to take into account the new c-stability of 2. This example is illustrative in that it shows that the functional privacy usage is affected only by p – it has nothing to do with the relative sizes of the X and X' .

REFERENCES

- [BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by sub-sampling: Tight analyses via couplings and divergences. *CoRR*, abs/1807.01647, 2018.
- [Vad17] Salil Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer, 2017.