Harvard Privacy Tools Project

Geometric Mechanism Notes

Christian Covington

February 3, 2020

1 Overview

This document is a write-up of extra notes regarding implementations of the Geometric mechanism in yarrow.

2 Simple Geometric Mechanism

2.1 Background

The *Simple Geometric Mechanism* is an implementation of the Geometric mechanism proposed in [GRS12]. For a counting query f with true value $f(d)$ and parameter value $\alpha \in (0, 1)$, the $\alpha-$geometric mechanism outputs $f(d) + \Delta$, where $\operatorname{supp} \Delta = \mathbb{Z}$ and

$$\Pr(\Delta = \delta) = \frac{1-\alpha}{1+\alpha}\alpha^{|\delta|}. \tag{2.1}$$

This mechanism respects pure differential privacy, with a privacy loss parameter of α. To accommodate privacy parameters outside of $(0, 1)$, we can choose a privacy parameter $\epsilon > 0$ and let $\alpha = e^{-\epsilon}$.

2.2 Approximate Implementation

Below is pseudocode that is pretty close to our implementation of the mechanism (more on the finer points later). This directly matches the α-Geometric mechanism from [GRS12].

Algorithm 1 (Almost) Simple Geometric Mechanism $M_{SG}(f(D), \epsilon)$

1: Let $f(D)$ be the count query we wish to privatize, ϵ be our privacy parameter and $\alpha = e^{-\epsilon}$.
2: $u \leftarrow \text{Unif}(0, 1)$
3: **if** $u < \frac{1-\alpha}{1+\alpha}$ **then**
4: return $f(D)$
5: **else**
6: $s \leftarrow$ uniformly random draw from $\{-1, 1\}$
7: $g \leftarrow \text{Geom}(1-\alpha)$ where $g \in \{1, 2, \ldots\}$
8: return $f(D) + s \cdot g$
9: **end if**

2.2.1 Proof that Algorithm 1 is equivalent to Equation (2.1)

First, note from Equation (2.1) that the mechanism returns $f(d)$ when $\Delta = 0$, which happens with probability $\frac{1-\alpha}{1+\alpha}$. This is reflected in line 3 of Algorithm 1.

Now, we have handled the case where $\Delta = 0$, so let's manipulate Equation (2.1) a bit more. For arbitrary $\delta \in \mathbb{Z} \setminus \{0\}$:

$$\begin{aligned}
\Pr(\Delta = \delta | \Delta \neq 0) &= \left(\frac{1}{1 - \Pr(\Delta = 0)}\right) \cdot \left(\frac{1-\alpha}{1+\alpha}\alpha^{|\delta|}\right) \\
&= \left(\frac{1}{1 - \frac{1-\alpha}{1+\alpha}}\right) \cdot \left(\frac{1-\alpha}{1+\alpha}\alpha^{|\delta|}\right) \\
&= \left(\frac{1+\alpha}{2\alpha}\right) \cdot \left(\frac{1-\alpha}{1+\alpha}\alpha^{|\delta|}\right) \\
&= \frac{1-\alpha}{2\alpha}\alpha^{|\delta|}.
\end{aligned}$$

We know that the noise induced by the geometric mechanism should be symmetric, so let's now consider only $\delta \in \mathbb{Z}^+$, with the knowledge that $\Pr(\Delta = \delta) = \Pr(\Delta = -\delta)$. This allows us to remove the factor of 2 from the denominator and remove the absolute value around n:

$$\begin{aligned}
\Pr(\Delta = \delta | \Delta \neq 0) &= \frac{1-\alpha}{\alpha}\alpha^{\delta} \\
&= \alpha^{\delta-1} \cdot (1-\alpha) \\
&= (1-p)^{\delta-1} p \text{ [for } p = (1-\alpha)].
\end{aligned}$$

Notice that this last statement is exactly the PDF of a Geom(p) defined on $\{1, 2, \ldots\}$ where $p = 1 - \alpha$. Thus, the combination of lines 6 and 7 from Algorithm 1 is sufficient to generate the modified distribution from Equation (2.1) where we condition on $\Delta \neq 0$.

2.3 Actual Implementation

To this point, we have assumed that the noise generated from our mechanism has support $\mathbb{Z}$ (we will call this the untruncated mechanism). This presents two major problems.

First, recall that we need to sample from a Geometric distribution within our mechanism. We do not want to use inverse transform sampling to do this, as doing so requires manipu-

lation of floating-point numbers that can lead to privacy violations.[1] Therefore, we induce a Geometric distribution by randomly sampling bits until we see a 1. If the distribution from which we are sampling has support $\mathbb{Z}$, we could hypothetically sample an arbitrarily large number of flips and not see a 1. We would like some kind of guarantee on the number of samples we need.

Perhaps more importantly, the untruncated mechanism will sometimes yield nonsensical answers. For a data set with known sample size n, the only reasonable answers to a counting query are $\mathcal{S} = \{0, 1, 2, \ldots, n\}$.

If the untruncated mechanism were to return a value outside of $\mathcal{S}$, then we can clip the value so that it is back in the set. This does not violate our privacy guarantee, as it is considered data-independent post-processing, and can only help us in terms of absolute error. We will refer to this as the *Truncated Geometric Mechanism*, as introduced in [GRS12]. We also consider what truncating the eventual mechanism output means for how we need to sample from the Geometric distribution.

Let's return to Algorithm 1, but include truncation so that it actually reflects the algorithm in yarrow.

Algorithm 2 Simple Geometric Mechanism $M_{SG}(f(D), \epsilon,$ count_min, count_max, EFC)

1: Let $f(D)$ be the count query we wish to privatize, ϵ be our privacy parameter, count_min be the minimum possible count (likely 0), count_max be the maximum possible count (probably n), and EFC (enforce_constant_time) be a boolean for whether or not we want to enforce our geometric sampling to always take the same number of steps.
2: Let $\alpha = e^{-\epsilon}$.
3: $u \leftarrow \text{Unif}(0, 1)$
4: **if** $u < \frac{1-\alpha}{1+\alpha}$ **then**
5: return $f(D)$
6: **else**
7: $s \leftarrow$ uniformly random draw from $\{-1, 1\}$
8: $g \leftarrow \text{Geom}_{Trunc}(1-\alpha)$ where $g \in \{1, 2, \ldots, \text{count\_max} - \text{count\_min}\}$
9: return $\max\Big(\text{count\_min}, \min\big(f(D) + s \cdot g, \text{count\_max}\big)\Big)$
10: **end if**

You can see in line 9 of Algorithm 2 where we clip the final output to the set

$$\mathcal{S} = \{\text{count\_min}, \text{count\_min} + 1, \ldots, \text{count\_max}\},$$

again keeping in mind that, in general, count_min and count_max will be 0 and n, respectively.

Our final step is to define Geom_{Trunc}. We know that our raw count $f(D)$ must be between count_min and count_max and that our mechanism will eventually return a result between those same bounds. Therefore, the absolute maximum noise we could ever add is $r =$ count_max$-$count_min; any more would always put us outside of the set $\mathcal{S}$ and eventually

[1] See [Mir12] and [Ilv19] for examples. [BV17] is the only place I have seen the known problems with floating-point numbers extended to their effects on inverse transform sampling.

be clipped back into the set. Therefore, we can sample at most r bits when generating the draw from the Geometric distribution. If we have sampled r bits and not seen a 1, then we can set $g = r$ without affecting our eventual result.

2.4 More notes on truncation

We noted above that truncation can only help our accuracy (in terms of ℓ_1 error), but it is very possible that our accuracy remains poor even after truncation. Imagine that we have data D with $n = 30$ and want to release a counting query on some predicate ϕ for which

$$\phi(D) = \left(\sum_{i=1}^{n} \mathbb{1}[\phi(d_i) = 1] \right) = 20,$$

where the d_i are the elements of D. If we wanted to release a differentially private version of $\phi(D)$, we could use the Geometric mechanism. The distribution of answers we would get for the mechanism, with a privacy parameter of $\epsilon = 0.1$ are shown in orange below.[2]

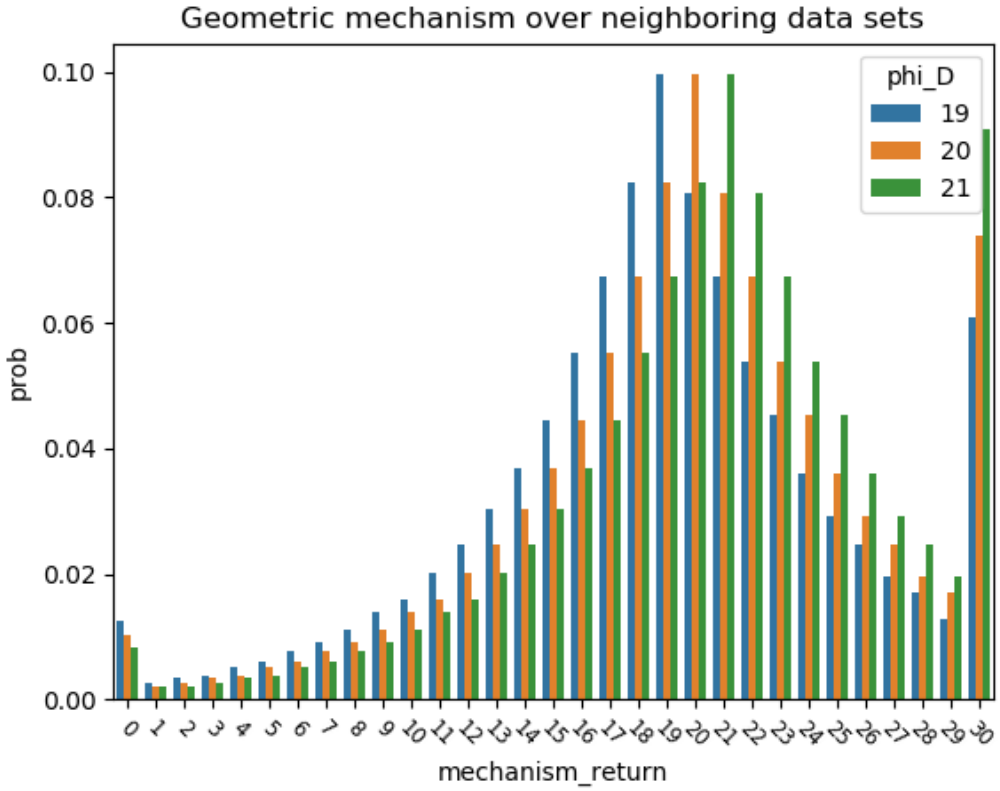

Figure 2.1: Truncated Geometric Mechanism Output

With this strategy, we are getting as bad an answer as possible (0) with probability ≈ 0.075 and are getting answers at the extreme ends of our support (0 and 30) with probability ≈ 0.25. We would like to see if there is a way to maintain the DP properties of our mechanism, while also concentrating more of the mass of this distribution around "good answers".

[2] We will come back to why we plotted results for $\phi(D) = 19, 20$, and 21.

One seemingly reasonably way to do this would be to simply ignore any events that produce truncated noise, that is if our generated noise Δ is either less than $-\phi(D)$ or greater than $n - \phi(D)$. We see that

$$\Pr\left(\Delta < -\phi(D)\right) = \frac{\alpha^{\phi(D)+1}}{1+\alpha}$$
$$\Pr\left(\Delta > n - \phi(D)\right) = \frac{\alpha^{n-\phi(D)+1}}{1+\alpha}$$

Let T_D be the event that our mechanism produces noise that must be truncated. Then,

$$\Pr(T_D) = \Pr\left(\Delta < -\phi(D)\right) + \Pr\left(\Delta > n - \phi(D)\right) = \frac{\alpha^{\phi(D)+1} + \alpha^{n-\phi(D)+1}}{1+\alpha}$$

We could think about creating a noise distribution in which we never produce noise that needs to be truncated, and the probability that we produce any allowable amount of noise is increased by a factor of $\frac{1}{1-\Pr(T_D)}$. This leads to a problem though; $\Pr(T_D)$ is a function of our data, and thus the multiplicative scaling factor will differ between neighboring data sets. For pure DP, we need the probability of all outcomes on neighboring data sets to be bounded by e^{ϵ}; we know that the regular geometric mechanism satisfies this, but if we are scaling probabilities on neighboring data sets by different factors, we can no longer be sure that the bound holds. We would like to be able to reason about the extent to which these different scaling factors could affect our privacy guarantee. We will attempt to do so by considering the maximum extent to which the scaling factors differ for given values of α and n. Note that we need to remove the dependence on $\phi(D)$ in order to retain our DP guarantee.

Let D', D'' be data sets that neighbor D such that $f(D') + 1 = f(D) = f(D'') - 1$. Then we have

$$\Pr(T_{D'}) = \frac{\alpha^{\phi(D)+2} + \alpha^{n-\phi(D)}}{1+\alpha}$$
$$\Pr(T_{D''}) = \frac{\alpha^{\phi(D)} + \alpha^{n-\phi(D)+2}}{1+\alpha}.$$

Let's now consider the the quantities

$$L_{\phi(D)} = \frac{\Pr(T_D)}{\Pr(T_{D'})} = \frac{\alpha^{2\phi(D)+1} + \alpha^{n+1}}{\alpha^{2\phi(D)+2} + \alpha^{n}} \tag{2.2}$$
$$U_{\phi(D)} = \frac{\Pr(T_D)}{\Pr(T_{D''})} = \frac{\alpha^{2\phi(D)+1} + \alpha^{n+1}}{\alpha^{2\phi(D)} + \alpha^{n+2}}.$$

We cannot use these quantities directly, as they will leak extra information about our data. However, if we can get global upper and lower bounds on both L and U, then we will have bounds on the extent to which $\Pr(T)$ can differ on arbitrary neighboring data sets. Let's start by looking at the partial derivatives of L, U with respect to $\phi(D)$,

$$\frac{\partial L_{\phi(D)}}{\partial \phi(D)} = -\frac{2(\alpha^2-1)\cdot\ln(\alpha)\cdot\alpha^{2\phi(D)+n+1}}{(\alpha^{2\phi(D)+2}+\alpha^n)^2} < 0$$
$$\frac{\partial U_{\phi(D)}}{\partial \phi(D)} = \frac{2(\alpha^2-1)\cdot\ln(\alpha)\cdot\alpha^{2\phi(D)+n+1}}{(\alpha^{2\phi(D)}+\alpha^{n+2})^2} > 0.$$

We know the sign of each of those because $\alpha \in (0,1)$ implies that the numerator is positive, and the denominator is always positive because it is a quantity squared. Therefore, we

know that both $L_{\phi(D)}$ and U are monotonic in $\phi(D)$, so we can just consider the values of $L_{\phi(D)}$ and $U_{\phi(D)}$ for our most extreme values, where $\phi(D) \in \{0, n\}$. We know then that, for all $n > 0$:

$$L_n = \frac{\alpha^{2n+1} + \alpha^{n+1}}{\alpha^{2n+2} + \alpha^n} \leq L_{\phi(D)} \leq \frac{\alpha^3 + \alpha^{n+1}}{\alpha^4 + \alpha^n} = L_1$$
$$U_0 = \frac{\alpha + \alpha^{n+1}}{1 + \alpha^{n+2}} \leq U_{\phi(D)} \leq \frac{\alpha^{2n-1} + \alpha^{n+1}}{\alpha^{2n-2} + \alpha^{n+2}} = U_{n-1}.$$

Note that we do not define L_0 or U_n because if D is such that $\phi(D) = 0$ or $\phi(D) = n$, there are no data sets that produce a smaller (or larger) count, respectively.

Because $L_n, U_0 < 1 < L_1, U_{n-1}$ and we want the maximum overall difference in our scaling factor, it is sufficient to find

$$M = \max\left(\frac{1}{L_n}, \frac{1}{U_0}, L_1, U_{n-1}\right).$$

Each of these is a function only of α and n, where $\alpha = e^{-\epsilon}$, so we can calculate M from only our privacy loss parameter and sample size. So, if the probabilities outputs on neighboring data sets were bounded by e^ϵ for the regular geometric, they are now bounded by $M \cdot e^\epsilon$. Noting that $M \cdot e^\epsilon = e^{\ln(M)+\epsilon}$, we see that if we use ϵ as our privacy loss parameter, we end up getting a guarantee as if we asked for $e^{\ln(M)+\epsilon}$. Likewise, if we truly want an ϵ guarantee, we could parameterize the geometric with $\epsilon' = \epsilon - \ln(M)$.

One plausible way to use this is to parameterize the *Truncated Geometric Mechanism* with ϵ' rather than ϵ and simply reject any mechanism return value that is outside of our range of feasible counts. In my testing, this method ended up being pretty low-utility, inducing a nearly uniform distribution of outputs over the set of possible counts.

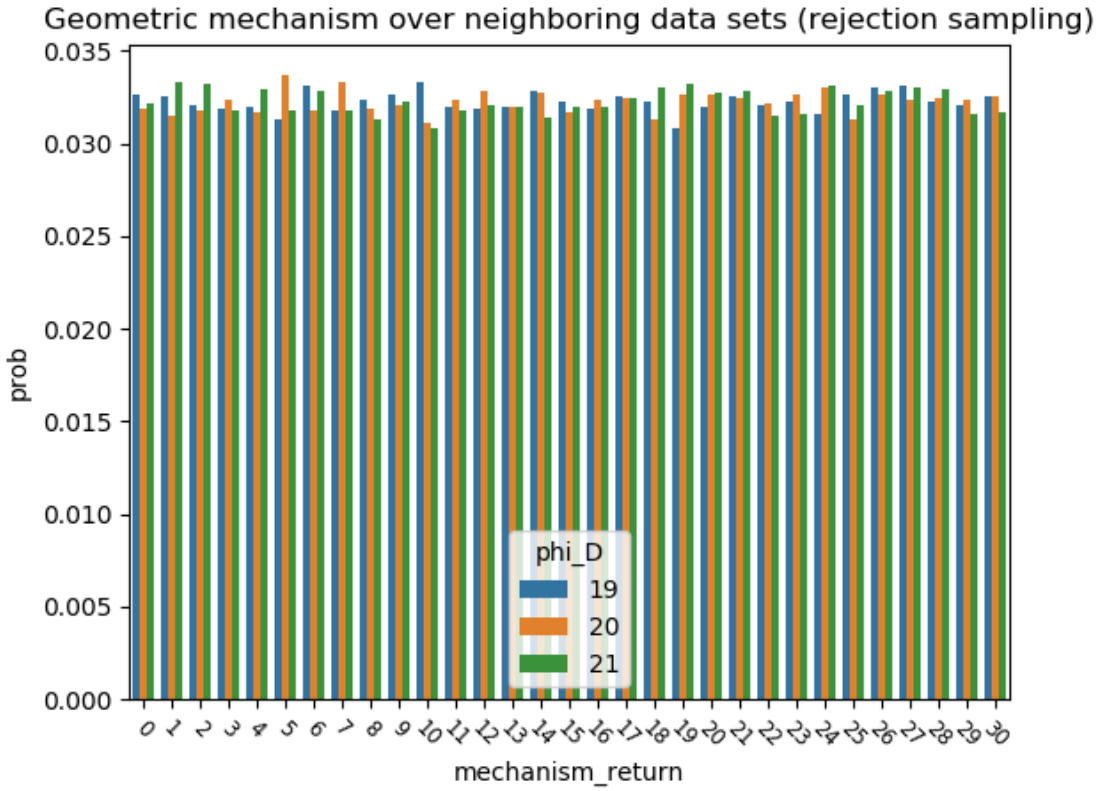

Figure 2.2: Truncated Geometric Mechanism Output with Rejection Sampling

The problem appears to be with ϵ' – for small ϵ, the additive change by $\ln(M)$ is, multiplicatively, quite large, and the noise we need to add increases dramatically. For large ϵ, we typically do not need to worry about pushing mass toward the center of the distribution, as it is already there.

One area I thought we might be able to improve was in the calculation of M. Earlier, we made a worst-case assumption about our data set D in order to remove the dependence of M on D. One could imagine, however, releasing a DP version of some function on D and having M have a dependence on that function. I tried releasing a DP estimate of $\phi(D)$, which we call $\phi(\hat{D})$ using the Exponential mechanism and using that estimate as input for L, U from Equation 2.2. However, because we are not actually using $\phi(D)$, the calculation of M based on $\phi(\hat{D})$ may not actually be an upper bound on the difference in the scaling factor for our real data.

This is a problem I do not know how to solve. We cannot restrict the feasible set for the Exponential mechanism to values of $\phi(\hat{D})$ for which the associated M is an actual upper bound on the actual scaling factor because the choice of feasible set must be independent of the data. I considered creating a score function for the Exponential mechanism that greatly penalizes values of $\phi(\hat{D})$ for which M is not an upper bound on the actual scaling factor, but this makes the sensitivity of the score function very difficult to reason about without just assuming a trivial worst-case.

References

[BV17] Victor Balcer and Salil Vadhan. Differential privacy on finite computers. *arXiv preprint arXiv:1709.05396*, 2017.

[GRS12] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693, 2012.

[Ilv19] Christina Ilvento. Implementing the exponential mechanism with base-2 differential privacy, 2019.

[Mir12] Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.