Privacy Doona: Why We Should Hide Among The Clones

(Written for the DifferentialPrivacy.org website)

Clément Canonne

May 2022

In this blog post, we will discuss a recent(ish) result of Feldman, McMillan, and Talwar [FMT21], which provides an improved and simple analysis of the so-called "amplification by shuffling" formally connecting local privacy (LDP) and shuffle privacy.[1] Now, I'll assume the reader is familiar with both LDP and Shuffle DP: if not, a quick-and-dirty refresher (with less quick, and less dirty references) can be found here, and of course there is also Albert Cheu's excellent survey on Shuffle DP [Che21].

I will also ignore most of the historical details, but it is worth mentioning that [FMT21] is not the first paper on this "amplification by shuffling," (which, for local reasons, I'll just call a *privacy doona*) but rather is the culmination of a rather long line of work involving many cool ideas and papers, starting with [CSU$^+$19, EFM$^+$19]: I'd refer the reader to Table 1 in [FMT21] for an overview.

Alright, now that the caveats are behind us, what *is* "amplification by shuffling"? In a nutshell, it is capturing the (false!) intuition that "anonymization provides privacy" (which, again, is false! Don't do this!) and making it... less false. The idea is that while *anonymization does not provide in itself any meaningful privacy guarantee*, it can *amplify existing, rigorous privacy guarantee.* So if I start with a somewhat lousy LDP guarantee, but then all the messages sent by all users are completely anonymized, then my lousy LDP guarantee suddenly gets *much* stronger (roughly speaking, the ε parameter goes down with the square root of of the number of users involved). Which is wonderful! Let's see what this means, quantitatively.

The result of Feldman, McMillan, and Talwar. Here, we will focus on the simpler case of *noninteractive* protocols (one-shot messages from the users to the central server, no funny business with messages going back and forth); which is conceptually simpler to state and parse, still very rich and interesting, and, well, very relevant in practice (being the easiest and cheapest to deploy). If you want the results in their full glorious generality, though, they are in the paper.

What the main theorem of [FMT21] is saying for this noninteractive setting can then be stated as follows: if I have an ε_L-*locally private* (LDP) protocol for a task, where all n users pass their

[1] The title of this post is a reference to the title of [FMT21], "Hiding Among The Clones," and to the notion of *privacy blanket* introduced by Balle, Bell, Gascón, and Nissim [BBGN19]. Intuitively, the "amplification by shuffling" paradigm can be seen as anonymizing the messages from local randomizers, whose message distribution can be mathematically decomposed as a mixture of "noise distribution not depending on the user's input" and "distribution actually depending on their input." As a result, each user randomly sends a message from the first or second distribution of the mixture. But the shuffling then hides the informative messages (drawn from the second part of the mixture) among the non-informative (noise) ones: so the noise messages end up providing a "privacy blanket" in which sensitive information is safely and soundly wrapped.

data through the same randomizer (algorithm) R and send the resulting message $y_i \leftarrow R(x_i)$, then just permuting the messages $y_1 \dots, y_n$ immediately gives an (ε, δ)-*shuffle* private protocol for the same task, for any pair (ε, δ) which satisfies

$$\varepsilon \leq \log\left(1 + 16\frac{e^{\varepsilon_L} - 1}{e^{\varepsilon_L} + 1}\sqrt{\frac{e^{\varepsilon_L}\log\frac{4}{\delta}}{n}}\right) \tag{1}$$

as long as $n \gg e^{\varepsilon_L}\log(1/\delta)$. That is quite a lot to parse, though: what does this actually *mean*?

First, the assumption that all users have the same randomizer (or at least cannot be distinguished by their randomizer) is quite natural: if they didn't, then we wouldn't be able to say anything in general, since the randomizer they use could just give away their identity completely. For instance, as an extreme case, the randomizer of user i could just append i to the message (it's OK, still LDP!), and then shuffling achieves exactly nothing: we know who sent what. So OK, asking for all randomizers to be the same is not really a restriction.

Second, each user only sends one message, and this preserves its length (we just shuffled the messages, didn't modify them!). So if you start with an LDP protocol with amazing features XYZ (e.g., the messages are 1-bit long, or users don't share a random seed, or the randomizers run in time $O(1)$), then the shuffle protocol enjoys exactly the same properties. (It only enjoys naturally some *robustness*, in the sense that if 10% if the n users maliciously deviate from the protocol, they can't really jeopardize the privacy of the remaining 90% of users.[2] Which is... good.)

Third, this is inherently approximate DP. Here we started with pure LDP (you can also extend that to approximate LDP) and ended up with approximate Shuffle DP: this is not a mistake, that's how it is. I am not a purist (erm) myself, and that looks more than good enough to me; but if you seek pure Shuffle DP, then this result is not the droid you're looking for.

Alright, *what* is this guarantee stated in (1) giving us? Let's interpret the expression in (1) in two parameter regimes, focusing on ε (fixing some small $\delta > 0$). If we start with $\varepsilon_L \ll 1$ for our LDP randomizers R, then a first-order Taylor expansion shows that we get

$$\varepsilon \approx \varepsilon_L \cdot 8\sqrt{\frac{\log\frac{4}{\delta}}{n}} \tag{2}$$

so that *shuffling improved our privacy parameter by a factor* $\sqrt{n}$.[3] 😲 This is great! With more users, comes more privacy!

But that was starting with small ε_L, that is, already pretty good privacy guarantees for our LDP "building block" R. What happens if we start with "somewhat lousy" privacy guarantees, that is, $\varepsilon_L \gg 1$? Do we get anything interesting then? Another Taylor expansion (everything is a Taylor expansion) shows us that, then,

$$\varepsilon \approx \log\left(1 + 8\sqrt{\frac{e^{\varepsilon_L}\log\frac{4}{\delta}}{n}}\right) \tag{3}$$

[2] More specifically, they can completely jeopardize the *utility* (accuracy) of the result, but in terms of privacy, all they can do is slightly reduce it: if 10% of users are malicious, the remaining 90% still get the privacy amplification of guarantee of (1), but with $0.9n$ instead of n.

[3] Of course, we started with a local privacy guarantee, and ended up with a shuffle privacy guarantee: so the two are incomparable, and one has to interpret this "amplification" in that context.

or, put differently,

$$\varepsilon \approx 8e^{\varepsilon_L/2}\sqrt{\frac{\log\frac{4}{\delta}}{n}} \tag{4}$$

That's a bit harder to interpret, but that seems... useful? It is: let us see how much, with a couple examples.

Learning. The first one is distribution learning, a.k.a. density estimation: you have n i.i.d. samples (one per user) from an unknown probability distribution $\mathbf{p}$ over a discrete domain of size k, and your goal is to output an estimate $\widehat{\mathbf{p}}$ such that, with high (say, constant) probability, $\mathbf{p}$ and $\widehat{\mathbf{p}}$ are close in *total variation distance*:

$$\mathrm{TV}(\mathbf{p},\widehat{\mathbf{p}}) = \sup_{S\subseteq[k]}(\mathbf{p}(S)-\widehat{\mathbf{p}}(S)) \leq \alpha$$

(if total variation distance seems a bit mysterious, it's exactly half the ℓ_1 distance between the probability mass functions). We know how to solve this problem in the non-private setting: $n=\Theta\big(\frac{k}{\alpha^2}\big)$ samples are necessary and sufficient. We know how to solve this problem in the (central) DP setting: $n=\Theta\big(\frac{k}{\alpha^2}+\frac{k}{\alpha\varepsilon}\big)$ samples are necessary and sufficient [DHS15]. We know how to solve this problem in the LDP setting:

$$n=\Theta\left(\frac{k^2}{\alpha^2(e^\varepsilon-1)^2}+\frac{k^2}{\alpha^2 e^\varepsilon}+\frac{k}{\alpha^2}\right) \tag{5}$$

samples are necessary and sufficient [ASZ19] (note that the first term is just $k/(\alpha^2\varepsilon^2)$ for small ε). Now, as they say in Mulan: *let's make a shuffle DP algo out of you.*

If we want to achieve (ε,δ)-shuffle DP, we need to select ε_L. Based on (3) and (4), and ignoring pesky constants we will choose it so that

$$\varepsilon_L \approx \varepsilon\sqrt{\frac{n}{\log(1/\delta)}} \qquad \text{or} \qquad e^{\varepsilon_L}\approx \varepsilon^2\cdot\frac{n}{\log(1/\delta)}\,. \tag{6}$$

depending on whether $\frac{\varepsilon^2 n}{\log(1/\delta)} \geq 1$. Plugging that back in (5), we see that the first case corresponds to the first term (small ε_L) and the second to the second term ($\varepsilon_L \geq 1$), and overall the condition on n for the original LDP algorithm to successful learn the distribution becomes

$$n \gtrsim \frac{k^2}{\alpha^2(e^{\varepsilon_L}-1)^2}+\frac{k^2}{\alpha^2 e^{\varepsilon_L}}+\frac{k}{\alpha^2} \approx \frac{k^2\log(1/\delta)}{\alpha^2\varepsilon^2 n}+\frac{k^2\log(1/\delta)}{\alpha^2\varepsilon^2 n}+\frac{k}{\alpha^2} \approx \frac{k^2\log(1/\delta)}{\alpha^2\varepsilon^2 n}+\frac{k}{\alpha^2}$$

(where $\gtrsim$ means "let's ignore constants"). There is an n in the RHS as well, so reorganizing and handling the two terms separately the condition on n becomes

$$n \gtrsim \frac{k\sqrt{\log(1/\delta)}}{\alpha\varepsilon}+\frac{k}{\alpha^2}$$

which... is great? We immediately get a sample complexity $O\Big(\frac{k}{\alpha^2}+\frac{k\sqrt{\log(1/\delta)}}{\alpha\varepsilon}\Big)$ in the shuffle DP model, which (ignoring the $\sqrt{\log(1/\delta)}$) matches the one in the *central* DP setting!

tl;dr: Taking an optimal LDP algorithm and just shuffling the messages *immediately* gives an optimal shuffle DP algorithm, no extra work needed.

(Uniformity) Testing. Alright, maybe it was a fluke? Let's look at another "basic" problem close to my heart: we don't want to learn the probability distribution $\mathbf{p}$, just test whether it is actually *the* uniform distribution[4] $\mathbf{u}$ on the domain $[k] = \{1, 2, \dots, k\}$. So if $\mathbf{p} = \mathbf{u}$, you've got to say "yes" with probability at least 2/3, and if $\mathrm{TV}(\mathbf{p}, \mathbf{u}) > \alpha$, then you need to say "no" with probability at least 2/3.

This is also well understood in the non-private setting ($n = \Theta(\sqrt{k}/\alpha^2)$) [Pan08] [see also my upcoming survey], in the central DP setting ($n = \Theta\Big(\frac{\sqrt{k}}{\alpha^2} + \frac{\sqrt{k}}{\alpha\sqrt{\varepsilon}} + \frac{k^{1/3}}{\alpha^{4/3}\varepsilon^{2/3}} + \frac{1}{\alpha\varepsilon}\Big)$) [ASZ18, ADR18], and in the LDP setting, where the result differs on whether the users can communicate or share a common random seed

$$n = \Theta\left(\frac{k}{\alpha^2(e^\varepsilon - 1)^2} + \frac{k}{\alpha^2 e^{\varepsilon/2}} + \frac{\sqrt{k}}{\alpha^2}\right) \tag{7}$$

or not

$$n = \Theta\left(\frac{k^{3/2}}{\alpha^2(e^\varepsilon - 1)^2} + \frac{k^{3/2}}{\alpha^2 e^{\varepsilon}} + \frac{\sqrt{k}}{\alpha^2}\right) \tag{8}$$

as established in a sequence of papers [ACT20, AJM20, ACF$^+$21, ACL$^+$22, CL22].

Now, say you want an (ε, δ)-shuffle DP algorithm for uniformity testing, but don't want to design one from scratch (though it *is* possible to do so, and some did [BCJM21, CL22, CY21]). Let's say you want to look at the "no-common-random-seed-shared-by-users" model (a.k.a. *private-coin* setting): so you stare at the corresponding LDP communication complexity, (8), and try to choose ε_L to start with before shuffling. This will be the same as in the learning example (i.e., (6)): based on (3) and (4), we will set

$$\varepsilon_L \approx \varepsilon\sqrt{\frac{n}{\log(1/\delta)}} \qquad \text{or} \qquad e^{\varepsilon_L} \approx \varepsilon^2 \cdot \frac{n}{\log(1/\delta)}\,. \tag{9}$$

depending on whether $\frac{\varepsilon^2 n}{\log(1/\delta)} \geq 1$. Plugging this back in (8) and quickly checking which case corresponds to each term, we then easily get that for our algorithm to correctly solve the uniformity testing problem, it suffices that the sample complexity (number of users) n satisfies

$$n \gtrsim \frac{k^{3/2}}{\alpha^2(e^{\varepsilon_L} - 1)^2} + \frac{k^{3/2}}{\alpha^2 e^{\varepsilon_L}} + \frac{\sqrt{k}}{\alpha^2} \approx \frac{k^{3/2}\log(1/\delta)}{\alpha^2\varepsilon^2 n} + \frac{\sqrt{k}}{\alpha^2}$$

which, reorganizing and solving for n, means that it suffices to have

$$n \gtrsim \frac{k^{3/4}\sqrt{\log(1/\delta)}}{\alpha\varepsilon} + \frac{\sqrt{k}}{\alpha^2}\,.$$

And, *voilá*! Even better, we also have strong evidence to suspect that this sample complexity $O\Big(\frac{k^{3/4}\sqrt{\log(1/\delta)}}{\alpha\varepsilon} + \frac{\sqrt{k}}{\alpha^2}\Big)$ is tight among all private-coin algorithms.[5]

[4] You can here replace uniform by any known distribution $\mathbf{q}$ of your choosing, that doesn't change the question (and result), but uniform is nice.

[5] As long as one is happy with approximate DP. One can achieve that in pure DP as well, but it's a bit more complicated [CY21].

Now, if you wanted to look at *public-coin* shuffle DP protocols (with a common random seed available), then you would start with an optimal public-coin LDP algorithm (and look at (7)), and setting ε_L the same way you'd get a shuffle DP algorithm with sample complexity

$$n = O\Big(\frac{k^{2/3}\log^{1/3}(1/\delta)}{\alpha^{4/3}\varepsilon^{2/3}} + \frac{\sqrt{k\log(1/\delta)}}{\alpha\varepsilon} + \frac{\sqrt{k}}{\alpha^2}\Big)$$

which, well, is *also* strongly believed to be optimal!

tl;dr: Here again, taking an optimal off-the-shelf LDP algorithm and just shuffling the messages *immediately* gives an optimal shuffle DP algorithm, no extra work needed.

Conclusion. I hope the above convinced you of how useful this privacy amplification can be: from an optimal LDP algorithm, featuring any extra appealing characteristics you like, *just adding an extra shuffling step as postprocessing* yields an (often optimal? At least good) shuffle DP algorithm, *with the same characteristics* and built-in robustness against malicious users.

All you need is to make sure that your starting point, the LDP algorithm satisfies a couple things: (1) all users have the same randomizer,[6] and (2) it works in all regimes of ε (both high-privacy, $\varepsilon \leq 1$, *and* low-privacy, $\varepsilon \gg 1$). Once you've got this, Bob's your uncle! You get shuffle DP algorithms for free.

It is not only appealing from a theoretical point of view, by the way! The authors of the paper worked hard to make their empirical analysis compelling as well, and their code is available on GitHub. But more importantly, from a practitioner's point of view, this means it is enough to design, implement, and test *one* algorithm (the LDP one we start with) to automatically get a trusted one in the shuffle DP model as well: this reduces the risks of bugs, security failures, the amount of work spending tuning, testing...
So yes, whenever possible, we *should* hide among the clones!

References

[ACF+21] Jayadev Acharya, Clément L. Canonne, Cody Freitag, Ziteng Sun, and Himanshu Tyagi. Inference under information constraints III: local privacy constraints. *IEEE J. Sel. Areas Inf. Theory*, 2(1):253–267, 2021. 4

[ACL+22] Jayadev Acharya, Clément L. Canonne, Yuhan Liu, Ziteng Sun, and Himanshu Tyagi. Interactive inference under information constraints. *IEEE Trans. Inf. Theory*, 68(1):502–516, 2022. 4

[ACT20] Jayadev Acharya, Clément L. Canonne, and Himanshu Tyagi. Inference under information constraints I: Lower bounds from chi-square contraction. *IEEE Trans. Inform. Theory*, 66(12):7835–7855, 2020. 4

[ADR18] Maryam Aliakbarpour, Ilias Diakonikolas, and Ronitt Rubinfeld. Differentially private identity and equivalence testing of discrete distributions. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 169–178. PMLR, 2018. 4

[6] This is not such a big assumption usually, and there are somewhat-general ways to get to that using a logarithmic factor in the number of users.

[AJM20] Kareem Amin, Matthew Joseph, and Jieming Mao. Pan-private uniformity testing. In *COLT*, volume 125 of *Proceedings of Machine Learning Research*, pages 183–218. PMLR, 2020. 4

[ASZ18] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. Differentially private testing of identity and closeness of discrete distributions. In *NeurIPS*, pages 6879–6891, 2018. 4

[ASZ19] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pages 1120–1129. PMLR, 2019. 3

[BBGN19] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In *CRYPTO (2)*, volume 11693 of *Lecture Notes in Computer Science*, pages 638–667. Springer, 2019. 1

[BCJM21] Victor Balcer, Albert Cheu, Matthew Joseph, and Jieming Mao. Connecting robust shuffle privacy and pan-privacy. In *SODA*, pages 2384–2403. SIAM, 2021. 4

[Che21] Albert Cheu. Differential privacy in the shuffle model: A survey of separations. *CoRR*, abs/2107.11839, 2021. 1

[CL22] Clément L. Canonne and Hongyi Lyu. Uniformity testing in the shuffle model: Simpler, better, faster. In *SOSA*, pages 182–202. SIAM, 2022. 4

[CSU$^+$19] Albert Cheu, Adam D. Smith, Jonathan R. Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In *EUROCRYPT (1)*, volume 11476 of *Lecture Notes in Computer Science*, pages 375–403. Springer, 2019. 1

[CY21] Albert Cheu and Chao Yan. Pure differential privacy from secure intermediaries. *CoRR*, abs/2112.10032, 2021. 4

[DHS15] Ilias Diakonikolas, Moritz Hardt, and Ludwig Schmidt. Differentially private learning of structured discrete distributions. In *NIPS*, pages 2566–2574, 2015. 3

[EFM$^+$19] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, pages 2468–2479. SIAM, 2019. 1

[FMT21] Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling. In *FOCS*, pages 954–964. IEEE, 2021. 1

[Pan08] Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Trans. Inform. Theory*, 54(10):4750–4755, 2008. 4