DSAID Data Privacy Protection Capability Centre (DPPCC)

from

GOVTECH SINGAPORE

Sharing Data with Differential Privacy and A Data Practitioner's Guide to Benchmarking Differential Privacy Python Tools

Author:
Anshu Singh, Research Engineer, GovTech

Special thanks:
Syahri Ikram, Data Engineer, GovTech
Alan Tang, Lead Product Manager of Data Privacy Protection, GovTech
Dr Ghim Eng Yap, Director and Functional Lead of Data Engineering and Data Privacy Protection, GovTech
Dr Damien Desfontaines, Staff Scientist, Tumult Labs
Dr Xiaokui Xiao, Professor, School of Computing, National University of Singapore

❝ Human intuition about what is private is not especially good. Computers are getting more and more sophisticated at pulling individual data out of things that a naive person might think are harmless. ❞

– Frank McSherry, Co-inventor of differential privacy, Co-founder and Chief Scientist, Materialize, Inc.

CONTENTS

1. EXECUTIVE SUMMARY

Across the globe, news related to data protection breaches focuses on cybersecurity incidents. These stories often involve security disasters such as data theft, system hacking, or phishing attacks. Rarely, if ever, is information reported on privacy violations resulting from data misuse, including attacks on anonymised datasets previously believed to be safe. Globally, *traditional anonymisation* techniques like *aggregation, suppression,* and *k-anonymity* have led to real-world privacy debacles and increased privacy risks. Recently, researchers discovered a new vulnerability in the widely used *k-anonymity* method, known as the "*decoding attack*," which can re-identify individuals in released datasets.

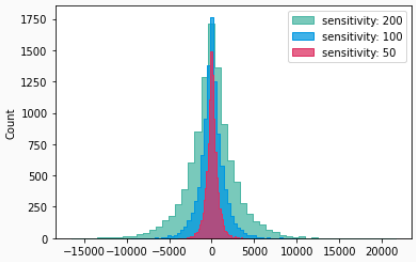
The limitations of the *traditional anonymisation* techniques have resulted in increased support and adoption of differential privacy by government institutions and commercial enterprises for data releases that could otherwise not be shared. Differential privacy is a tool for preserving individuals' privacy by adding statistical uncertainty when sharing sensitive data. The National Institute of Standards and Technology (NIST) recently added differential privacy in their proposed data anonymisation approaches to government data sharing. However, data practitioners have a limited understanding of the concept, use case suitability, privacy guarantee explainability, and other technical challenges in its adoption.

As interest in differential privacy grows, practitioners are turning to tools that simplify its implementation and make it more accessible for various applications. These tools aim to mitigate the complexities involved in implementing differential privacy effectively. However, practitioners face a challenge in choosing the right differential privacy tool for their needs, as each tool offers different functionalities, security, usability, performance guarantees, and so on. A comparative analysis of the prominent tools is crucial to guide practitioners in making informed decisions about their use.

This work aims to equip the readers to get started with differential privacy for responsible statistics release and understand how it can address the privacy gaps left by *traditional anonymisation* techniques. We further aim to guide practitioners to choose the right differential privacy tool through a comparative analysis of four open-source differential privacy libraries and frameworks developed by prominent researchers and institutions, specifically – *PipelineDP* by Google and Openmined, *Tumult Analytics* by Tumult Labs, *OpenDP* by privacy team at Harvard and *Diffprivlib* by IBM. Our analysis can be helpful to ensure that the tools meet the needs to be used effectively in real-world applications.

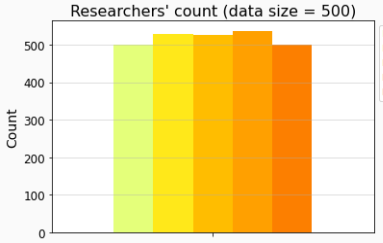
2. INTERACTIVE DEMOS AND OPEN-SOURCED CODE

- We have created Jupyter notebooks that provide a hands-on experience that complements the theoretical discussions in the paper, enabling the reader to gain a deeper understanding of differential privacy and its applications.



Basic mechanisms that satisfy differential privacy definition

This notebook will help you understand the two commonly used algorithms in differential privacy – Laplace and gaussian mechanisms.

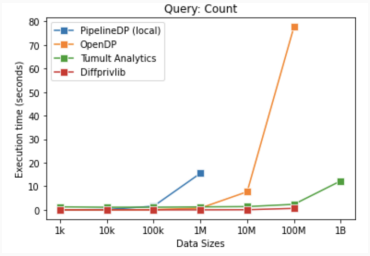


Using differential privacy for privacy-preserving statistical releases on synthetic survey data

This guided notebook aims to study

- Differentially private count, sum, mean, histogram, and contingency table computation
- Impact of certain parameters on the accuracy of the differentially private aggregate statistics.

- We open-source our code for the experiments conducted on benchmarked differential privacy tools to facilitate replication and encourage further exploration of our work to further the research in this direction.



Benchmarking differential privacy libraries and frameworks

Measuring the utility and scalability performance of the four differential privacy tools – Tumult Analytics, OpenDP, Diffprivlib, and PipelineDP.

3. INTRODUCTION

The use of personal information by government agencies, research organisations, and enterprises has been growing, fueled by the advancement of analytical techniques such as machine learning. Big data analysis drives innovation, scientific research, and improving public policies. However, with the growing collection and use of personal information, new risks are evolving, jeopardising individual privacy protection. Real-world privacy attacks have shown that traditional anonymisation techniques (such as *suppression, aggregation,* and *k-anonymity*) are insufficient in protecting against evolving and unpredictable risks.

3.1. LIMITATIONS OF TRADITIONAL ANONYMISATION TECHNIQUES

Ad-hoc Methods

Ad-hoc anonymisation techniques, including *statistical disclosure limitation* (SDL) methods, aim to protect sensitive information by replacing them with privacy-preserving sanitised values. For example, by suppressing, perturbing, swapping, and generalising attributes of individuals in the dataset. These techniques often aim to prevent re-identification attacks (reversing anonymisation and tracing an individual record back to its human source).

These techniques are usually ruled-based (such as applying stringent anonymisation depending on the data-sharing environment), guided by policies, or driven by heuristics. This makes it difficult to assess their privacy offering and limits the potential uses of the data. Furthermore, these techniques require practitioners to design attack models based on the available auxiliary information (external information such as public datasets and background knowledge), computational power, and current threats.

Some privacy models, such as *k-anonymity*, require classifying attributes based on their identifiability – *direct identifiers* (e.g., NRIC, names) and *quasi-identifiers* (e.g., age, gender) – and sensitivity – sensitive attributes (e.g., medical condition, salary). The attributes that seem not identifying/sensitive now might cause potential risk later or simply because they were not deemed to cause a privacy risk.

Even ex-post remedies, such as revoking access to the data after a vulnerability has been discovered, are ineffective in addressing the issue. This is because multiple copies of the data may exist and persist online indefinitely, making it impossible to erase the data completely.

Aggregates Release

Simply innocuous-looking *aggregates*, such as contingency tables, mean, and count, can be vulnerable to *differencing* and *database reconstruction attacks*. The fundamental law of information reconstruction states that

> *"Overly accurate" estimates of "too many" statistics are blatantly non-private and can compromise the entire confidential dataset.*

With the differencing attack, an attacker can isolate an individual value by combining multiple aggregate statistics about a dataset. Consider the below ~~toy~~ simple example:

Difference attack

- Let's say the mean income of 49 people in a workshop is $6000
- Then, say a new person joins in, and the mean income of 50 people becomes $6200
- Then the new person's income is abs(6200*50 - 6000*49) = $16,000

Database reconstruction attacks (DRA) leverage large volumes of published statistical aggregates to narrow down the possible values of individual-level records. This is demonstrated by the US Census Bureau researchers, who conducted an experiment on summary statistics from the 2010 census and could putatively re-identify 52 million people or 17% of the total population[1].

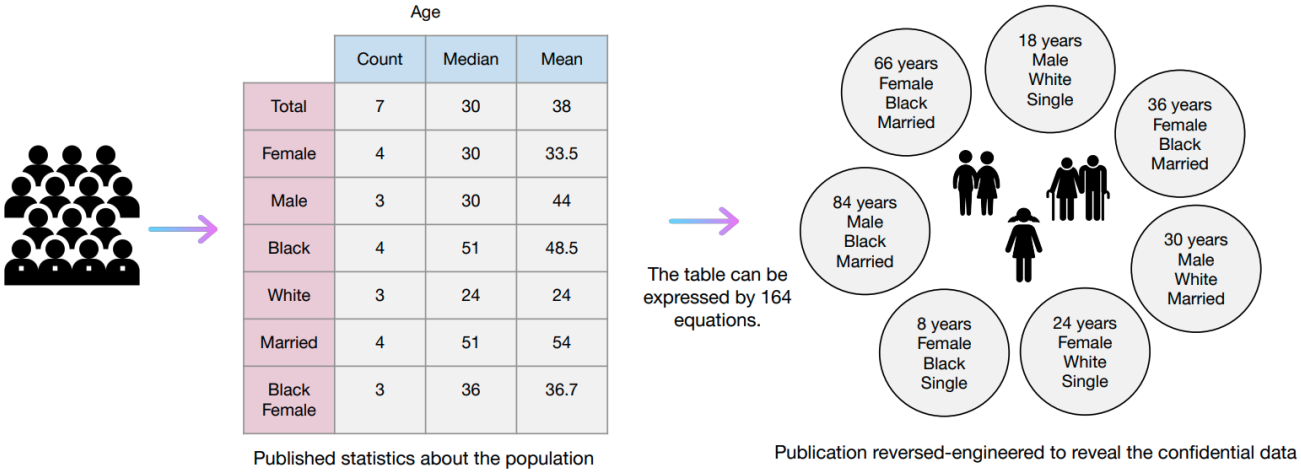


Figure 1: The Database Reconstruction attack (DRA) illustration based on Garfinkel's work [1].

Because of these attacks, data controllers are left with two choices to publish statistics: *fewer* or *less accurate.* With the "*fewer statistics*" choice, it is hard to know how many statistics are "too many" that can destroy privacy. Releasing "*less accurate statistics*" has been considered a plausible solution that can reduce the risk of re-identification. However, how much noise is enough to get a reasonable privacy-utility trade-off? Too much noise can degrade utility, and too less noise can make a dataset vulnerable to DRA and *differencing* attacks. It turns out differential privacy is exactly what can be helpful here. It provides a mathematical approach for adding enough noise to get desired privacy and prevent attacks.

3.2. ADDRESSING DATA PRIVACY CONCERNS WITH DIFFERENTIAL PRIVACY OVER TRADITIONAL ANONYMISATION TECHNIQUES

Putting together the key concerns with traditional anonymisation techniques are that they:

- **Provide privacy only in a heuristical and empirical sense:** even seemingly safe data releases can be vulnerable to re-identification.
- **Require careful analysis of auxiliary information:** there is no control over an adversary's knowledge gain, evolving data collection methods, or public releases of data in the future.

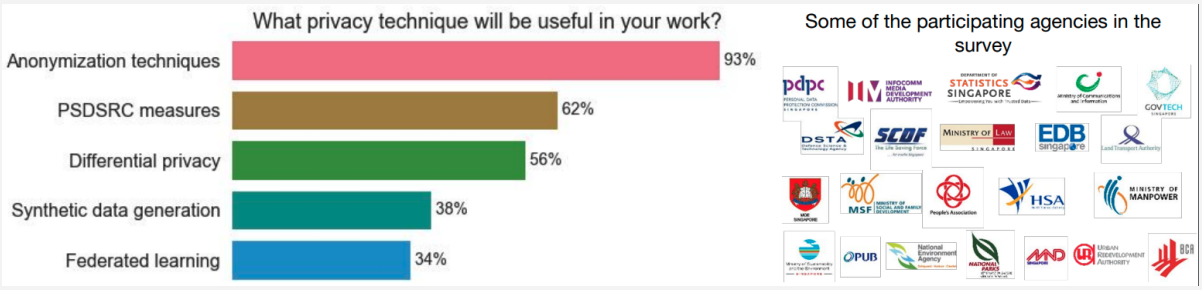
[1] From remarks of John Maron Abowd, Chief Scientist, US Census Bureau, at the annual AAAS meeting, February 16, 2019, Washington, DC.

- **Require careful analysis of an adversary's computational power and current threats:** there is only evidence of more powerful computational resources (such as quantum computing), and attacks can become more advanced and difficult to predict over time.
- **Cannot assess the overall accumulated risk:** each statistical release about an individual in a dataset leaks some information and can make them vulnerable to re-identification. By quantifying accumulated privacy risk, statistical release can be prudently controlled.

Differential privacy [2] is a new notion of privacy that enables the release of data or results from data analysis with a mathematical guarantee of privacy. It focuses on data-releasing algorithms (such as computing the mean, sum, count, etc. of a dataset) that take in analytical queries and produce outputs that have been altered in a controlled, random manner. These algorithms are considered differentially private if, based on the outputs, it is difficult to determine whether any individual's information was included in the original dataset. This is accomplished by guaranteeing that a differentially private algorithm's behaviour hardly changes whether or not a single individual is a part of the dataset.

The rising transition to differential privacy is primarily because it can address these concerns.

- It provides a **quantifiable, provable guarantee** about the worst-case privacy risk.
- Its privacy guarantee **requires no assumption about auxiliary information:** an attacker with detailed information or access to future data releases can still not get an advantage.
- Its privacy guarantee **requires no assumption about the threat from an adversary's computational power** (even if unlimited) and can prevent any arbitrary threat.
- It allows **quantifying the accumulated privacy risk** of releasing multiple statistics about individuals.



A survey of 120 participants from 57 government agencies conducted by GovTech in 2022 found that traditional anonymisation techniques were considered the most useful. Interestingly, the agencies also expressed interest in differential privacy. While the traditional anonymisation measures remain an essential privacy weapon for data sharing, they can have privacy risks that differential privacy can address for statistical datasets.

4. PRIMER ON DIFFERENTIAL PRIVACY

4.1. PRIVACY GUARANTEE OF DIFFERENTIAL PRIVACY

We will understand the privacy guarantee of differential privacy with the help of an example[2].

Assume a scenario where the Ministry of Health is surveying to gather information about smokers and their smoking habits for research purposes that could benefit society. However, Emma, who values her privacy, is concerned that the data collected from the survey may one day be leaked, causing her harm through reputational damage, monetary loss, or an increase in her insurance premium.

Differential privacy offers a solution to Emma's concerns in this scenario. Intuitively,

"Any analysis protects the privacy of individuals in the data if its output

does not reveal any information about any specific individual."

If Emma's information is removed from the analysis, her privacy is protected as the output of the analysis does not depend on her specific information. Differential privacy aims to protect Emma's privacy in the scenario, with her record included in a way that mimics the privacy protection she is afforded in her opt-out scenario. This means, with differential privacy, the outcome of the data analysis will essentially be the same with or without her data and will not disclose anything specific to her.

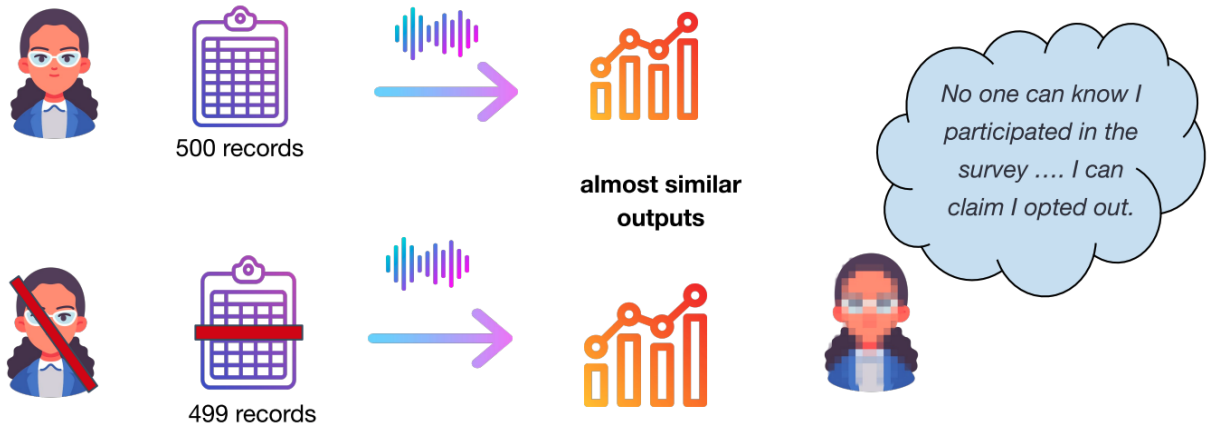


Figure 2: Differential privacy adds a calculated amount of noise to hide each individual's contribution to a dataset.

To get this similarity, differential privacy adds a calculated amount of randomness or noise to the statistic. The magnitude of noise to add, which determines the degree of privacy, depends on the type of analysis. It must be sufficient to hide the largest contribution that can be made to the output by one individual.

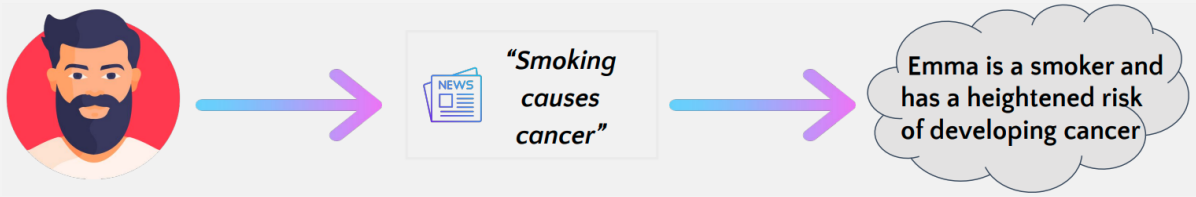
The plausible deniability offered by differential privacy is synonymous with the privacy guarantee. With this guarantee, Emma might as well participate in the survey and answer truthfully. Crucially, this differential privacy guarantee is made not only with respect to Emma but also with respect to every other individual contributing his or her information to the analysis. This can increase survey participation, leading to more accurate statistical releases about a population.

[2] Our work provides a condensed and more visually-rich summary of the example provided in Wood, Alexandra, et al.'s seminal work on differential privacy [3].

"Differential Privacy allows us to learn about the population …

Assume Peter, who knows Emma, reads the outcome of the survey analyses in the news – *"Smoking causes cancer."*

He concludes - *"Emma is a smoker and has a heightened risk of developing cancer."*



This is a privacy violation of Emma, right? 😳

No! Differential privacy does not guarantee that information about Emma remains completely private. The outcome about the "population" would have been released regardless of Emma's participation. Peter could have drawn the same conclusion from the outcome about Emma, but he cannot determine whether Emma took part in the survey. This safeguards Emma's privacy by preventing the disclosure of information specific to her. Moreover, the survey outcome has the potential to benefit the wider population, including Emma, by promoting healthy behaviours.

… but not about an individual."

Differential privacy is unsuitable for cases where the goal is to identify or learn specific individuals or a small group of people (such as outliers). This is antithetical to the differential privacy definition. The amount of noise needed to hide their contribution will be significantly high, rendering the result of the analysis useless.

4.2. FORMAL MATHEMATICAL INTERPRETATION OF DIFFERENTIAL PRIVACY

We understood that the outcome of a differential privacy analysis would be similar whether or not an individual's data is included. By 'similar,' we mean that the probabilities are close. Let's now translate this into a formal definition. Suppose exactly one person's data is added or removed from the dataset. These are called *neighbouring datasets*, denoted by **D** and **D'**, that differ in exactly one person's record. Then all possible outputs **O** of mechanism[3] **M** will be probabilistically similar (similar approximate outputs) for all **D** and **D'**.

The degree of similarity depends on the privacy loss parameter **ε** (epsilon): the smaller it is, the more similar the outputs are from the neighbouring datasets, and mathematically given as

$$Pr[M(D) \in O] \leq e^{\epsilon} Pr[M(D' \in O)]$$

[3] A "mechanism" is differential privacy lingo for an algorithm. The mechanism adds random noise from a probability distribution to aggregate statistics.

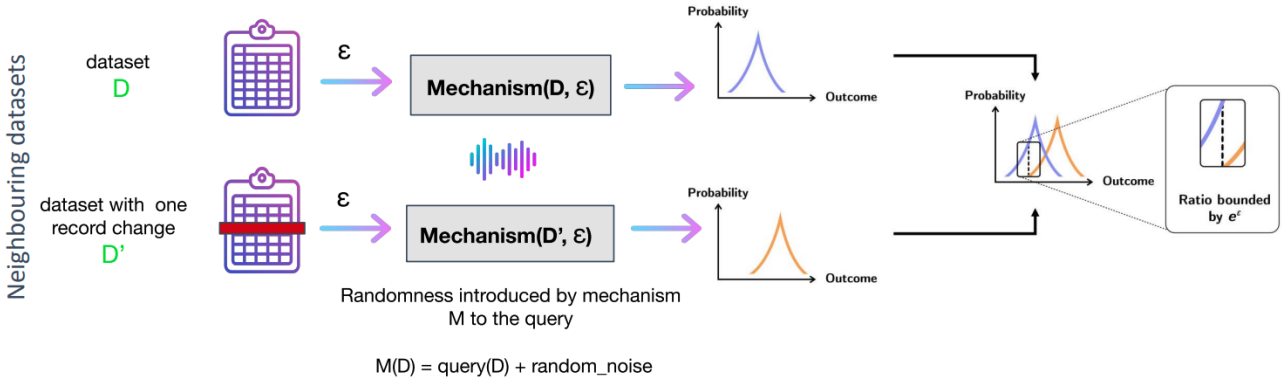


Figure 3: The output of the mechanism M will probabilistically be the same for the neighbouring datasets D and D'. The similarity depends on the privacy parameter **ε.**

Notice that the exponential function e^{ϵ} is applied to **ε**. If ε is very close to zero, then e^{ϵ} is close to one, resulting in similar probabilities. As **ε** increases, the probabilities can become more different.

This is the definition of differential privacy that, if a mechanism **M** satisfies, is called **ε (pure)-differentially private.**

ε – the privacy parameter

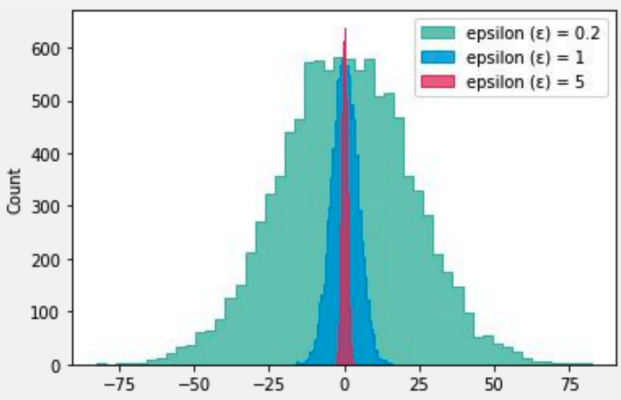


Figure 4: Noise scale decreases with the increasing epsilon value.

Epsilon is a privacy parameter like a tuning knob to handle the privacy utility trade-off. The relationship between epsilon and accuracy[4] is inversely proportional, meaning that as the value of epsilon decreases, the level of privacy increases, but the accuracy of the results decreases. Conversely, as the value of epsilon increases, the level of privacy decreases, but the accuracy of the results improves.

While the choice of epsilon for differential privacy ultimately depends on the specific context and use case, it is generally recommended to set epsilon less than 1 to achieve a conservative level of privacy while maintaining reasonable utility [3]. However, practitioners may sometimes use higher epsilon values to achieve greater analysis accuracy [4]. It is important to remember that a larger epsilon value may compromise privacy guarantees, which underscores the need for careful consideration of privacy and accuracy trade-offs in selecting the epsilon value.

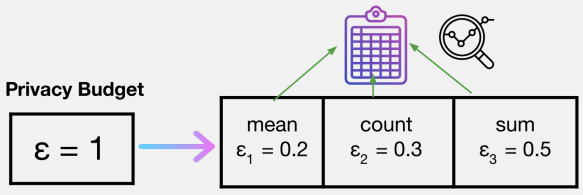


Figure 5: Uneven splitting of the privacy budget

[4] Accuracy is used as the proxy for utility throughout this work.

ε – privacy budget.

Privacy budgeting limits the types and number of queries that can be made on data to prevent privacy breaches. A good analogy for privacy budgeting is shopping with a fixed budget, where you have to prioritise which items are most important for your wardrobe. Similarly, for privacy budgeting, you might prioritise the accuracy of certain statistics and allocate a larger portion of the privacy budget to those. The privacy budget is determined based on the level of risk the data holder is willing to tolerate. Overspending the budget can increase the privacy risk and lead to attacks such as reconstruction attacks.

ε set to zero.

Setting epsilon to zero ensures perfect privacy but zero utility. The mechanism's output will be indistinguishable from changing a user's record. Therefore, adding or removing users will not affect the mechanism's output. This also means that the output does not reveal any interesting information about the data and is entirely independent of the input.

We will now try to understand differential privacy with the help of a concrete example.

Counting the number of survey participants with differential privacy

Suppose we want to count the number of survey participants in a differentially private manner. To do this, we need to add random noise to the count. This noise should come from a probability distribution with the necessary property[5] to satisfy the definition of differential privacy. One such distribution is the standard Laplace distribution, denoted as *laplace*(*noise_scale*).

The mechanism that adds random noise to a query output (count in this case) from the Laplace distribution is called the Laplace mechanism. For a query output on dataset D, the *Laplace mechanism* is given as

laplacian_mechanism(D, noise_scale) = query(D) + *laplace*(*noise_scale*)

Here, **noise_scale = sensitivity/ε,** where ***sensitivity*** is the maximum possible change in a query's output when an individual's record is added/removed. This ensures that a sufficient amount of noise is added to hide each individual's contribution in a dataset. Notice the noise scale is directly proportional to the sensitivity and inversely proportional to epsilon.

For the count query, the change can be a maximum of 1 on adding or removing an individual from the survey data. So the sensitivity of the count query is always 1. Let **ε** = 0.01, then noise_scale = 1/0.01 for the Laplace distribution. The differentially private count can be given as

private_count = laplacian_mechanism(D, noise_scale)

[5] There is some R (called the width of the distribution) such that for any two consecutive numbers, the ratio of their probabilities is R.

Suppose the true count is 500. On executing the laplacian_mechanism multiple times, private_count can take on different random values (501.23, 499.81….) nearly close[6] to the true count. Adding or removing an individual's data can still result in nearly accurate outputs for meaningful analysis and decision-making.

It is possible for someone to determine which probability distribution the algorithm uses to estimate the distortion of the true output. However, the specific random number added by the mechanism remains unknown. *This randomness and the similarity of outputs increase the uncertainty of inferring an individual's contribution to a dataset.*

4.3. GLOBAL VS. LOCAL DIFFERENTIAL PRIVACY

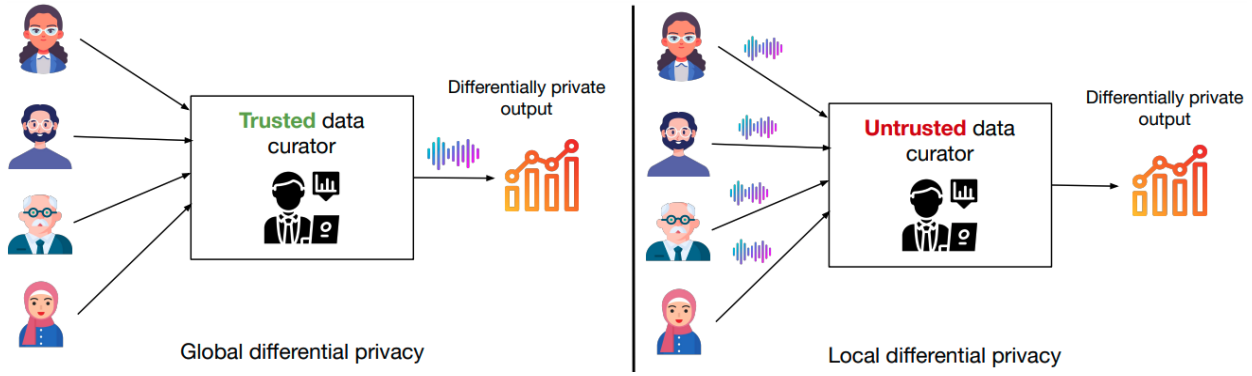


Figure 6: Global vs Local differential privacy differs with respect to the trust in the data curator.

Differential privacy can be implemented in two ways: *global* or *local*. *Global* differential privacy[7] adds noise to aggregates, while *local* differential privacy adds noise to individual data points before aggregation. The former assumes a trusted data curator holds sensitive information. In contrast, the latter assumes an untrusted data curator and therefore maintains privacy at the source before data leaves the data subject's control.

Local differential privacy can result in higher noise levels than the *global* model for the same level of privacy protection and often requires many data points. While *global* differential privacy is more widely adopted due to its more accurate results, *local* differential privacy can be applicable mostly for analysis over massive data sets (in untrusted data curator setting), such as the user bases of large technology companies like Google, Microsoft, and Apple [4], who have deployed *local* differential privacy for various use cases.[8]

[6] In general, accuracy of a differential private query output depends on various factors including sensitivity of the query, epsilon value and dataset size.
[7] This work focuses on global differential privacy and applying it to aggregate statistics.
[8] Alternative distributed models for differential privacy are being researched, such as computational multi-party differential privacy, the hybrid model, and the shuffle model. These models lie in between *global* and *local* models and aim to provide stronger privacy protections with improved accuracy.

4.4. APPLICATIONS

A spectrum of statistical analyses, including count queries [5], histograms [5], cumulative distribution functions [6], and linear regression [7], can be conducted using differentially private algorithms. Other techniques used in deep learning [8] and machine learning, such as clustering and classification [9], as well as synthetic data generation, can also be performed using differential privacy methods. Some well-known applications of differential privacy [4] can be seen in companies like Apple, which uses it to collect aggregate statistics on Emoji usage, and Google, which leverages it to gather information from Google Chrome crash reports. The US Census Bureau has also adopted differential privacy to protect sensitive information in the summary statistics for the 2020 Decennial Census [10].

Figure 7: Examples of real-world uses of differential privacy. Refer [4] for the informative article on the practitioners' choice of epsilon values (ranging from low to high).

4.5. KEY PROPERTIES

Composition

This is the most powerful property of differential privacy for quantifying cumulative privacy loss. Each query leaks information about the data when running multiple queries on an individual-level dataset. With differential privacy, the privacy loss can be accumulated by adding epsilons used for each query.

Future-proof

Unlike other privacy models, you do not have to worry about the attacker's knowledge or computational power, secondary dataset releases, or future data releases. The privacy guarantee offered by differential privacy will still hold.

Post-processing

One can perform further computations on differential private statistics releases, such as replacing negative results with zeros or improving noisy output. The privacy guarantee will still hold if the original dataset is not touched.

Group privacy

The differential privacy guarantee is not limited to a single individual's contribution to a dataset. It allows the guarantee to decay gracefully with multiple contributions from individuals or groups of size k (e.g., protecting a group of family members).

5. EVALUATION OF DIFFERENTIAL PRIVACY TOOLS

We have seen an increase in the development of differential privacy tools to improve accessibility and reduce the need for expert input. These tools provide a higher-level interface, abstracting implementation complexities and ensuring meaningful differential privacy guarantees, automated utility optimisations, etc. However, practitioners face difficulty choosing the right tool for their needs, as each tool offers different functionalities, security, performance, and usability guarantees. To address this, we evaluated existing tools[9] – libraries (provide specific functionalities) and frameworks (provide abstractions to build specific applications) – through qualitative analysis of their features and characteristics and quantitative analysis of their performance and accuracy across various queries.

5.1. RELATED WORK

Our research builds upon the benchmarking efforts of differential privacy libraries and frameworks conducted by Gonzalo et al. in [11] and [12]. In [11], the authors conducted a qualitative and quantitative analysis of five differentially private analytics tools, including Google DP,[10] SmartNoise[11], Diffprivlib[12], diffpriv[13], and Chorus[14]. However, some of these libraries – SmartNoise, diffpriv, and Chorus, are either deprecated or not actively maintained. With new tools and features emerging, we have benchmarked the latest evolving libraries and frameworks developed by prominent researchers and institutions, which have the potential for wider adoption. Moreover, we have extended the quantitative analysis to datasets with up to 1 billion data points, which is a significant improvement over the previous work's limit of 10 million data points. Additionally, we have tested the scalability supporting libraries in the distributed environment, which is crucial for practical deployment in big data systems. While [12] provided a qualitative analysis of the various emerging libraries, frameworks, and systems, our work offers more comprehensive and structured key desiderata for qualitative analysis with a focus on features that we believe can be helpful for both experts and non-experts in the field.

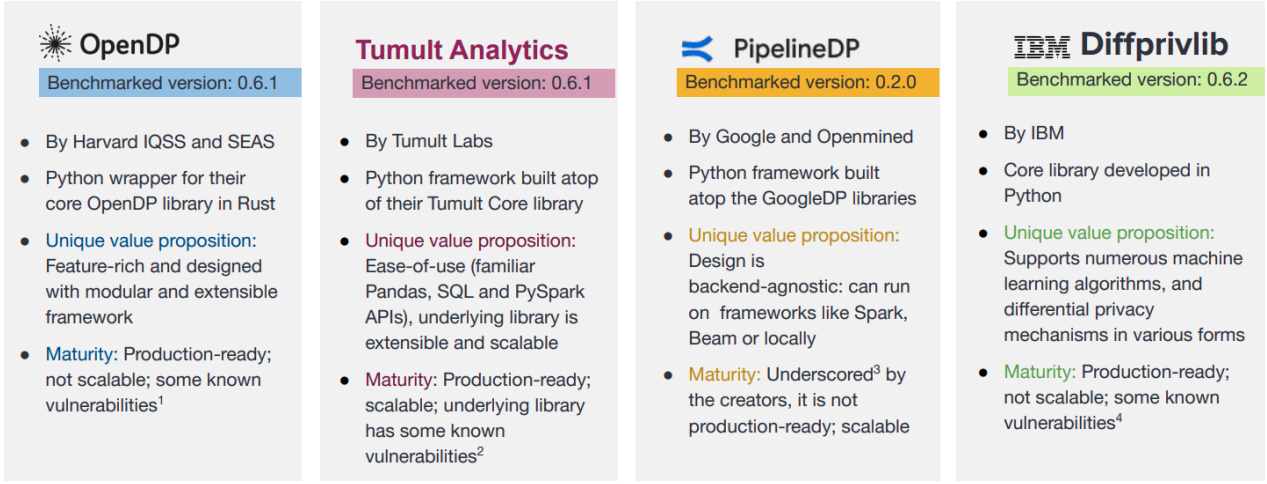
5.2. BENCHMARKED DIFFERENTIAL PRIVACY LIBRARIES AND FRAMEWORKS

We consider four major open-source libraries and frameworks[15] that provide support in the following aspects

- Written in Python, the most popular language among practitioners.
- An active community for maintainability and support.
- Suitable for non-experts.
- Open-source and developed by prominent researchers and institutions
- Scalable to support real-world use cases with large datasets.

We consider the following tools: **OpenDP**[16] (library) by Harvard privacy team[17]**, Tumult Analytics**[18] (framework) by Tumult Labs[19]**, PipelineDP**[20] (framework**)** by Google and Openmined[21], and **Diffprivlib** (library) by IBM.

[9] We refer libraries and frameworks as tools.
[10] https://github.com/google/differential-privacy
[11] https://github.com/opendp/smartnoise-core
[12] https://github.com/IBM/differential-privacy-library
[13] https://github.com/brubinstein/diffpriv
[14] https://github.com/querqy/chorus
[15] Frameworks are considered over underlying differential privacy libraries as they are intended for non-experts.
[16] https://github.com/opendp/opendp
[17] https://opendp.org/our-people
[18] https://gitlab.com/tumult-labs/analytics
[19] https://www.tmlt.io/
[20] https://github.com/OpenMined/PipelineDP
[21] https://www.openmined.org/



Benchmarked differential privacy tools — libraries and frameworks. References for 1, 2, 3, and 4.

OpenDP library [13], developed by Harvard's Institute for Quantitative Social Science (IQSS) and the School of Engineering and Applied Sciences (SEAS) team, is part of the larger OpenDP Project[22], a community effort to build trustworthy, open-source software tools for the analysis of private data. It is feature-rich and built atop a modular and extensible framework [14] for expressing privacy-aware computations using several different privacy models. It is implemented in Rust with Python binding support currently.

Tumult Analytics [15], developed by Tumult Labs - a startup specialising in differential privacy solutions and founded by pioneers in the field - is a user-friendly framework (familiar Pandas, SQL, and PySpark APIs) that provides scalable capabilities (100M+ rows and output statistics). The underlying framework (similar to OpenDP's), Tumult Core, is a collection of components and composition operators that allow for implementing complex differential privacy mechanisms. Tumult Analytics is feature-rich and utilised by organisations such as the U.S. Census Bureau and the Wikimedia Foundation[23].

PipelineDP [16], developed by Google and Openmined, is an end-to-end framework for generating differentially private aggregations on large datasets using batch processing systems. The differentiating factor about PipelineDP is that it allows backend-agnostic data processing: Apache Spark, Apache Beam, and locally. The framework is built atop Google's differential privacy building block library in C++. It is designed to handle SQL type of queries with APIs to handle user contributions. Practitioners should note that the framework is experimental and subject to changes; it is not recommended in production systems. Nevertheless, as the framework evolves, it can serve various large-scale applications.

Diffprivlib [17], developed by IBM, is a general-purpose library offering extensive collections of differential privacy mechanisms beyond standard ones. Diffprivlib offers some of these mechanisms in various forms, namely their truncated and the bounded form. It relies on NumPy for all the underlying computations and has scalability limitations and security vulnerabilities. It also supports features to apply differential privacy on machine learning models[24] such as classification and clustering, which run like Sklearn classifiers with the option to track the privacy budget.

[22] https://opendp.org/
[23] https://www.usenix.org/conference/pepr22/presentation/triedman
[24] The application of differential privacy in machine learning is beyond the scope of this work.

5.3. QUALITATIVE ANALYSIS

The following key desiderata, built upon the work of Gonzalo et al. [12], are considered for the qualitative analysis of the libraries and frameworks that can be helpful to basic and advanced users (refer to Table 1).

- **Analytics** - Implementing differentially private statistics can be challenging for various reasons, including calculating the sensitivity of a query, mechanism to use, optimisations, and so on. The tool abstracts these computations and provides high-level APIs for the same. The tool supports
    - **Differentially private aggregate statistics**
        - For single-valued (e.g., count and mean) and multi-valued statistics (e.g., histogram, contingency table).
        - For GROUP BY queries, the tool supports (i) public (known categories that require explicit user input) or private categories (unknown or sensitive categories that are computed with privacy budget spending), (ii) individual contribution bounding cross- and per-group, (iii) empty categories handling that can leak privacy, and (iv) group-by multiple columns.
    - **Adaptive/interactive querying** - Dynamic querying where the output of a query can depend on the previous query's result, including determining how much privacy budget is consumed.
- **Security** - The tool
    - **Blocks data visibility** - Blocks the user from "seeing" the data execute arbitrary code or visualise the dataset.
    - **Generates cryptographically secure random number generation** - A randomised algorithm is as good as the randomness it brings in the random number generation.
    - **Protects against floating-point vulnerabilities** - Computers have limited time and memory, so they round continuous numbers and arithmetic results, creating vulnerabilities in differential privacy. The limited representation of floating points can leak privacy and break its guarantees. This is known as a floating-point vulnerability [18] in differential privacy.
- **Usability** - The tool supports
    - **Scalability**: The user can input arbitrary-sized datasets. Many real-life datasets are too large to be loaded into memory and require distributed computing using engines such as Apache Spark.
    - **Accuracy adjustment and error estimation:** The user can set their desired accuracy level or get information about the noise scales and a confidence interval.
    - **Parameter search:** The user can control the accuracy of a query by defining the notion of neighbouring datasets and epsilon, estimating the smallest epsilon for the desired accuracy, and determining the necessary dataset size or maximum user contribution.
    - **Loading multiple public/private data sources:** The user can input multiple public and private datasets for advanced computations such as performing a JOIN.
    - **Pre-processing functionality** - Data transformation might be needed before differential private statistical computation. These operations might include imputing or dropping null values[25], grouping into categories, type casting (e.g., string to float), etc.
    - **Post-processing functionality:** The user can apply operations on the computed differentially private statistics usually to minimise noisy signals.
- **Differential privacy features** - The tool supports:
    - **Privacy budget accounting**
        - Set and track the privacy budget spending

[25]Handling null values in a differential privacy setting can be challenging. By allowing aggregations to propagate null, they provide a non-differentially-private bit revealing the existence of nullity in the dataset, and by implicitly dropping nulls from aggregations (of known dataset size) the sensitivity of non-null individuals is underestimated. Therefore, aggregators must be fed completely non-null data by imputing or dropping them.

        - Blocks queries on exhausting the budget
        - Automatic budget splitting across queries
    - **Differential privacy mechanisms:** such as Laplace, Gaussian, Exponential mechanisms. (Refer to Appendix Table A)
    - **Differential privacy definitions:** such as pure, approximate, zero-concentrated differential privacy [19]. (Refer to Appendix Table A)
    - **Composition**: Computing all statistics together in a batch; this leads to more efficient usage of budget.
    - **Population amplification** [20]: Optimization of epsilon utilisation based on the sampling rate of the population
    - **Privacy definition casting**: Casting a privacy definition to another for privacy interpretation. For example, some privacy definitions cannot be interpreted in terms of epsilon and require definition casting.
    - **Local differential privacy**: Algorithm to apply differential privacy at the individual level (local differential privacy) rather than the aggregates (global differential privacy).

Table 1: Qualitative comparison of the tools on the key desiderata. ✔ denotes functionality exists; ✘ denotes functionality does not exist; † denotes that the functionality is publicly stated by the creators to be added in the future. Please note some of the functionalities might be supported by the underlying core libraries but missing in the framework in the experimented versions. (refer to Appendix Table A for an understanding of the concept)

	OpenDP	Tumult Analytics	PipelineDP	Diffprivlib
Analytics				
Differentially private aggregate statistics	Count (rows, distinct values), Sum (various cases), Mean, Variance, Histogram (supports public & private categories)	Count (rows, distinct values over multiple columns), Sum, Mean, Variance, Standard Deviation, Quantile, Min, Max, Median	Count (rows, privacy IDs[26]), Sum, Mean, Variance, Percentile	Count, Sum, Mean, Variance, Standard Deviation, Quantile, Percentile, Median, Histogram (2-dimensional and multidimensional variants)
GROUP BY	Only GROUP BY followed by count supported	✔ Supports public and private † categories, Individual contribution bounding, drops null values, group-by multiple columns.	✔ Supports public and private categories, individual contribution bounding, drops null values, group-by multiple columns	✘
Adaptive querying	✘ †	✔	✘	✘
Security				
Cryptographically secure random number generation	✔	✔	✔	✔
Protection against floating-point vulnerabilities	✔	✔	✘	✘
Block data visibility	✘	✔	Spark backend supports private RDDs[27].	✘
Usability				
Scalability	✘	✔ (Spark)	✔ (Spark and Beam)	✘
Accuracy adjustment and	✔	✘	✘	✘

[26]A Privacy ID is an identifier for a privacy unit that we protect with differential privacy. This unit can refer to a single individual or multiple values, e.g., setting privacy ID as individual + restaurant pair means protecting all the visits by an individual to a specific restaurant. Refer https://pipelinedp.io/key-definitions/

[27] https://www.databricks.com/glossary/what-is-rdd

error estimation				
Parameter search	✔	✘ †	✘	✘
Loading multiple public/private data sources	✘	✔ (with truncation strategies to drop excess and unique rows)	✘	✘
Pre-processing functionalities	✔	✔	✔	Only casting from int to string and vice versa, and integer rounding
Post-processing functionality	✔	✘	✘	Only casting from int to string and vice versa, and integer rounding
Differential privacy features				
Privacy budget accounting	✘	✔ Allows infinite budget setting and query blocking	✔ Utilizes the entire budget for the queries and private categories computation; allows query blocking	✔ Allows infinite budget setting and query blocking
Differential privacy mechanisms	Laplace, Discrete Laplace, Gaussian, Discrete Gaussian, Stability histogram	Laplace, Discrete Laplace (Geometric), Gaussian, Discrete Gaussian, Exponential	Laplace, Gaussian	Laplace, Gaussian, Binary, Bingham, Exponential, Geometric, Snapping, Staircase, Uniform, Vector
Differential privacy definitions	Pure, Approximate, Zero-concentrated	Pure, Zero-concentrated, Approximate †	Pure, Approximate	Pure, Approximate
Composition	✔	✘	✔	✘
Population amplification	✔	✘	✘	✘
Privacy definition casting	✔	✘	✘	✘
Local differential privacy	✔ (Randomised response boolean or categorical [23])	✘	✘	✘

5.4. QUANTITATIVE ANALYSIS

In this section, we will evaluate the utility and scalability of the benchmarked tools on synthetic datasets. Our experimental setup is largely based on the work by Gonzalo et al. [11] and built upon its findings. For completeness, we will provide a brief overview of the experimental settings.

5.4.1. EXPERIMENT SETTINGS

Evaluation principles

Diversity of dataset characteristics: We used synthetic datasets generated using the skew-normal distribution[28] to provide *scale, skew,* and *size* (refer to Figure 8) diversity.

- *Scale:* The datasets of varying spread with values 50, 250, and 500.
- *Skew:* The datasets of varying shapes with values 0, 5, and 50.
- *Size:* The datasets of varying sizes ranging from 1000 to 1 billion data points. This wide range of sizes helps to test the tools' scalability on large datasets in the standalone and distributed computing environment.

The diversity in the dataset characteristics can help simulate a reasonable model of real-world data distributions. We can comprehensively understand their behaviour and performance with diverse dataset characteristics by exposing the tools to datasets with varying scales, skewness, and sizes. This analysis is crucial for real-world applications where data is often complex and diverse in nature.

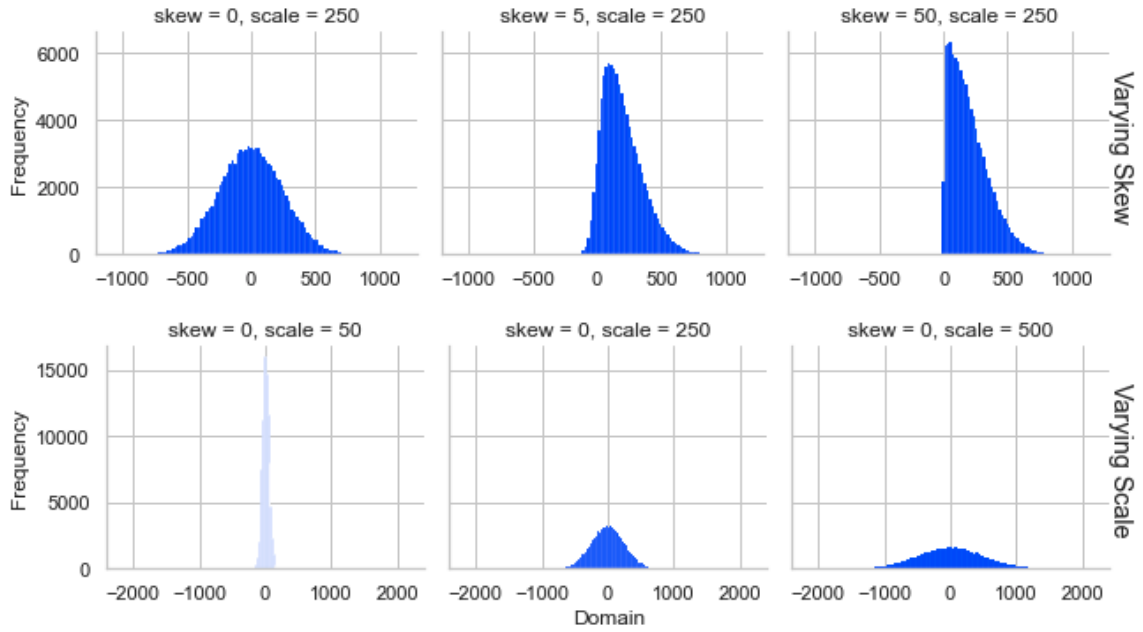


Figure 8: Sample of the synthetic datasets' distribution with varying *skew* and *scale* of size 100k data points.

Diversity of epsilon values: We ran the experiments on 28 epsilon values ranging from 0.01 to 10 with logarithm steps. The choice of the epsilon values is informed by a combination of academic recommendations and practical considerations: experts in academics suggest positive epsilon values of less than 1, and practitioners often use values beyond this range in real-world applications. Additionally, our pilot experiments indicated that the performance of the

[28]https://en.wikipedia.org/wiki/Skew_normal_distribution

tools marginally differed for epsilon values greater than 10. As a result, we limited the epsilon values to 10 to provide a more focused evaluation of the tools' performance.

Diversity of analytics queries: We evaluate the commonly-used queries supported by all the tools – *count, sum, mean,* and *variance.* Except for the count query, other queries depend on the content of the dataset and require setting the *clamping bounds*.[29] To ensure our evaluation was consistent across all the tools, we set the maximum and minimum values in the datasets as the *clamping bounds.*

Metrics: We used the mean of relative error (MRE) for the utility analysis to measure the accuracy of the queries' output. MRE is widely used in the extant literature [21] and allows the comparison of results from datasets of different scales. Lower MRE values indicate better performance. The error calculation baseline is the deterministic output of the queries. For the scalability performance, we measure the execution time (in seconds) of the queries.

Implementation details

The experiments were conducted on the AWS Glue[30] (python environment) using a single DPU and 16 GB of memory. For the Spark-supported tools – Tumult Analytics and PipelineDP, experiments were conducted on AWS EMR [31] utilising two r4.8xlarge instance types. As PipelineDP is backend-agnostic and supports standalone and distributed computing, we ran experiments on it in both environments.

5.4.2. EXPERIMENT RESULTS

Utility analysis

Setup: We conducted utility analysis on synthetic datasets with varying *sizes* (1k, 10k, and 100k data points), *skew*, and *scale*, resulting in 27 datasets, each evaluated on the four queries – count, sum, mean, variance. Each query was further run with 28 different epsilon values.

Some concepts to understand the experimental results

Neighboring definition: There are two definitions of neighbouring datasets (referring to Figure 3, D and D' that differ by one individual): *unbounded* and *bounded*. The *unbounded* neighbouring definition is used when a record is added or removed to hide the presence or absence of an individual in a dataset. For example, if a dataset contains information about patients in a hospital, the *unbounded* neighbouring definition might be used to ensure that the presence or absence of a particular patient is not revealed in the released data. The bounded neighbouring definition is used when a record is changed to protect the privacy of an individual attribute value when its presence is already known in the dataset. For example, if a dataset contains information about employees in a company, the bounded neighbouring definition might be used to ensure that the salary of a specific employee is not revealed in the

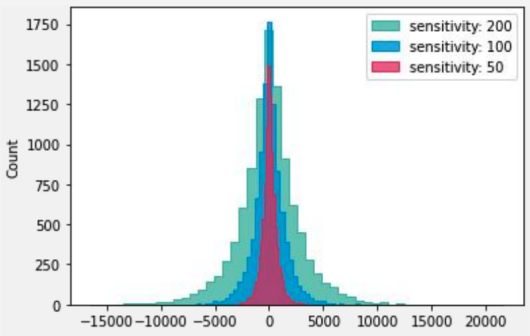


Figure 9: Noise scale increases with the sensitivity

[29] Clamping bounds limit data values within a specified range to minimize the influence of any individual, especially the outliers, on queries like sum, mean, etc. This is necessary to protect individual privacy and ensure the utility of the results.
[30] https://aws.amazon.com/glue/
[31] https://aws.amazon.com/emr/

released data while still allowing the employee to be identified.
Both definitions are important for different use cases of differential privacy and help ensure that the release of sensitive data is protected against privacy breaches.

Sensitivity: It is the maximum change in a query's output on changing one record in a dataset, and the noise scale of a distribution increases with sensitivity (refer to Figure 9). Sensitivity[32] computation depends on the mechanism used, query to compute, and neighbouring definition. Given that this work uses the Laplace mechanism[33], the L1 sensitivities of the query functions are given in Appendix Table B. We also refer[34] the readers to Harvard's papers on calculating the sensitivity of a query based on the neighbouring definition.

Most tools have default mechanisms[35] and neighbouring dataset definitions (bounded or unbounded) for query computation (see Table 2). One consideration for the default mechanism [36] is based on the data type; integer-valued data points default to the Geometric mechanism (also called discrete Laplace mechanism) that outputs integer random values, and the float type defaults to the Laplace mechanism. In cases where there is no default mechanism, we resort to the Laplace mechanism (pure-differential privacy).

Moreover, our implementation (refer to our GitHub repository[37]) is tailored for a non-expert who might depend on automated parameter computation (such as sensitivity) and minimal inputs – epsilon value, clamping bounds (set as maximum and minimum values of the dataset), and max user influence bounding (set as 1).

	OpenDP	Tumult Analytics	PipelineDP	Diffprivlib
Count	**ND:** Unbounded **DM**: Geometric	**ND:** Unbounded **DM**: Geometric	**ND:** Unbounded **DM:** Laplace	**ND:** Unbounded **DM:** Geometric Truncated
Sum	**ND:** Bounded **DM:** Laplace	**ND:** Unbounded **DM (int):** Geometric **DM (float):** Laplace	**ND:** Unbounded **DM:** Laplace	**ND:** Bounded **DM (int):** Geometric Truncated **DM (float):** Laplace Truncated
Mean	**ND:** Bounded **DM:** Laplace	**ND:** Unbounded **DM (int):** Geometric **DM (float):** Laplace	**ND:** Unbounded **DM:** Laplace	**ND:** Bounded **DM:** Laplace Truncated
Variance	**ND:** Bounded **DM:** Laplace	**ND:** Unbounded **DM (int):** Geometric **DM (float):** Laplace	**ND:** Unbounded **DM:** Laplace	**ND:** Bounded **DM:** Laplace Bounded Domain

Table 2: Default mechanisms (DM) and neighbouring definitions (ND) of the tools for each query in our benchmark. Tumult Analytics uses privacy definitions as input, which can be pure (PureDP) or zero-concentrated (RhoZCDP) differential privacy. For PureDP, the default mechanism is listed, while RhoZCDP defaults to the Discrete Gaussian mechanism. Refer to Appendix Table A for details.

[32] Sensitivity is inherent of a query function and not chosen by users.
[33] We refer readers to our interactive notebook on the Laplace mechanism for a mathematical interpretation of the mechanism.
[34] https://github.com/opendp/smartnoise-core/tree/develop/whitepapers/sensitivities
[35] As the tools evolve, the default mechanisms can change.
[36] The default mechanism may not always yield the highest accuracy. The selection of the mechanism can be influenced by various factors, such as the number of queries to be executed in one run. For instance, in some cases the Gaussian mechanism might exhibit improved performance over the Laplace mechanism due to its smaller noise scale.
[37] https://github.com/dsaidgovsg/benchmarking-differential-privacy-tools

Impact of dataset skew, scale, size, and epsilon value on the accuracy of the queries: Our experimental results demonstrate a correlation between the noise scale and the error (MRE) of a query: decreasing noise scale resulting in reduced error. Given the Laplace distribution's noise scale is inversely proportional to epsilon and directly proportional to the sensitivity of the query:

- The error is higher for epsilon values less than 1, which is a range recommended by some differential privacy pioneers. However, the error approaches zero as the epsilon increases.
- The sensitivity of queries increases as the spread of a distribution increases. And spread increases with low skew and high scale values. The neighbouring definition (bounded or unbounded) used to compute queries' sensitivities also affected the noise scale. Notably, the noise scale is directly related to the privacy protection offered — higher noise scale results in high privacy protection.

Our findings further show that query performance improves with larger dataset sizes as the impact of noise diminishes. However, a large spread can still affect results due to probable outliers, even on large datasets.

We recommend that practitioners be aware of the high sensitivity of queries and compensate for it by selecting a higher value of epsilon, which improves accuracy. However, it's important to consider the privacy implications carefully.

Performance of tools on the queries:

- *Count* (refer to Figure 10)*:* The sensitivity of the count query is always one and independent of the skew and scale of a dataset. Hence, we excluded skew and scale variables from the count query results. Diffprivlib outperforms other tools for epsilon ≤ 0.1 and smaller datasets (observable for ≤10k data points) due to the underlying mechanism – Geometric truncated (Refer to Table A of Appendix) - which improves accuracy by mapping the negative count value to 0.

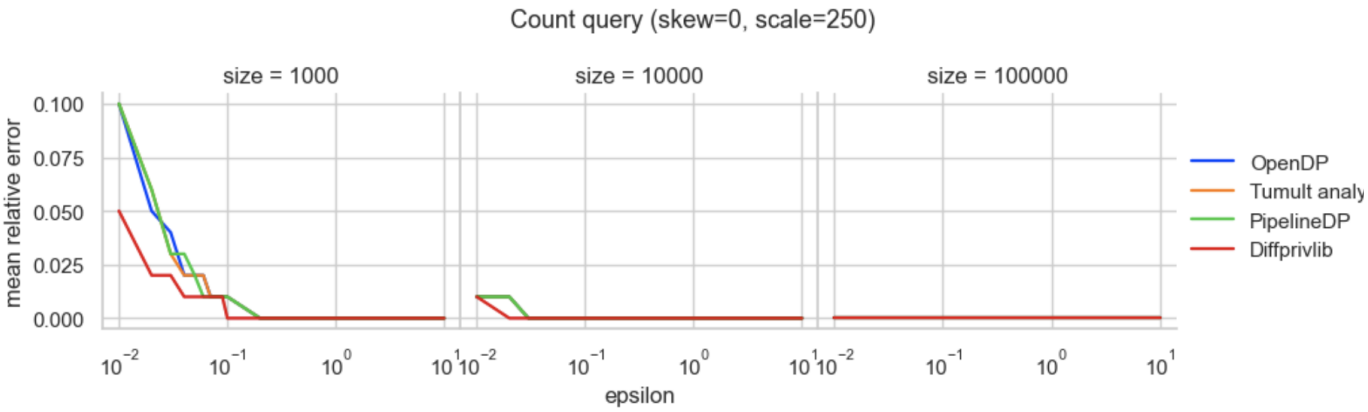


Figure 10: Mean relative error of the count query for experiments with synthetic datasets.

- *Sum* (refer to Figure 11)*:* Tumult Analytics and PipelineDP perform better than the other tools for epsilon less than 1. This is due to the unbounded definition for sensitivity calculation used by Tumult Analytics and PipelineDP, while others use the bounded notion. Our synthetic datasets consist of negative and positive values, resulting in higher sensitivity for the bounded definition than the unbounded one. For example, if the maximum value is 100 and the minimum value is -5, with bounded definition, the sensitivity of the sum query

will be 100 - (-5) = 105, whereas for unbounded, it will be max(abs(-5), abs(100)) = 100. However, please note that bounded differential privacy will give stronger privacy protection than the unbounded differential privacy.

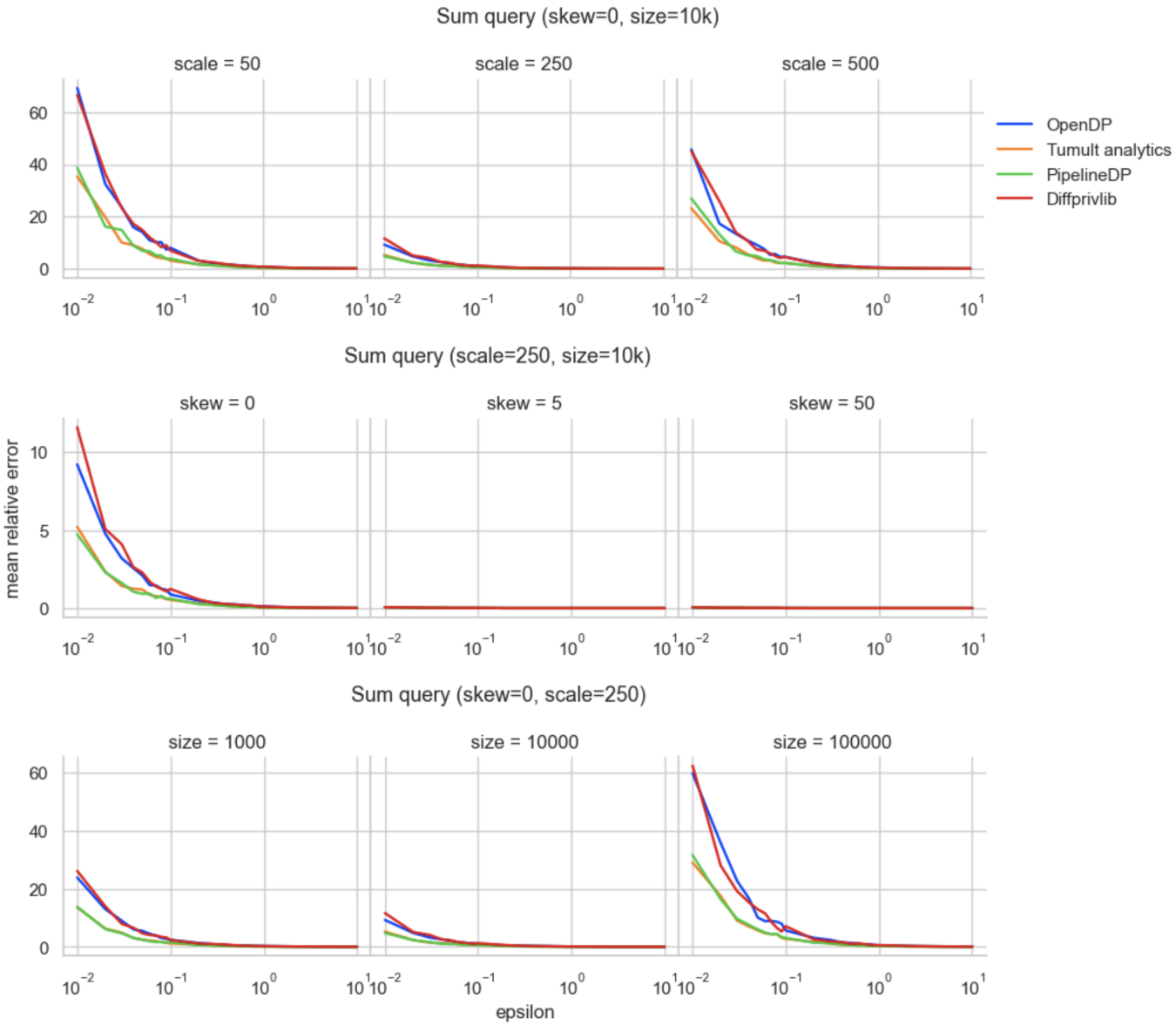


Figure 11: Mean relative error of the sum query for experiments with synthetic datasets.

- *Mean* (refer to Figure 12)*:* The sensitivity calculation for the bounded and unbounded definition is similar; therefore, we can observe nearly similar performance among the tools. Furthermore, all the tools use the Laplace mechanism as the default (or we used it in the code). Moreover, the skewness and scale impact follows a similar sum query reasoning. This is because, except for the Diffprivlib (uses NumPy mean API), other tools use sum and count queries as the building block for the computation.

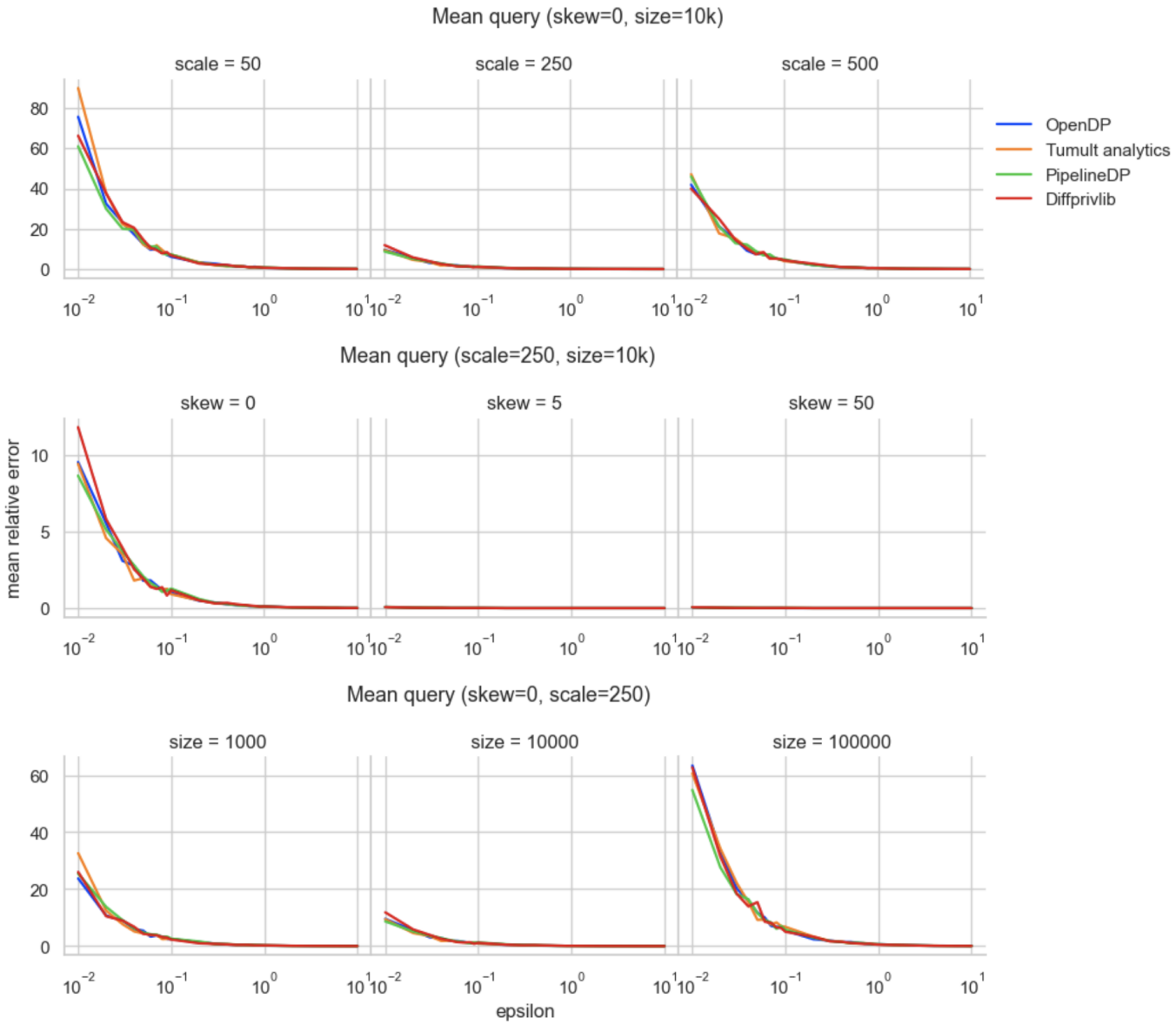


Figure 12: Mean relative error of the mean query for experiments with synthetic datasets.

- *Variance* (refer to Figure 13)*:* Tumult Analytics and PipelineDP perform better than other tools on large-scale and low-skew values (higher noise distribution spread). OpenDP performs better on the lower scale and large skew values where the epsilon is less than 1. The difference in performance can be attributed to the varying notions of bounded and unbounded neighbouring definitions, which result in different sensitivity calculations for the variance query. Diffprivlib performs the worst on the variance query, although its outputs are consistent with the actual variance values and are greater than zero. This is due to Diffprivlib's use of the Laplace Bounded Domain (refer to Table A of Appendix), which increases noise while maintaining output consistency for the variance query.

Most tools use sum and count queries as building blocks for variance computation or use similar mechanisms. As a result, the overall performance of the tools on variance demonstrates similar behaviour to the sum query across the three independent variables – *skew, scale,* and *size*, except for Diffprivlib, which uses NumPy's variance API.

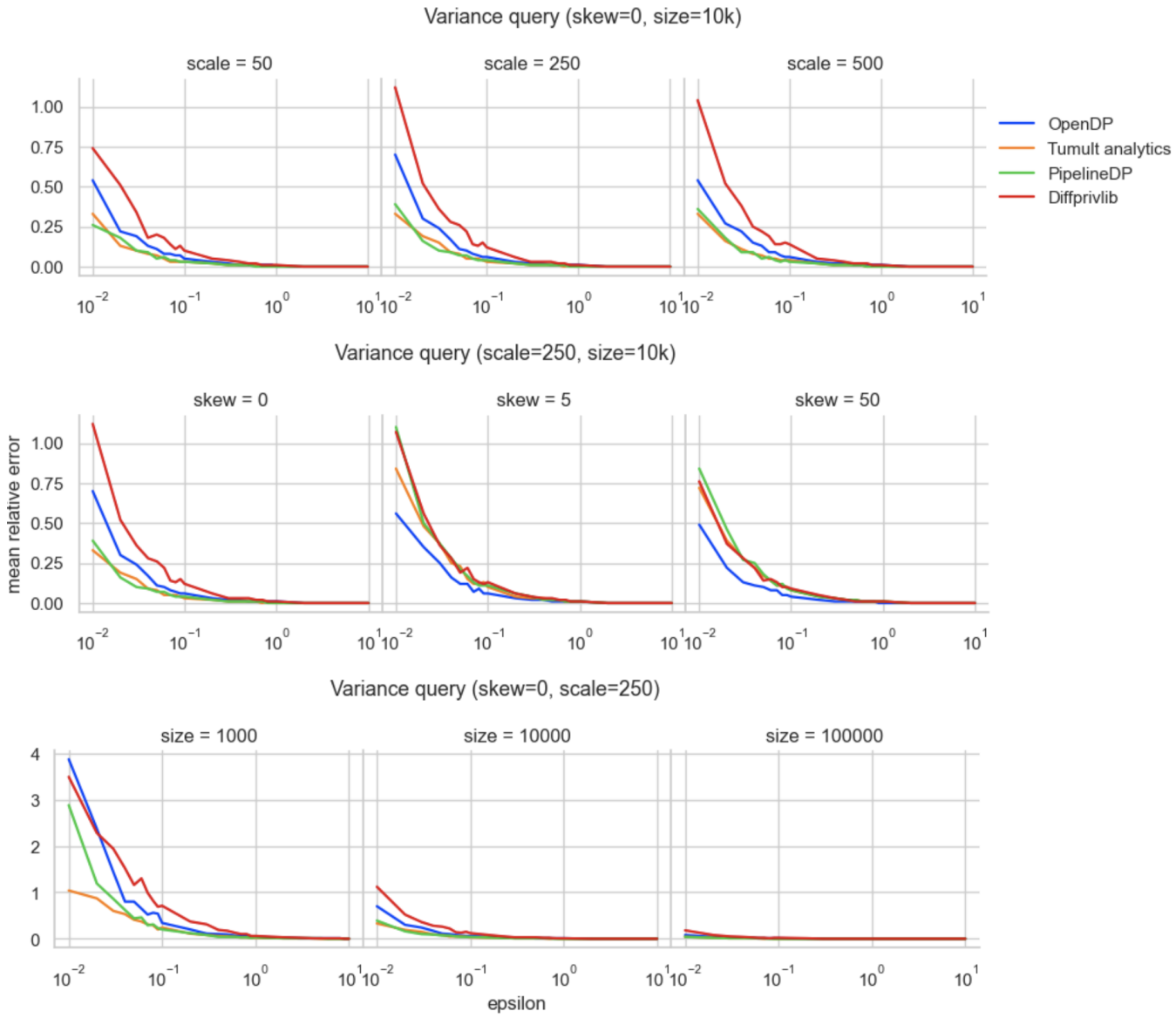


Figure 13: Mean relative error of the variance query for experiments with synthetic datasets.

Scalability analysis

Setup: In our scalability benchmark, we measured the execution time of the queries on datasets ranging from 1000 to 1 billion data points. We used fixed values of epsilon (0.01), *skew* (0), and *scale* (250), as existing literature [11] suggests scalability is largely independent of sampled noise. Further, our computation time only measured the query execution and excluded pre-processing steps, as practitioners' implementations can vary.

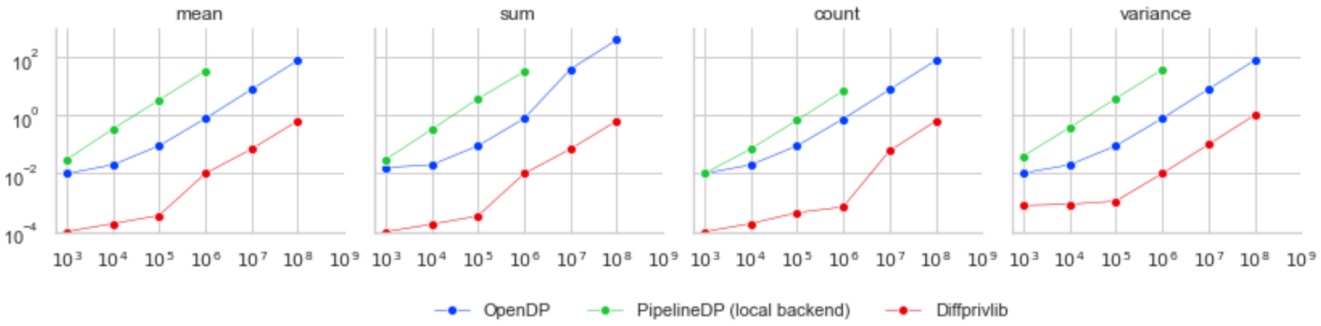


Figure 14: Execution time of the tools – supporting the standalone environment, experimented on synthetic datasets of varying sizes.

Results: For scalability assessment, we compare the tools based on the environment they were executed in – standalone and distributed.

Figure 14 shows the execution time of the Diffprivlib, PipelineDP, and OpenDP tools in standalone environments. The missing results in the figure indicate instances where the query computation for a library led to a memory error.

The Diffprivlib library performed best across all queries and scaled relatively well with increasing dataset size. This is because the underlying core functionalities of Diffprivlib are implemented in NumPy, as opposed to PipelineDP and OpenDP, which are Python- wrappers around their core functionalities in C++ and Rust, respectively. In general, all the tools do not scale on larger datasets, resulting in memory error or a substantial increase in time.

Also, our implementation of OpenDP includes using a binary search algorithm to search for a noise scale and prevent manual input. This feature ensures a differential privacy guarantee but also adds to the execution time when running the queries. For PipelineDP, the version we experimented with is tailored for GROUP BY type queries that create partitions (subsets of the dataset). There are pre-processing steps for partition handling (such as adding empty partitions for public partitions) and individual contribution bounding across and within partitions. Global aggregations (which we experimented with), i.e., computing statistics on all the data points, are treated as one partition. The unnecessary pre-processing steps add some computational overhead, but in upcoming versions, we can expect added functionality to prevent such overhead for global aggregations, which will benefit practitioners whose use case involves running GROUP BY queries.

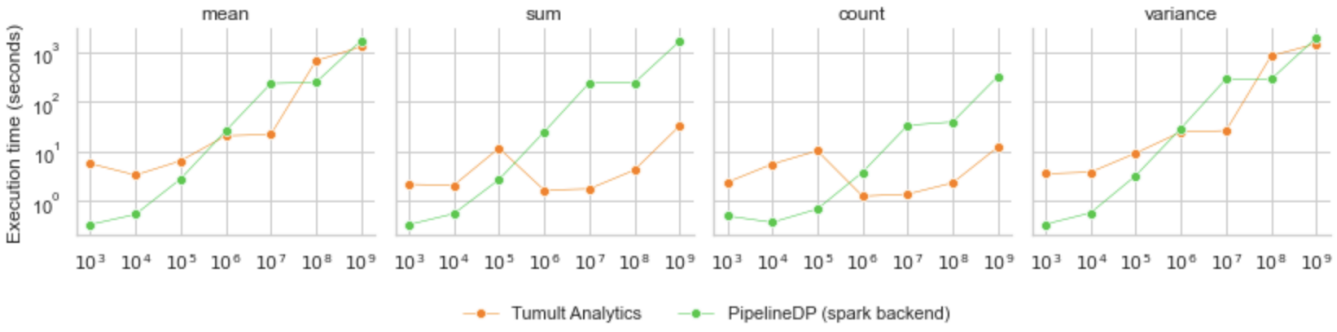


Figure 15: Execution time of the tools – supporting the distributed environment, experimented on synthetic datasets of varying sizes.

Figure 15 compares the execution time performance of Tumult Analytics and PipelineDP, two tools that support distributed computing, on the Spark engine. Tumult Analytics predominantly outperforms PipelineDP on datasets with

greater than 1 million rows across the queries. For a dataset of less than a million data points, PipelineDP largely outperforms Tumult Analytics. We also observed that Tumult Analytics has a higher memory consumption than PipelineDP. One reason is that Tumult Analytics creates temporary tables to avoid re-evaluating the query with fresh randomness for repeated queries.

5.5. GUIDANCE FOR PRACTITIONERS ON CHOOSING A LIBRARY/FRAMEWORK

Our guidance is based on global differential privacy (refer to Section 4.3.), which all the tools we evaluated support. Our evaluation suggests that while no single tool excels in qualitative and quantitative analysis, the differences among the tools can be bridged with engineering efforts. The teams behind the tools are responsive and active, and we expect libraries and frameworks to evolve rapidly.

Practitioners' needs may differ, with some prioritising high security for critical and production use cases, while others may value feature-richness, such as support for diverse analytical queries and differential privacy mechanisms. Some may prefer features that support accuracy adjustment and automated parameter search. Based on our experimental analysis, we have compiled desiderata in Figure 16, which practitioners can use when choosing a tool that suits their needs. For practitioners prioritising utility and execution time, our analysis in Section 6.2.2 and the summarised version in Figure 16 may be helpful.

Overall, we recommend using Tumult Analytics or OpenDP, which are production-ready and generally score well in the criteria listed in Figure 16.

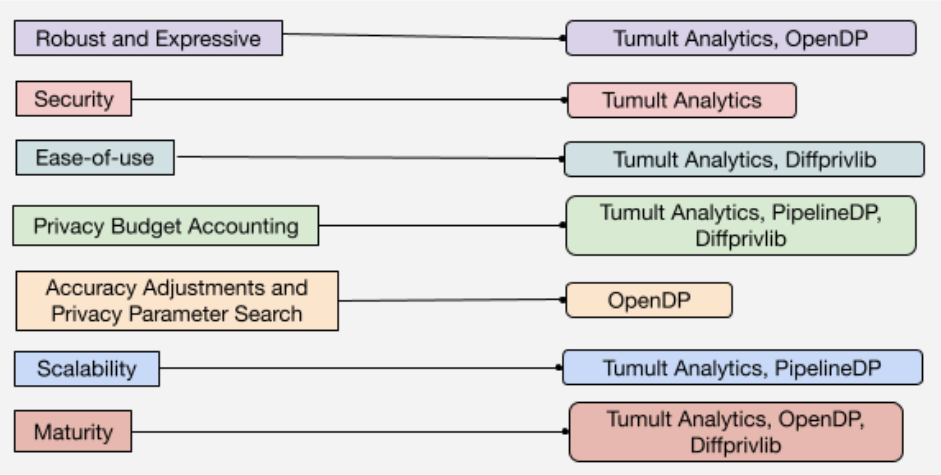
Notably, each library also has a unique value proposition. Tumult Analytics and OpenDP have an extensible underlying framework that can easily integrate privacy computation and provide robust guarantees. Diffprivlib has differential privacy machine learning capabilities. PipelineDP is still experimental but offers a groundbreaking feature by providing backend-agnostic data processing and privacy-utility trade-off visualisation features.

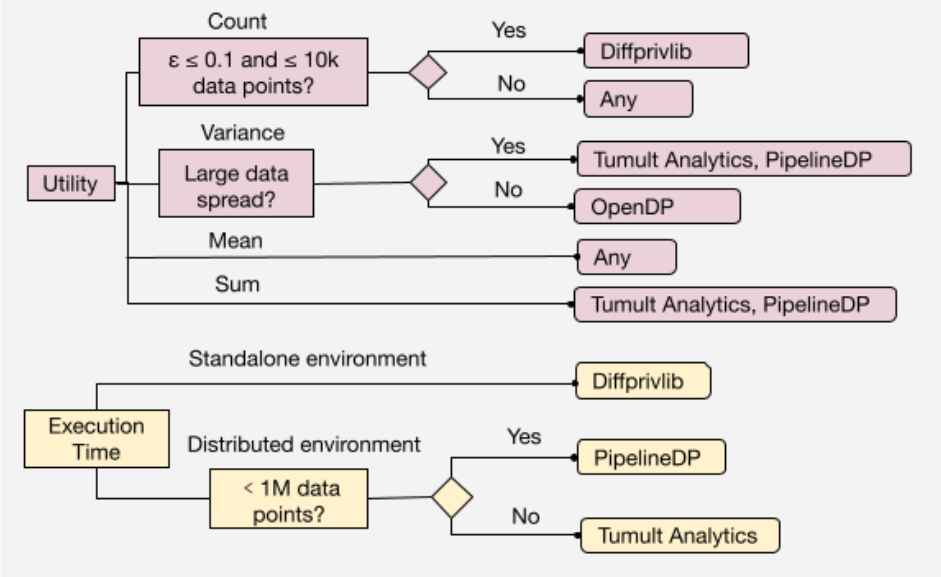
5.6. DISCUSSION

Our analysis is based on the latest version of the libraries and frameworks available at the time of our experiments. Significant changes are expected as they continue to mature with new features, improvements, and research in differential privacy. However, our analysis provides a helpful starting point for practitioners to get insights into these tools and choose the one that suits their needs.

In our qualitative analysis, we identified the key desiderata of the tools to the best of our knowledge, although it is not exhaustive. In our empirical quantitative analysis, we provided insights into the tools' implementation and how various factors can impact the accuracy of a query beyond the epsilon value, i.e., the scale, size, and skew of the data. Our analysis focused on the four most commonly used queries. However, there are other queries beyond these to experiment with (e.g., quantile and GROUP BY queries), including running multiple queries in a run that can optimise the privacy budget usage. In the future, we look forward to experimenting with other algorithms (such as Gaussian and Exponential algorithms), advanced privacy definitions (e.g., zero-concentrated differential privacy), and capabilities (e.g., population amplification and private dataset JOIN) offered by the tools.

As these tools evolve, it is paramount to get active support from the creators. Crucially, we received active support during our study to help us implement the AWS environment. We appreciate the creators' assistance and recommend following the future releases of the tools to take advantage of any new features or improvements. We encourage researchers and library creators to build upon our findings and support improving these tools for the betterment of society through responsible data sharing. We also value feedback from the community to improve our analysis further and ensure its accuracy.





Robust: The tool can be confidently used to obtain the desired privacy guarantee.

Expressive: Feature-rich to power real-world use cases and extensible enough to support the addition of new functionalities and features.

Security: The tool supports full security functionality as listed in Section 5.3., with additional measures taken to mitigate security concerns using the underlying libraries.

Ease-of-use: Limited understanding of differential privacy required, with a familiar syntax.

Privacy budget accounting: The tool can track the total budget spent and block further queries from exhausting it.

Accuracy adjustment and privacy parameter search: The tool can adjust accuracy and search privacy parameters.

Scalability: The tool can support arbitrary-sized datasets with distributed computing.

Maturity: The tool can be used for production use cases.

Utility: The accuracy of a differentially private query output.

Execution time: The time taken by the tool to compute a differentially private query.

Figure 16: Recommended tools based on empirical analysis and qualitative features that practitioners might gravitate towards. Note that the recommendation on utility and execution time is based on our experimental analysis with default settings of the tools.

6. CONCLUSIONS AND FUTURE WORK

The collection and dissemination of personal data by various organisations have become ubiquitous today, posing significant privacy concerns. Despite efforts to anonymise data, the re-identification of sensitive information remains a major issue, especially for vulnerable communities. The need for stronger privacy protection measures has become increasingly pressing in the face of rising cybercrime rates worldwide.

Differential privacy has emerged as a robust alternative to traditional anonymisation techniques, as it offers provable data privacy guarantees while allowing for meaningful data analysis. By adding controlled noise to datasets, differential privacy can prevent the re-identification of sensitive information and protect individuals' privacy. Moreover, it enables accurate and reliable data analysis without the loss of data quality and analytical value that often results from traditional anonymisation techniques. Differential privacy allows for exploring sensitive data across silos, potentially shortening data access times by relaxing the adversity of data request processes, and can fulfill some types of use cases.

The academic and industrial communities have developed various tools to facilitate the adoption of differential privacy. However, no single tool excels in all aspects, indicating that library creators can learn from one another. Although libraries and frameworks exist to apply differential privacy conveniently to sensitive data, a gap remains in providing a user-friendly interface for non-experts to visualise and enable interactive privacy-utility tradeoffs, such as tuning parameters and adjusting accuracy. This is crucial in addressing the negotiation challenges between data curators and analysts.

To address this need, the **Data Privacy Protection Capability Centre** at GovTech is working towards this goal by developing a friendly web interface – a one-stop portal for data controllers and analysts.

As the frontier of differential privacy continues to expand rapidly, collaborations among industry, government, nonprofit organisations, and academics will be helpful in its success in responsible data sharing.

7. REFERENCES

1. Garfinkel, Simson, John M. Abowd, and Christian Martindale. "Understanding database reconstruction attacks on public data." Communications of the ACM 62.3 (2019): 46-53.
2. Dwork, Cynthia, and Aaron Roth. "The algorithmic foundations of differential privacy." *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014): 211-407.
3. Wood, Alexandra, et al. "Differential privacy: A primer for a non-technical audience." Vand. J. Ent. & Tech. L. 21 (2018): 209
4. https://desfontain.es/privacy/real-world-differential-privacy.html
5. Dwork, Cynthia, et al. "Calibrating noise to sensitivity in private data analysis." *Theory of Cryptography:* Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3. Springer Berlin Heidelberg, 2006.
6. Bun, Mark, et al. "Differentially private release and learning of threshold functions." 2015 IEEE 56th Annual Symposium on Foundations of Computer Science. IEEE, 2015.
7. Alabi, Daniel, et al. "Differentially private simple linear regression." arXiv preprint arXiv:2007.05157 (2020).
8. Abadi, Martin, et al. "Deep learning with differential privacy." Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016.
9. Stemmer, Uri, and Haim Kaplan. "Differentially private k-means with constant multiplicative error." Advances in Neural Information Processing Systems 31 (2018).
10. Abowd, John M. "The US Census Bureau adopts differential privacy." Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018.
11. Garrido, Gonzalo Munilla, et al. "Do I get the privacy I need? Benchmarking utility in differential privacy libraries." arXiv preprint arXiv:2109.10789 (2021).
12. Garrido, Gonzalo Munilla, et al. "Lessons Learned: Surveying the Practicality of Differential Privacy in the Industry." arXiv preprint arXiv:2211.03898 (2022).
13. https://projects.iq.harvard.edu/files/opendp/files/opendp_white_paper_11may2020.pdf
14. Gaboardi, Marco, Michael Hay, and Salil Vadhan. "A programming framework for opendp." Manuscript, May (2020).
15. Berghel, Skye, et al. "Tumult Analytics: a robust, easy-to-use, scalable, and expressive framework for differential privacy." *arXiv preprint arXiv:2212.04133* (2022).
16. https://pipelinedp.io/
17. Holohan, Naoise, et al. "Diffprivlib: the IBM differential privacy library." arXiv preprint arXiv:1907.02444 (2019).
18. Mironov, Ilya. "On significance of the least significant bits for differential privacy." Proceedings of the 2012 ACM conference on Computer and communications security. 2012.
19. https://desfontain.es/privacy/renyi-dp-zero-concentrated-dp.html
20. Balle, Borja, Gilles Barthe, and Marco Gaboardi. "Privacy amplification by subsampling: Tight analyses via couplings and divergences." Advances in Neural Information Processing Systems 31 (2018).
21. Hay, Michael, et al. "Principled evaluation of differentially private algorithms using dpbench." Proceedings of the 2016 International Conference on Management of Data. 2016.
22. Bun, Mark, and Thomas Steinke. "Concentrated differential privacy: Simplifications, extensions, and lower bounds." *Theory of Cryptography: 14th International Conference, TCC 2016-B, Beijing, China, October 31-November 3, 2016, Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016.

23. Warner, Stanley L. "Randomized response: A survey technique for eliminating evasive answer bias." *Journal of the American Statistical Association* 60.309 (1965): 63-69.

Infographics are created using Flaticon.com

APPENDIX

	Description	Supporting libraries/frameworks
Privacy definitions		
Pure (ε)- differential privacy	Provides worst case or strict privacy guarantee, i.e., the privacy definition will always be true.	OpenDP, Tumult Analytics, PipelineDP, Diffprivlib
Approximate (ε, *δ*) - differential privacy	Provides relaxation of pure differential privacy and keeps room for privacy failure captured with the parameter ***δ***: the probability of failure that should be less than $1/n^2$.	OpenDP, Tumult Analytics, PipelineDP, Diffprivlib
zero-concentrated differential privacy	Provides stronger privacy guarantees than pure differential privacy. It includes the privacy parameter **ρ** (rho). Refer [19, 22]	OpenDP, Tumult Analytics
Privacy mechanisms		
Laplace	Noise is sampled and added from the Laplace distribution of domain (– inf, inf) without post-processing.	OpenDP, Tumult Analytics, PipelineDP, Diffprivlib
Laplace Bounded Domain	Samples output until one falls within the pre-defined range. While this mechanism is suitable to prevent not desired values, e.g., a negative variance value, or implemented in classifiers, it requires more tailoring to satisfy differential privacy	Diffprivlib
Laplace Truncated	If after adding noise to the true value, this output falls outside a pre-defined range. The output is mapped to the closest bound of the output range; e.g., a count output of less than zero could be mapped to the lower bound 0. If the domain bounds coincide with, e.g., changes in behaviour because the probability of returning values at the domain bounds is non-zero, it is recommended to use the Bounded Domain mechanism instead	Diffprivlib
Geometric (Discrete Laplace)	Employs a discrete variant of the Laplace mechanism by satisfying differential privacy with equality, and thus, producing tighter guarantees for integer-value outputs, and, in turn, higher accuracy	OpenDP, Tumult Analytics, PipelineDP
Geometric Truncated	Uses the same technique as Laplace Truncated, but the underlying mechanism is Geometric.	Diffprivlib
Gaussian	Noise is sampled and added from the Gaussian distribution of domain (– inf, inf) without post-processing.	OpenDP, Diffprivlib, Tumult Analytics, PipelineDP,
Discrete Gaussian	Modifies the Gaussian mechanism so that discrete noise may be	OpenDP, Tumult Analytics

	sampled without losing privacy or accuracy guarantees	
Exponential	Achieves differential privacy for categorical query outputs by randomly choosing a category proportionally to its utility value, i.e., a category with a higher utility score than another is as much more likely to be picked. The utility values decay exponentially, making the true results more likely to be picked while maintaining differential privacy.	OpenDP, Tumult Analytics, PipelineDP

Table A: The privacy definitions and mechanism referred to in the paper. The table has been referenced from [11]

	Bounded	Unbounded
Count	1	1
Sum	$M - m$	$max(abs(M), abs(m))$
Mean	$\frac{M-m}{n}$	$\frac{M-m}{n}$
Variance	$(M-m)^2\frac{n-1}{n^2}$	$(M-m)^2\frac{1}{n+1}$

Table B: Sensitivity calculations of the experimented queries with the bounded and unbounded neighbouring definitions. M is the upper bound and m the lower bound of a dataset. Refer to Harvard's papers on calculating the sensitivity of a query based on the neighbouring definition.