Differentially Private Online Aggregation

by

Harry Sivasubramaniam

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Masters of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2021

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Database operations are often performed in batch mode, i.e. the analyst issuing the query must wait till the database has been processed in its entirety before getting feedback. Batch mode is inadequate for large databases since queries can take several hours to process and often an analyst is satisfied with an approximation. Online aggregation greatly improves user experience and saves resources by providing continuous feedback through running confidence intervals. Further, it provides an interface for users to terminate early and allocate resources elsewhere once a sufficient accuracy level has been achieved. Until now, online aggregation has not been studied in a differentially private setting. In this work, we formulate differentially private online aggregation such that it captures the tradeoffs between privacy, accuracy, and usability. Further, we develop a family of differentially private mechanisms, which includes our optimal Gap mechanisms, for answering AVG, COUNT, and SUM queries with WHERE conditions. Also, we develop various optimizations to improve the accuracy of the Gap mechanism and empirically confirm that the Gap mechanisms preform the best overall.

Acknowledgements

First and foremost, I would like to thank my supervisor Xi He for introducing me to this exciting field and for being an excellent mentor. I am grateful for all the insight, support, and encouragement Xi has provided me over the course of my masters. I would also like to thank Florian Kerschbaum and Gautam Kamath for being on my thesis committee and providing valuable feedback.

I am grateful to Joseph Cheriyan for providing me with my first research experience and for constantly guiding me throughout my undergraduate study.

To my fellow differential privacy buddies Shubhankar, Karl, Chang, Shufan, Christian, David, Pranav, and Vikrant. I appreciate all the valuable discussions we have had over the past 2 years. Also, thank you for doing mock presentations with me in preparation for my thesis defence. Special thanks are due to David Li who was always eager to help me implement and run experiments.

During my time at Waterloo, I have met several interesting characters who have influenced my growth as a person. I would like to thank all my friends from the party office, Pure Math Club, UW Tricking Club, and BJJ. In particular, Justin, Logan, Sharat, Cedric, Ed, Akshay, Abhinav, Sifat, Clair, Maggie, Mohamad, Pranav, Sanjay, Sidhant, Zouhaier, Lily, Doc Trivial, Sally, Adam x 2 (Brown and Jaffe), Henry, Ilia, Sam, Kaleb, Bryan, Sean, Felix, Simon, Stephen, Letian, Jason, Bailey, Shelly, Madison, Timothy, Manuel, Oliver, Oscar and everyone else who has made life in Waterloo an enjoyable experience.

Lastly, I would like to thank my parents and siblings for their love and support.

Dedication

To the two beautiful ladies in my life, my grandmothers Sivaneswary and Nageswary.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Aggregation in relational databases plays an important role in policy evaluation, medical applications, and gathering financial and business insights. A common approach to handle aggregate queries is the use of online aggregation to return fast and accurate approximations that improve at each time step. Due to the sensitive nature of such data, an individual's privacy is of the utmost concern. The main problem is that aggregation does not guarantee privacy [17] and there are many instances of successful reconstruction attacks [9] after the release of aggregate statistics that have not been perturbed appropriately.

Due to the rapid growth of available data, many aggregation tasks can take several minutes and even hours to compute exactly. For example, when dealing with encrypted databases or multiple data federations as in SAQE [5], the overhead from the cryptographic protocols can often make the runtime 1000x slower. This is because the databases need to be padded with tuples forcing the worst-case runtime as private information can be inferred from the runtime. This is unacceptable as data analysis tends to be an interactive process with the user issuing a set of queries and using the results to determine the next set of queries. There are many solutions such as approximate query processing and online aggregation [21], for database systems, that leverage some loss in accuracy for significantly improved response times. In online aggregation, the user gets estimates of an aggregate query in an online manner as soon as the query is issued. For example, if the user issues the following query "SELECT AVG(Salary) FROM D" and the true answer is 100,000 and would normally have taken several minutes to compute. An online aggregation system, after k milliseconds, might output something like $[120,000 \pm 30,000]$ with probability at least 95%. This confidence interval will keep on shrinking as the system gets more and more samples. To date, there has been no work that tackles one of the biggest modern roadblocks which is data privacy, for the online aggregation setting. Due to government

policy and regulations, a lot of interesting analyses may not be permitted using non-private aggregation methods.

Differential privacy (DP) [13, 15] has emerged as the gold standard for measuring and protecting an individuals privacy. Intuitively, it ensures a level of user privacy while still allowing for the release of meaningful aggregate statistics by guaranteeing that the output on similar-looking databases is close. A privacy parameter ϵ is used to capture the privacy loss guarantees of a differentially private mechanism, where larger ϵ corresponds to greater privacy loss. Furthermore, differential privacy has several composition properties making it easy to track the privacy loss when answering multiple private queries on the same database. The first is sequential composition which allows one to track the privacy loss of applying two differentially private mechanisms by simply adding the ϵ values. When two DP mechanisms take as input separate partitions of a database, the overall privacy loss is instead the maximum loss of the two mechanisms by the parallel composition property of differential privacy. A classic way to achieve differential privacy for a query with a real-valued output is to perturb the output with Laplace noise inversely proportional to ϵ and the sensitivity of the query, where sensitivity refers to the maximum difference in values between similar inputs.

Online aggregation releases a sequence of estimates based on the aggregate over all the data seen so far at every time step. A naive way to achieve differentially private online aggregation is to perturb each estimate with Laplace noise, but each release requires a fresh privacy budget. The total privacy budget is split among all the timestamps and the last estimate may be very poor due to large noise. On the other hand, rather than releasing at every time step, we may release one estimate at the very end with the full privacy budget. This estimate has very good accuracy, but the user experience will be poor, as the user has to wait for too long. Hence, the introduction of a privacy constraint in online aggregation poses new challenges. In particular, the challenges are optimizing the privacy budget to maintain accuracy, handling accuracy loss from the privacy mechanism, and designing simple interfaces that allow users to prioritize time-steps based on accuracy needs.

A related setting to differentially private online aggregation is differentially private streaming and privacy under continual observation [14, 8, 27, 16, 7]. In these settings, an online sequence of updates is being observed and the requirement is to privately compute some statistic. For example, Chan et al. and Dwork et al. [8, 14] study the setting in which there is a stream of 1's and 0's and the goal is to privately maintain a count of the 1's. Chan et al. [8] develop the binary mechanism, which privately updates the count using a binary tree structure, and the estimate for the count at time step t has an error of $O(\frac{(\ln t)^{3/2}}{\epsilon})$. Continual observation is different as the input database is static in online

aggregation and also the database size is known ahead of time. While there exists prior work on differentially private approximate query processing (AQP) such as SAQE [5], the focus has been solely on the offline setting. In SAQE, Bater et al. develop various sampling-based algorithms that quickly output an approximate answer to SQL queries over the union of multiple datasets. The algorithms in SAQE cannot be directly extended to the online setting, as there is no mechanism that can optimize the privacy budgets over estimates released at different time steps.

In this work, we first define a set of desired properties for a differentially private online aggregation mechanism. It is desired that such mechanism provide (i) ϵ-DP, (ii) bounded confidence intervals that shrink in length over time, (iii) a well-defined trend line that can be computed before query execution. We also introduce a family of mechanisms which we refer to as Gap mechanisms to optimize the utility of the released estimates. Unlike the binary mechanism which is designed to output at each time step, the Gap mechanisms reduce the error from private noise addition by releasing a noisy aggregate and then waiting for a specified gap in time. During this gap, sufficiently many new samples are accumulated in a manner that warrants creating another noisy partial sum. The structure of these gap based mechanisms combined with the sequential and parallel composition properties of differential privacy allows for logarithmic and even constant error from private noise addition as opposed to error linear in the number of time steps. Furthermore, we identify several optimizations related to which noisy sums to aggregate together, privacy amplification [30, 2] and improvements in common concentration bounds [3, 4]. We also provide experiments comparing the various mechanisms on the flight delay dataset [1]. The results show that the Gap mechanisms are the best choice overall.

The main contributions of our work are as follows:

- We are the first to introduce differential privacy to online aggregation and formulate the desired features for differentially private online aggregation.
- We design and implement several ϵ-differentially private online aggregation mechanisms for a popular class of relational database queries including AVG, COUNT and SUM queries with and without WHERE conditions. This includes a family of Gap-based mechanisms that outperforms the rest.
- We develop optimizations that improve the results of the Gap mechanism. We do so by optimizing the budget splitting when given WHERE conditions, leveraging privacy amplification results, and using improved concentration bounds.
- We provide a tighter analysis of the Laplace concentration bound from [8] that can achieve up to a $\sqrt{2}$ factor improvement in many cases.

This thesis is organized as follows. We first look at the related work on online aggregation in the non-private setting and then on approximate query process with differential privacy guarantees in Chapter 2. In Chapter 3, we provide the preliminary definitions and define online aggregation in the non-private setting along with providing the necessary theorems and lemmas that will aid in designing the private version. Next, in Chapter 4, we formally define the problem of differentially private online aggregation and contrast it with the non-private version. In Chapter 5, we introduce several baseline mechanisms along with the optimal Gap mechanisms for AVG queries without WHERE condition and then AVG, COUNT, and SUM queries with WHERE condition. Furthermore, in Chapter 6, we provide optimizations based on privacy amplification, improved concentration bounds for the sums of Laplace variables and for sampling without replacement. An extensive evaluation of our algorithms over various queries and parameter ranges is provided in Chapter 7. Finally, we conclude this thesis and discuss future work in Chapter 8.

Chapter 2

Related Work

In this section, we will discuss the related work. First, we look at online aggregation in the non-private setting and recent works that seek to improve the accuracy of estimators [26]. Next, we will look at differentially private aggregation under various settings and contrast it with our work.

The initial task of adapting relational databases queries from batch mode, in which the user received feedback only after the entire query had been processed, to an online manner was done by Hellerstein, Haas and Wang [21]. By using Hoeffding's inequality, Hellerstein et al. demonstrated how to construct running aggregates and confidence intervals for common statistical queries such as AVG, COUNT, SUM, VARIANCE, and STD DEV. Furthermore, Hellerstein et al. provided implementation detail on how to modify traditional SQL databases to support running aggregates. This is important as the validity of the statistical estimators used relies on the assumption that the records of the database are accessed in random order. Our work will also handle AVG, COUNT, and SUM queries under pure differential privacy but will omit VARIANCE and STD DEV as they have high sensitivity and are difficult to privately estimate in a general setting.

For queries over single tables, non-private online aggregation is quite easy. Simply take samples from the database repeatedly and compute the average over the sampled elements. Unbiased estimators can be generated by scaling up and common statistical tools provide confidence intervals. The challenging queries in online aggregation are ones involving multiple tables and JOIN operations [18, 25, 19]. By sampling tuples from each table and taking the JOIN of the sampled tuples, the result is a sample of the JOIN results. However, estimating the aggregate of the joint tuples is not as straightforward as the elements are no longer independent, making it difficult to apply the same statistical

tools as the single table algorithms. In [18], Haas and Hellerstein proposed the ripple join algorithm which takes samples in a round-robin fashion and keeps all the sampled tuples in memory. It then joins all the tuples in memory with the new tuple that has been sampled. Li et al. [25] identify two problems with ripple join; (i) its performance depends on the fraction of the selected tuples that actually join and (ii) makes the unreasonable assumption that tuples in each table are in random order. To remedy this, Li et al. propose wander join, which tackles these two issues by performing random walks over the underlying join graph. Solving queries with JOIN constraints are quite difficult without privacy constraints but with privacy constraints even single table queries become challenging. Specifically, (i) join queries are highly sensitive [29, 12, 22]; (ii) both ripple join and wander join uses correlated sampling which will further increase the sensitivity of the query output. Macke et al. [26] identify how approximate query processing (AQP) systems can be improved by using concentration inequalities tailored specifically for sampling without replacement and a novel range trimming technique. The setting of Macke et al. is slightly different as they are taking a sufficiently large sample and approximating based on the single sample rather than maintaining running aggregates, but the techniques still apply to non-private online aggregation. Unfortunately, the techniques used by Macke et al. are difficult to adapt to private online aggregation as it relies on maintaining the maximum and minimum values of the data sampled so far. These are quantities that are challenging to make private and showcase how even single table online aggregation is difficult with DP.

A related setting to private online aggregation is differentially private continual observation [8, 14] in which data aggregators continually release updated statistics while preserving individual privacy. The privacy guarantee for online aggregation is different as the input is a static database in which the total number of rows is known while in private continual observation, the input is a stream of data of potentially unbounded size. A fundamental problem addressed by Chan et al. and Dwork et al. [8, 14] is that of maintaining a private counter given a stream of 0's and 1's. The continual observation setting is different from online aggregation as there is no notion of error from sampling when maintaining the current count privately. A naive solution for private counting under continual observation is to add Laplace noise to each item in the stream resulting in $O(\frac{\sqrt{t}}{\epsilon})$ error at time step t. Chan et al. [8] develop the binary mechanism which elegantly partitions the data so that the overall mechanism is ϵ-differentially private and the error at time step t is $O(\frac{(\ln t)^{3/2}}{\epsilon})$. While a similar approach can be used to solve online aggregation tasks, it is often not ideal and in our setting, our proposed Gap mechanisms have the much lower private error $O(\frac{\sqrt{\ln t}}{\epsilon})$ from the addition of Laplace noise. The key difference between COUNT queries in online aggregation and counting under continual observation is that when convenient, we may choose not to use additional samples when estimating the true statistic but the

continual counters seek to keep updating when new data is made available. This allows for the use of less noise as old aggregates can be reused until sufficient data is made available before generating the next noisy partial sum. The binary mechanism may still have a place in online aggregation but it tends to be more useful in the cases where the privacy budget is really high and so the error from sampling is dominating the error from the private noise.

Related work in the area of private approximate query processing is SAQE [5] by Bater et al. SAQE enables users to query the union of data from multiple federations without revealing any extra information to the user or the other federations. To achieve this, Bater et al. use strong cryptography protocols which incur extremely large overhead costs making exact computation of queries difficult. Bater et al. remedy this by using various sampling strategies to create private estimators while leveraging privacy amplification results [30] to decrease the error from privacy when sampling. Our work is similar in that we address the same set of queries, namely AVG, SUM and COUNT, and also leverage sampling and privacy amplification results. The main difference is that Bater et al. sample once and provides an estimate while our work offers continuous feedback. Our private online aggregation techniques can be applied to the federated settings and we leave this as future work.

There is a large body of work on private statistics and especially on private mean estimation [24, 6, 23, 10]. Often, these works focus on a specific distribution or families of distributions and seek to obtain tight upper and lower bounds on the error. Common techniques include choosing a suitable truncation interval so that no data points get truncated with high probability. The benefit of this is that a smaller interval means the query is less sensitive and so less noise needs to be added to make it private. Our work is slightly different as we do not make any distributional assumptions on the data other than that it is contained in some interval $[a, b]$ which the database system knows ahead of time. This matches the non-private work which tends not to make distributional assumptions on the data. Another difference is that our source of randomness is from the private noise addition and from shuffling the data while the other is from the distributional assumptions on the data and private noise addition. In future work, we will explore how the two areas can be combined so that private online aggregation can be optimized given more assumptions on the underlying distribution.

Chapter 3

Preliminaries

Let $\mathcal{D}_n$ be the set of all relational databases instances on n elements. A given row $D \in \mathcal{D}_n$ consists of one table where the columns correspond to the attributes and the n rows consist of a unique identifying key along with values for each column. We further assume the rows of the database D are partitioned into blocks $B_1, ..., B_m$ of equal size B, except for potentially the last block. The size of the blocks is generally determined by the hardware limitations (ex. cache size) and a single block may be accessed and processed at each time step.

Our setting assumes that the rows have been shuffled uniformly at random and this may be implemented by paying a one-time up-front cost to shuffle the data. The benefit of pre-shuffling is improved cache locality and ease of implementation as we do not need to implement sampling without replacement directly. Further, we assume that the numerical values we wish to aggregate are bounded in some range $[a, b]$ which the user provides beforehand. The former assumptions will allow for the use of concentration bounds to derive confidence intervals.

3.1 Online Aggregation

Let D be a table containing n rows $t_1, ..., t_n$. We will focus on SELECT queries $Q : \mathcal{D}_n \to \mathbb{R}$, of the following form

```
SELECT op(expression) FROM D;
```

and

```
SELECT op(expression) FROM D WHERE predicate;
```

where 'op' can be common aggregation functions like AVG, COUNT, SUM, STD DEV and VARIANCE. Further, 'expression' is an arithmetic expression involving the attributes of R and 'predicate' is an arbitrary predicate involving the attributes of D. Let $v(i)$ be the value of the expression when applied to row t_i given that the row satisfies the predicate, otherwise $v(i) = 0$.

To illustrate this, consider the following example for computing the average arrival delay of flights at Phoenix airport in the month of January.

```
SELECT AVG(arr_del15) FROM flights WHERE airport="PHX" AND month=1;
```

Let $v(i)$ be the arrival delay value of t_i and let $u(i) = 1$ be the indicator for when the tuple satisfies the predicate of 'airport="PHX" AND month=1'. Then we wish to compute

$$\frac{\sum_{i=1}^{m} v(i)u(i)}{\sum_{i=1}^{m} u(i)}.$$

Some examples of other queries we handle in this work include.

```
Q1: SELECT AVG(arr_del15) FROM flights;
Q2: SELECT AVG(arr_del15) FROM flights WHERE airport="PHX" AND month=1;
Q3: SELECT COUNT(arr_del15) FROM flights WHERE airport="PHX" AND month=1;
Q4: SELECT SUM(arr_del15) FROM flights WHERE month=1;
```

An online aggregation task seeks to maintain a running aggregate for queries of the above form while giving the user some level of control over the accuracy and priority of queries answered. Let μ denote the true aggregate value which is obtained by running the specified aggregate over all applicable tuples in the database. The user also chooses a confidence parameter $p \in (0, 1)$ and at each time step t, the system outputs an approximate aggregate μ_t and precision parameter α_t with a guarantee that μ_t lies in interval $[\mu \pm \alpha_t]$ with probability at least p. i.e.

$$\Pr[\mu_t \in [\mu - \alpha_t, \mu + \alpha_t]] \geq p.$$

Since the rows are shuffled at random, a linear scan is essentially equivalent to sampling with replacement. Furthermore, the values being bounded in the user specified range

$[a, b]$ allows us to apply concentration bounds such as Hoeffdings inequality to compute confidence intervals for AVG. By Hoeffding's inequality (theorem 5), we can show that we get p-confidence intervals by taking $\mu_t = \frac{\sum_{i=1}^{t} v(i)u(i)}{u(i)}$ and

$$\alpha_t = (b-a)\left(\frac{1}{2n}\ln(\frac{2}{1-p})\right).$$

3.2 Differential Privacy

Definition 1 (Differential Privacy)**.** A randomized algorithm A is ϵ-Differentially Private (DP) if for all events R in the output space of A, and for all databases $D, D' \in \mathcal{D}_n$ differing on a single element, we have $\Pr[A(D) \in R] \leq e^{\epsilon} \Pr[A(D') \in R]$.

A classic method of achieving differential privacy is the Laplace Mechanism [15][13].

Theorem 1 (Laplace Mechanism)**.** *Let $f : \mathcal{D}_n \to \mathbb{R}$ be any function. The global sensitivity of f is defined as $GS_f = \max_{D \sim D'} |f(D) - f(D')|$. Then the algorithm $A(D) = f(D) + Lap(GS_f/\epsilon)$, satisfies ϵ-DP, where $Lap(x|b) = \frac{1}{2b}e^{-\frac{|x|}{b}}$.*

The accuracy of the Laplace Mechanism can be measured using the following lemma.

Lemma 1. *For a query function $f : \mathcal{D}_n \to \mathbb{R}$, the noisy answer $\hat{f}(D) = f(G) + Lap(GS_f/\epsilon)$ satisfies*

$$\Pr[|\hat{f}(D) - f(D)| > tGS_f/\epsilon] \leq e^{-t}.$$

Theorem 2 (Sequential Composition)**.** *Let $A_i : \mathcal{D} \to R$ be an ϵ_i-differentially private mechanism for each $i \in [k]$. Then the composition $A = (A_1, ..., A_k) : \mathcal{D} \to R^k$ satisfies $\sum_{i=1}^{k} \epsilon_i$-DP.*

Theorem 3 (Parallel Composition)**.** *Let $A_i : \mathcal{D} \to R$ be an ϵ_i-differentially private mechanism for each $i \in [k]$. Given a database $D \in \mathcal{D}$, let $\{D_i\}_{i=1}^{k}$ be a disjoint partition of D. Then the composition $A(D) = (A_1(D_1), ..., A_k(D_k))$ satisfies $\max_i \epsilon_i$-DP.*

Theorem 4 (Post-Processing)**.** *Let $A : \mathcal{D} \to R$ be an ϵ-differentially private mechanism. Let $f : R \to R'$ be an arbitrary randomized mapping. Then $f \circ A : \mathcal{D} \to R'$ satisfies ϵ-DP.*

3.3 Concentration Bounds

Theorem 5 (Hoeffding's inequality)**.** *Let $\mathcal{D} = x_1, ..., x_N$ be a set of N values in $[a, b]$. Let $X_1, ..., X_n$ denote a random sample with replacement from $\mathcal{D}$. Then, for any $t > 0$, we have*

$$\Pr\left[\sum_{i=1}^{n} \frac{(X_i - \mathbb{E}\, X_i)}{n} \geq t\right] \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

and

$$\Pr\left[\sum_{i=1}^{n} \frac{(X_i - \mathbb{E}\, X_i)}{n} \leq -t\right] \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right).$$

Remark. *By union bound the above can be combined to get*

$$\Pr\left[\left|\sum_{i=1}^{n} \frac{(X_i - \mathbb{E}\, X_i)}{n}\right| \geq t\right] \leq 2\exp\left(-\frac{2nt^2}{(b-a)^2}\right).$$

Lemma 2 (Hoeffding)**.** *Let $\mathcal{D} = x_1, ..., x_N$ be a finite population of N real points. Let $X_1, ..., X_n$ denote a random sample with replacement from $\mathcal{D}$ and $Y_1, ..., Y_n$ denote a random sample without replacement from $\mathcal{D}$. If $f : \mathbb{R} \to \mathbb{R}$ is a continuous and convex function, then*

$$\mathbb{E}\, f\left(\sum_{i=1}^{n} X_i\right) \leq \mathbb{E}\, f\left(\sum_{i=1}^{n} Y_i\right)$$

Remark. *The above lemma allows us to use Hoeffding's inequality even when the sampling is without replacement.*

When combining several noisy queries, the following concentration bound on sums of Laplace variables is useful to measure the overall error.

Theorem 6 (Sums of Independent Laplace Distributions)**.** *Let $Y = \sum_i Y_i$ where each $Y_i \sim Lap(b_i)$ are independent. Let $b_{\max} = \max_i b_i$, $\nu \geq \sqrt{\sum_i b_i^2}$ and $0 < \lambda < \frac{2\sqrt{2}\nu^2}{b_{\max}}$ suppose $0 < \delta < 1$, then*

$$\Pr\left[|Y| > \lambda\right] \leq \exp\left(-\frac{\lambda^2}{8\nu^2}\right). \tag{3.1}$$

Corollary 1 (Measure Concentration)**.** *Let $Y, \{b_i\}_i, b_{\max}$ and ν be defined as before (Theorem 6). Suppose $0 < \delta < 1$ and $\nu > \max\{\sqrt{\sum_i b_i}, b_{\max}\ln(\frac{2}{\delta})\}$, Then,*

$$\Pr\left[|Y| > \nu\sqrt{8\ln\left(\frac{2}{\delta}\right)}\right] \leq \delta.$$

An immediate corollary of the above is this simple but slightly less tight bound.

Corollary 2. *Let* $Y = \sum_i Y_i$ *where each* $Y_i \sim Lap(b_i)$ *and suppose* $0 < \delta < 1$*, then*

$$\Pr\left[|Y| > \sqrt{8 \sum_i b_i^2} \ln\left(\frac{2}{\delta}\right)\right] \leq \delta.$$

Chapter 4

Problem Formulation

Online aggregation is a great feature for query answering in database systems [21] as it provides instant feedback and allows the user to save on resources via early termination once a desired approximation has been achieved. In online aggregation, the data analyst receives a sequence of estimates with confidence levels in an online fashion as soon as the query is issued. Realizing this online aggregation feature over sensitive data with differential privacy guarantee is non-trivial due to this additional privacy constraint.

In this chapter, we will formally define the setup and problem statement for *differentially private online aggregation*. The main problem is that of picking a utility metric that captures the requirements of private online aggregation. Specifically, (i) the overall algorithm should satisfy ϵ-DP; (ii) a sequence of improving estimates and confidence intervals over time should be released, just as the non-private setting; (iii) additionally some level of control should be given to the user over the speed of updates based on their preferences. The second challenge is to design differentially private algorithms for private online aggregation that meet these requirements and to find an optimal algorithm. We will show the problem and opportunities in algorithm design after our problem formulation. These include improving the estimates by leveraging sampling-based privacy amplification and tightening confidence intervals with better concentration bounds.

4.1 Desiderata for Private Online Aggregation

Desideratum I: Differential privacy guarantee. In private online aggregation, we require the cumulative privacy loss over the entire output sequence of estimates and confidence intervals to be at most ϵ. It is clear each new estimate must be made private

but extra care must be taken to also make the confidence intervals private when WHERE conditions are involved. Unlike the online setting in which the total privacy loss must be ϵ, in the offline private aggregation setting, the mechanism takes as input a sample or the entire data and release a single query answer that satisfies ϵ-DP.

Desideratum II: Continuous observations. Statistical and graphical interfaces are desired so that users may observe the processing and get a sense of the current level of precision when executing queries. The set of interfaces must be extensible so that each aggregate function and or combination of functions can be presented.

Desideratum III: Control of Time. Users should be able to terminate processing of a query at any time. This gives a trade-off between time and precision in non-private online aggregation. In private online aggregation, the trade-off is now between time, precision, and total privacy loss. Also, in the non-private setting, accuracy will always increase as time progresses but this isn't necessarily true of any arbitrary private mechanism for online aggregation. Thus, in private online aggregation, we desire a class of DP algorithms rather than just a single algorithm used in the non-private setting. Each of the DP algorithms has its own trend lines corresponding to error from sampling and error from privacy. These trend lines should be made available to the data analyst so that they may choose the algorithm that best matches their preferences. Alternatively, if a utility function is provided, the algorithm that will score the best should be chosen. This can be captured by using weighted priority functions and choosing the algorithm whose output sequence will minimize the sum of the weighted confidence interval lengths which we will refer to as the score of an algorithm.

The original goals of non-private online aggregation [21] also focused on the fairness of updates when there are multiple simultaneous aggregation tasks. These fairness goals are more applicable when dealing with GROUP BY queries which we leave to future work. Also, they have main performance goal, minimum time to accuracy, and secondary performance goal of minimum time to completion. In differentially private online aggregation, the former can be obtained by picking the appropriate algorithm based on the trend lines and the latter must be considered as overly complex private mechanisms can add a lot of overhead. Non-private online aggregation also has a performance goal of consistent pacing between new updates and is easily achievable since confidence intervals are shrinking as new rows are sampled. Such requirements in private online aggregation are undesirable since attempting to update estimates uniformly will cause the confidence intervals to degrade over time. The pacing requirement in private online aggregation is not as rigid and can be realized through the choice of utility function and trend selection.

4.2 Problem Setup

Given a database D consisting of n rows. Let $Q : \mathcal{D}_n \to \mathbb{R}$ denote the aggregate query and let $\mu = Q(D)$ be the true value. Let a block be the basic unit consisting of B rows for query processing. Rather than have the system process queries row by row, we generalize it to block by block. Consider all the rows of this database are stored continuously by blocks, $B_1, \ldots, B_m$, where $m = \lceil \frac{n}{B} \rceil$. At each time step $t = 1, 2, \ldots$, a new block is processed and an approximation of μ along with a confidence interval based on a user-specified probability p is released. The queries we wish to answer in this work include AVG, COUNT, and SUM given arbitrary WHERE conditions as seen in section 3.1.

This problem statement directly follows the notation introduced in the first paragraph of Sec 4.2. This problem statement is a mathematical formulation of all the desiderata proposed in Sec 4.2.

Problem Statement: Given a database D consisting of m blocks of data, an aggregate query $Q : D \to \mathbb{R}$, a privacy budget ϵ, we say an algorithm $\mathcal{A}$ is ϵ-differentially private online aggregation algorithm, if $\mathcal{A}$ releases a sequence of $(\mu_t, \alpha_t)_{t=1}^{T}$, where $T \leq m$, such that for $t = 1, 2, \ldots$

1) $\mathcal{A}$ satisfies ϵ-differential privacy, i.e. for any possible output $O = (\mu_t, \alpha_t)_{t=1}^{T}$ by $\mathcal{A}$, and any neighboring databases $D, D' \in \mathcal{D}_n$, $\Pr[O|D] \leq e^{\epsilon} \Pr[O|D']$.

2) $\mathcal{A}$ provides continuous observations with valid confidence intervals where valid means (i) bounded and (ii) improving accuracy, i.e., with high probability p, for

$$\Pr[\mu_t \in [Q(D) - \alpha_t, Q(D) + \alpha_t]] \geq p$$

and

$$\alpha_{t+1} \leq \alpha_t.$$

3) $\mathcal{A}$ provides a well-defined trend line. Before running $\mathcal{A}$, we can provide a trend line of $\mathcal{A}$ to user, i.e. the sequence of the alphas $(\alpha_1, \alpha_2, \ldots)$.

4) (Optional) One of the estimate will reach a desired accuracy requirement for early stoppage, i.e., $\exists t \in [1, T]$, such that $\alpha_t \leq \alpha^*$.

If there are multiple ϵ-differentially private online aggregation algorithms $\{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$, we would like to choose the best algorithm for the data analyst either by (i) providing the

trend line of each algorithm to the data analyst; or (ii) optimizing a utility function over the sequence of confidence intervals provided by the data analyst.

A utility function we may use is the the sum of weighted confidence interval lengths which we shall refer to as score. Let the weights $\{w_t\}_{t=1}^{T}$ be a set of increasing positive numbers. Let

$$score(\mathcal{A}) := \sum_{i=1}^{T} w_t \cdot L_t$$

where $L_t = 2\alpha_t$ denotes the length of the confidence interval $[\mu_t \pm \alpha_t]$, generated by $\mathcal{A}$ at time step t. Given a set of ϵ-DP mechanisms $\{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$, we wish to chose the mechanism that minimizes the score.

There are a couple of reasons that the sum of the weighted confidence interval lengths is the preferred metric. Firstly, smaller confidence interval lengths imply the estimate and true value are close so algorithms that have a smaller score will generally produce estimates close to the true query value with respect to common distance metrics. Secondly, using metrics related to the distance between the true value and the estimate is data-dependent and hence not private.

Thirdly, this metric is preferred as the confidence interval lengths can often be computed exactly beforehand and in the case of WHERE conditions, can at least be approximated with assumptions on the count. This allows us to present trend lines to the user beforehand or after taking a small initial sample. Lastly, the wights are increasing which reflects the need for confidence intervals to improve over time. Since the user can choose the weights, they can emphasize which time steps they wish to prioritize giving them some control over the privacy, time, and accuracy trade-offs of private online aggregation. Choice of weight functions will be discussed more in the evaluation (section 7.4.1).

Some challenges and opportunities for DP online aggregation algorithm design are as follows. The first challenge is related to online aggregation itself as it requires randomization of the data so that concentration bounds can be applied. In all the algorithms introduced in Chapter 5, we assume the data has been pre-shuffled ahead of time. This is a limitation, but at the same time, it allows for better accuracy. Secondly, as there are many noisy estimates released, compounding these estimates to get a more accurate estimate and tighter bounds is important. Hence, an interest is to get better concentration bounds. Another challenge is utilizing privacy amplification to reduce the noise added in the earlier stages of the algorithms. Specifically, when sampling with replacement, if the sample size is sufficiently small, one may use a larger privacy budget and still get a small privacy loss. These optimizations will be addressed in Chapter 6.

Chapter 5

DP Algorithm Design

In this section, we outline the various DP mechanisms for AVG, SUM, and COUNT queries. The mechanism of the following section takes in as input a database D of size n, privacy budget ϵ, confidence parameter p, block size B and total number of time steps T. Where T can be specified by the user or can be simply set to the maximum possible number of time-steps $\lceil \frac{n}{B} \rceil$. An interval $[a, b]$ is also specified for where the data lies. To start, we look at AVG queries with no WHERE conditions in section 5.1. This is the simplest query to make private as each element is counted towards the query so the only thing that needs to be made private is the sum of values. We develop five different mechanisms and empirically evaluate them in Chapter 7. It is determined through both empirical and theoretical analysis that the proposed Gap mechanisms tend to perform the best overall. In sections 5.2 and 5.3, the Gap mechanism is adapted to work for AVG and COUNT queries with WHERE conditions. Lastly, in section 5.4, it is shown how to combine the mechanisms of 5.2 and 5.3 to handle SUM queries given WHERE conditions.

5.1 Average Query

The first mechanism, Baseline 1, maintains the current sum so far and is made private by adding fresh Laplace noise proportional to $\frac{T(b-a)}{\epsilon}$ each iteration. The second mechanism, Baseline 2, samples a new block each iteration, sums the elements of the block and adds Laplace noise proportional to $\frac{(b-a)}{\epsilon}$. It then accumulates all blocks seen so far and so the error from private noise addition is of magnitude $O(\frac{\sqrt{t}}{\epsilon})$. The next three mechanisms are various Gap mechanisms, which are named because they essentially wait for gaps of time

to accumulate enough samples before a noisy partial sum is created similar to in Baseline 2. Gap sizes can vary and be optimized based on weights provided by the user but powers of two generally work well and are easy to implement. The first Gap mechanism is the Single Gap mechanism, which outputs the noised partial sum over the most recent gap. Next is the Multi Gap mechanism which accumulates all the noisy partial sums from each gap and lastly we have the Hybrid Gap mechanism that essentially interpolates between Single and Multi gap mechanisms. The error from private noise addition for the Gap mechanisms are $O(\frac{1}{\epsilon})$ for single, $O(\frac{\sqrt{\log t}}{\epsilon})$ for multi and $O(\frac{\sqrt{\log s}}{\epsilon})$ for hybrid, where s is the number of gaps used in the hybrid. Note that while the Single Gap mechanism adds the least noise, it discards roughly half the samples causing the sampling error to be worse. Thus the hybrid achieves the best of both worlds by optimizing over both the sampling error and the error from noise addition.

Algorithm 1 Baseline 1 for AVG

INPUT: Randomly permuted database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p
for $t = 1, 2, \ldots, T$ **do**
  $c_t = \sum_{i=1}^{t} |B_i|$
  $s_t = \sum_{i=1}^{t} \sum_{j \in B_i} x_j$
  $\tilde{\mu}_t = s_t/c_t + Lap\left(\frac{T(b-a)}{\epsilon c_t}\right)$
  $\tilde{\alpha}_t = \min_{\lambda \in (0,1)} \left((b-a)\sqrt{\frac{1}{2c_t} \ln\left(\frac{2}{\lambda(1-p)}\right)} + \frac{T(b-a)}{\epsilon c_t} \ln(\frac{1}{(1-\lambda)(1-p)}) \right)$
  return interval $[\tilde{\mu}_t - \tilde{\alpha}_t, \tilde{\mu}_t + \tilde{\alpha}_t]$
end for

Theorem 7. *Algorithm 1 satisfies ϵ-differential privacy.*

Proof. Each $\tilde{\mu}_i$ has global sensitivity $\frac{b-a}{c_t}$ so by Laplace Mechanism, satisfies $\frac{\epsilon}{T}$-DP. By sequential composition of DP, the overall privacy loss is ϵ. □

Theorem 8. *Algorithm 1 outputs a valid confidence interval.*

Proof. By Hoeffding's inequality we have that

$$\Pr[|\bar{\mu}_t - \mu| \geq \beta_1] \leq 2 \exp\left(-\frac{2c_t\beta_1^2}{(b-a)^2}\right) = \lambda(1-p).$$

By Laplace tail bound, the noisy estimate satisfies

$$\Pr[|\tilde{\mu}_t - \bar{\mu}_t| \geq \beta_2] = \Pr\left[\left|Lap\left(\frac{T(b-a)}{c_t\epsilon}\right)\right| \geq \beta_2\right]$$

$$\leq \exp\left(-\frac{\epsilon\beta_2 c_t}{T(b-a)}\right) = (1-\lambda)(1-p).$$

By union bound and triangle inequality, we get that

$$\Pr[|\tilde{\mu}_t - \mu| \leq \beta_1 + \beta_2] \geq 1 - \lambda(1-p) - (1-\lambda)(1-p) = p$$

where

$$\beta_1 = \sqrt{\frac{(b-a)^2}{2c_t}\ln\left(\frac{2}{\lambda(1-p)}\right)}$$

and

$$\beta_2 = \frac{T(b-a)}{\epsilon c_t}\ln\left(\frac{1}{(1-\lambda)(1-p)}\right).$$

Minimizing $\beta_1 + \beta_2$ over $\lambda \in (0,1)$ gives us the smallest possible α_t with the desired confidence parameter p.

It is clear that the $\tilde{\alpha}_t$ is decreasing and so the confidence intervals are valid. □

Remark. *Note that all algorithms of this section provide trendlines as the alpha values for the confidence intervals, can be generated ahead of time as they are not data-dependent.*

Another simple approach is to make private each block sum and this would lead to $\Theta(\sqrt{t})$ replacing T in the error term. We spell out the details in the Baseline 2 algorithm (algorithm 2).

Theorem 9. *Algorithm 2 satisfies ϵ-Differential Privacy.*

Proof. Each $\tilde{b}_t$ is generated by sampling with replacement from the remaining $m-(t-1)$ blocks. By the Laplace mechanism, adding noise proportional to $\frac{b-a}{\epsilon}$ to each block provides ϵ-DP. Since the blocks are disjoint and partition D, by parallel composition, the overall privacy loss is ϵ.

□

Theorem 10. *Algorithm 2 outputs a valid confidence interval.*

Algorithm 2 Baseline 2 for AVG

1: **INPUT**: Randomly permuted database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p
2: **for** $t = 1, 2, \ldots, T$ **do**
3: $\quad c_t = \sum_{i=1}^{t} |B_i|$
4: $\quad \tilde{b}_t = \sum_{j \in B_t} x_j + Lap\left(\frac{(b-a)}{\epsilon}\right)$
5: $\quad \tilde{\mu}_t = \sum_{i=1}^{t} \tilde{b}_i / c_t$
6: $\quad \alpha_t = \min_{\lambda \in (0,1)} \left((b-a)\sqrt{\frac{1}{2c_t} \ln(\frac{2}{\lambda(1-p)})} + \frac{(b-a)\sqrt{8t}}{\epsilon c_t} \ln(\frac{2}{(1-\lambda)(1-p)})\right)$
7: $\quad$ **return** interval $[\tilde{\mu}_t - \alpha_t, \tilde{\mu}_t + \alpha_t]$
8: **end for**

Proof. Let $\bigcup_{i=1}^{t} B_i = \{x_1, ..., x_{c_t}\}$ be the randomly sampled elements, $X = \frac{\sum_{i=1}^{c_t} x_i}{c_t}$ and $Y_i \sim Lap(\frac{b-a}{\epsilon})$ be the Laplace random variables from line 4 with respect to the time step i. Observe that

$$|\tilde{\mu}_t - \mu| = \left| X + \frac{\sum_{i=1}^{t} Y_i}{c_t} - \mathbb{E}[X] \right| \leq |X - \mathbb{E}[X]| + \frac{1}{c_t} \left| \sum_{i=1}^{t} Y_i \right| \tag{5.1}$$

$$\leq (b-a)\sqrt{\frac{1}{2c_t} \ln(\frac{2}{\lambda(1-p)})} + \frac{(b-a)\sqrt{8t}}{\epsilon c_t} \ln(\frac{2}{(1-\lambda)(1-p)}) \tag{5.2}$$

where the first inequality follows by triangle inequality. The second follows from sums of Laplace concentration, Hoeffding and a union bound as seen in the proof of the theorem 8. Minimizing over $\lambda \in (0, 1)$ gives us the smallest possible α_t with the desired confidence parameter p.

It is clear that the $\tilde{\alpha}_t$ is decreasing and so the confidence intervals are valid. □

The rest of this subsection focuses on the Gap mechanisms. While the Gap strategy and proofs of privacy are similar, the accuracy proofs for Single Gap relies on the Laplace Tail 1 while the Multi Gap relies on the corollary of Sums of Laplace concentration bound 2. We omit the proofs for the Hybrid Gap Mechanism as it is essentially the same as the proofs of the other Single and Multi Gap Mechanisms.

Theorem 11. *Algorithm 3 satisfies ϵ-Differential Privacy.*

Proof. By the Laplace mechanism, adding noise proportional to $\frac{b-a}{\epsilon}$ to each partial sums provides ϵ-DP. Since the blocks are disjoint and partition D, by parallel composition, the overall privacy loss is ϵ.

Algorithm 3 Single Gap Mechanism

1: **INPUT**: Randomly permuted database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p
2: **for** $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
3: $\quad c_t = \sum_{i=\frac{t}{2}+1}^{t} |B_i|$
4: $\quad s_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} x_j$
5: $\quad \tilde{\mu_t} = s_t/c_t + Lap\left(\frac{b-a}{\epsilon c_t}\right)$
6: $\quad \tilde{\alpha_t} = \min_{\lambda \in (0,1)} \left((b-a)\sqrt{\frac{1}{2c_t} \ln\left(\frac{2}{\lambda(1-p)}\right)} + \frac{(b-a)}{\epsilon c_t} \ln(\frac{1}{(1-\lambda)(1-p)}) \right)$
7: $\quad$ **return** interval $[\tilde{\mu_t} - \tilde{\alpha_t}, \tilde{\mu_t} + \tilde{\alpha_t}]$
8: **end for**

□

Theorem 12. *Algorithm 3 outputs a valid confidence interval.*

Proof. At time step t, let $\bigcup_{i=\frac{t}{2}+1}^{t} B_i = \{x_1, ..., x_{c_t}\}$ be the randomly sampled elements, $X = \frac{\sum_{i=1}^{c_t} x_i}{c_t}$ and $Y \sim Lap(\frac{b-a}{\epsilon c_t})$ be the Laplace random variables from line 5 with respect to the time step. Observe that

$$|\tilde{\mu_t} - \mu| = |X + Y - \mathbb{E}[X]| \leq |X - \mathbb{E}[X]| + |Y| \tag{5.3}$$

$$\leq (b-a)\sqrt{\frac{1}{2c_t} \ln\left(\frac{2}{\lambda(1-p)}\right)} + \frac{(b-a)}{\epsilon c_t} \ln(\frac{1}{(1-\lambda)(1-p)}) \tag{5.4}$$

where the first inequality follows by triangle inequality. The second follows from sums of Laplace concentration, Hoeffding and a union bound as seen in the proof of the theorem 8. Minimizing over $\lambda \in (0, 1)$ gives us the smallest possible α_t with the desired confidence parameter p.

It is clear that the $\tilde{\alpha_t}$ is decreasing and so the confidence intervals are valid. □

The Multi Gap mechanism is similar but rather than discarding the previous gaps, it now combines them with the new gap, improving on the error from sampling while slightly increasing the error from Laplace noise as there are now several noisy sums being aggregated together. This is especially beneficial when it is known that the privacy parameter ϵ is large.

Algorithm 4 Multi Gap Mechanism

1: **INPUT**: Randomly permuted database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p
2: **for** $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
3: $\quad c_t = \sum_{i=1}^{t} |B_i|$
4: $\quad s_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} x_j$
5: $\quad \tilde{s}_t = s_t + Lap\left(\frac{b-a}{\epsilon}\right)$
6: $\quad \tilde{\mu}_t = \sum_{i=1}^{t} \tilde{s}_t / c_t$
7: $\quad \alpha_t = \min_{\lambda \in (0,1)} \left((b-a)\sqrt{\frac{1}{2c_t} \ln(\frac{2}{\lambda(1-p)})} + \frac{(b-a)\sqrt{8(\log_2 t+1)}}{\epsilon c_t} \ln(\frac{2}{(1-\lambda)(1-p)}) \right)$
8: $\quad$ **return** interval $[\tilde{\mu}_t - \alpha_t, \tilde{\mu}_t + \alpha_t]$
9: **end for**

Theorem 13. *Algorithm 4 satisfies ϵ-Differential Privacy.*

Proof. By the Laplace mechanism, adding noise proportional to $\frac{b-a}{\epsilon}$ to each partial sum provides ϵ-DP. Since the blocks are disjoint and partition D, by parallel composition, the overall privacy loss is ϵ.

□

Theorem 14. *Algorithm 4 outputs a valid confidence interval.*

Let $\bigcup_{i=1}^{t} B_i = \{x_1, ..., x_{c_t}\}$ be the randomly sampled elements, $X = \frac{\sum_{i=1}^{c_t} x_i}{c_t}$ and $Y_i \sim Lap(\frac{b-a}{\epsilon})$ be the Laplace random variables from line 5. Note that there are $\log_2 t + 1$ such Laplace variables and observe that

Proof.

$$|\tilde{\mu}_t - \mu| = \left| X + \frac{\sum_{i=1}^{t} Y_i}{c_t} - \mathbb{E}[X] \right| \leq |X - \mathbb{E}[X]| + \frac{1}{c_t} \left| \sum_{i=1}^{\log_2 t+1} Y_i \right| \tag{5.5}$$

$$\leq (b-a)\sqrt{\frac{1}{2c_t} \ln(\frac{2}{\lambda(1-p)})} + \frac{(b-a)\sqrt{8(\log_2 t+1)}}{\epsilon c_t} \ln(\frac{2}{(1-\lambda)(1-p)}) \tag{5.6}$$

where the first inequality follows by triangle inequality. The second follows from sums of Laplace concentration, hoeffding and a union bound as seen in the proof of the theorem 8. Minimizing over $\lambda \in (0, 1)$ gives us the smallest possible α_t with the desired confidence parameter p. It is clear that the $\tilde{\alpha_t}$ is decreasing and so the confidence intervals are valid. □

Based on the parameter ranges, the user may desire the Gap mechanism using noisy sums over the Gap mechanism using a single sum. In particular, when the error from noise addition is much smaller than the error from sampling, using many sums is preferred whereas using a single sum benefits the cases where the Laplace noise is overpowering. The following mechanism is a happy medium between the two and works by taking only the noisy sums that minimize the private confidence interval length. The proofs of correctness and privacy essentially follow from that of the previous two mechanisms. Note that we can implement line 5 of Algorithm 5 by iterating over all pairs of start and end indices (s, f) and output the smallest confidence interval.

Algorithm 5 Hybrid Gap Mechanism

1: **INPUT**: Randomly permuted database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p
2: **for** $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
3: $\quad c_t = \sum_{i=1}^{t} |B_i|$
4: $\quad s_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} x_j$
5: $\quad (s, f) = sumOptimizer(t, \tilde{c}_t, n, p)$
6: $\quad \tilde{s}_t = s_t + Lap\left(\frac{b-a}{\epsilon}\right)$
7: $\quad \tilde{\mu}_t = \sum_{i=1}^{t} \tilde{s}_t / c_t$
8: $\quad$ **if** $s == t$ **then**
9: $\quad\quad \alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{(1 - \frac{k-1}{n}) \frac{1}{2k} \ln(\frac{2}{\lambda(1-p)})} + \ln(\frac{1}{(1-\lambda)(1-p)}) \right)$
10: $\quad$ **else**
11: $\quad\quad \alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{(1 - \frac{k-1}{n}) \frac{1}{2k} \ln(\frac{2}{\lambda(1-p)})} + \frac{\sqrt{8(f-s+1)}}{\epsilon k} \ln(\frac{2}{(1-\lambda)(1-p)}) \right)$
12: $\quad$ **end if**
13: $\quad$ **return** interval $[\tilde{\mu}_t - \alpha_t, \tilde{\mu}_t + \alpha_t]$
14: **end for**

5.2 Average Query with Where Condition

In this section we adapt the Gap mechanism for average queries with no condition to work with general WHERE conditions. Extra care must be taken when releasing an aggregate and confidence interval as the denominator, which is the counts so far, also needs to be made private. Further, we provide a practical and effective budget optimization strategy that splits the privacy budget for the noisy sum and noisy count in a more optimal way. Our experiments in Chapter 7 provide some evidence for when this optimization is useful.

Algorithm 6 Gap Mechanism for AVG with WHERE condition

INPUT: Database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p, query Q
for $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
    $(\epsilon_1, \epsilon_2) = BudgetOptimizer(t, \tilde{c}_{t-1}, \epsilon)$
    $c_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} \mathbf{1}_{x_j \in Q}$
    $s_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} x_j \mathbf{1}_{x_j \in Q}$
    $\tilde{s_t} = s_t + Lap\left(\frac{b-a}{\epsilon_2}\right)$
    $\tilde{c}_t = c_t + Lap\left(\frac{1}{\epsilon_1}\right)$
    $\tilde{c}_{lb} = \tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1$
    $\tilde{c}_{ub} = \tilde{c}_t + \ln(\frac{3}{1-p})/\epsilon_1$
    $\tilde{\mu}_r = \tilde{s_t}/\tilde{c_{lb}}$
    $\tilde{\mu}_l = \tilde{s_t}/\tilde{c_{ub}}$
    $\tilde{\gamma}_t = (b-a)\sqrt{\frac{1}{2\tilde{c_{lb}}} \ln\left(\frac{6}{1-p}\right)} + \frac{(b-a)}{\epsilon_2 \tilde{c_{lb}}} \ln(\frac{3}{1-p})$
    $\tilde{\mu_t} = \frac{\tilde{\mu_l} + \tilde{\mu_r}}{2}$
    $\tilde{\alpha_t} = \frac{\tilde{\mu_r} - \tilde{\mu_l}}{2} + \tilde{\gamma}_t$
    return interval $[\tilde{\mu_t} - \tilde{\alpha_t}, \tilde{\mu_t} + \tilde{\alpha_t}]$
end for

The BudgetOptimizer essentially wishes to solve the following optimization problem each iteration of the Gap mechanism. The main difficulty is that we cannot privately solve such optimization without knowing beforehand what $\tilde{c}_t$ is. However, this can be approximated by solving the optimization problem by using $2\tilde{c_{t-1}}$ in its place. This approach performs well in practice as can be seen in our empirical analysis of this algorithm (figures 7.2, 7.3, and 7.4).

$$\begin{aligned} &\text{minimize} \quad (b-a)\sqrt{\frac{1}{2\tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1}\ln\left(\frac{6}{1-p}\right)} + \frac{(b-a)}{\epsilon_2(\tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1)}\ln(\frac{3}{1-p}) \\ &\text{subject to} \quad \epsilon_1 + \epsilon_2 = \epsilon \end{aligned} \tag{5.7}$$

Theorem 15. *Algorithm 6 outputs a bounded confidence interval.*

Proof. At time step t, let $\bigcup_{i=\frac{t}{2}+1}^{t} B_i = \{x_1, ..., x_{c_t}\}$ be the randomly sampled elements, $X = \frac{\sum_{i=1}^{c_t} x_i}{c_t}$ and $Y \sim Lap(\frac{b-a}{\epsilon_2})$ be the Laplace random variables from line 6 with respect to the time step. Let $\overline{\mu}_t = X + \frac{Y}{c_t}$ Observe that

$$|\overline{\mu}_t - \mu| = \left|X + \frac{Y}{c_t} - \mathbb{E}[X]\right| \leq |X - \mathbb{E}[X]| + \frac{|Y|}{c_t} \tag{5.8}$$

$$\leq (b-a)\sqrt{\frac{1}{2c_t}\ln\left(\frac{6}{1-p}\right)} + \frac{(b-a)}{\epsilon_2 c_t}\ln\left(\frac{3}{1-p}\right) \tag{5.9}$$

$$\leq (b-a)\sqrt{\frac{1}{2(\tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1)}\ln\left(\frac{6}{1-p}\right)} + \frac{(b-a)}{\epsilon_2(\tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1)}\ln\left(\frac{3}{1-p}\right) = \tilde{\gamma}_t \tag{5.10}$$

where the first inequality follows by triangle inequality. The second inequality follows from Hoeffding and Laplace concentration and each hold with probability at least $1 - \frac{1-p}{3}$. Since $\tilde{c}_t = c_t + Lap(\frac{1}{\epsilon_1})$, by the Laplace tail we have

$$\tilde{c}_{lb} = \tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1 \leq c_t \leq \tilde{c}_t + \ln(\frac{3}{1-p})/\epsilon_1 = \tilde{c}_{ub} \tag{5.11}$$

with probability at least $1 - \frac{1-p}{3}$. Union bounding over the three events gives us $|\overline{u}_t - \mu| \leq \tilde{\gamma}_t$ with probability at least p.

By using (5.11) with the fact that $\tilde{\mu}_t = \frac{s_t+Y}{c_t}$, we get that $\tilde{\mu}_l = \frac{s_t+Y}{\tilde{c}_{ub}} \leq \overline{\mu}_t \leq \frac{s_t+Y}{\tilde{c}_{lb}} = \tilde{\mu}_r$. Since $\mu \in [\overline{\mu}_t \pm \tilde{\gamma}_t]$ with probability at least p, we get that $\tilde{\mu}_l - \tilde{\gamma}_t \leq \mu \leq \tilde{\mu}_r + \tilde{\gamma}_t$, i.e. $\mu \in [\tilde{\mu}_l - \tilde{\gamma}_t, \tilde{\mu}_r + \tilde{\gamma}_t]$. By taking $\tilde{\mu}_t = \frac{\tilde{\mu}_l + \tilde{\mu}_r}{2}$ and $\tilde{\alpha}_t = \frac{\tilde{\mu}_r - \tilde{\mu}_l}{2} + \tilde{\gamma}_t$, we equivalently get that $\mu \in [\tilde{\mu}_t \pm \tilde{\alpha}_t]$ with probability at least p as needed. □

Remark. *While $\tilde{\alpha}_t$ may not always be decreasing, we may apply a simple post processing step to ensure the confidence interval is valid. Specifically, we take the smallest $\tilde{\alpha}_t$ and corresponding estimate seen so that $\{\tilde{\alpha}_t\}_t$ are decreasing.*

Theorem 16. *Above Algorithm satisfies ϵ-DP.*

Proof. Note at each iteration t, we compute a noisy count $\tilde{c}_t = c_t + Lap(\frac{1}{\epsilon_1})$ and also a noisy sum $s_t + Lap(\frac{b-a}{\epsilon_2})$. Since c_t has sensitivity 1 and s_t has sensitivity $(b-a)$, by Laplace mechanism and the guarantee of the budget optimizer, each iteration of the algorithm satisfies $\epsilon_1 + \epsilon_2 = \epsilon$-DP. Since the database is partitioned in to disjoint blocks, and for each t, s_t, c_t do not reuse blocks, by parallel composition, the total privacy loss is still ϵ. □

Remark. *Note that trend lines cannot be computed exactly for this algorithm as the alpha values in the confidence intervals are dependent on the count of database elements satisfying the WHERE condition. However, we can approximate them by processing a small sample to estimate the density, then building trend lines based on the approximate density so that the user may chose which algorithm to run. In this section only the Gap mechanism was provided but all algorithms from section 5.1 may be adapted to handle WHERE conditions in a similar manner to algorithm 6.*

5.3 Count Query

Since the size of our database $|D| = n$ is known, it is possible to estimate the counts of elements in D that satisfy a given WHERE condition. Let Q be a counting query with any WHERE condition and let $\mathbf{1}_{x \in Q}$ be the indicator for when x satisfies the condition in Q. Since we are essentially sampling with replacement, the corresponding distribution is hypergeometric and so we may use the hypergeometric tail bounds or related approximations to derive confidence intervals. We adapt the Gap mechanism of the previous section to handle COUNT queries with arbitrary WHERE conditions.

Note that we may implement *sumOptimizer* on line 5 of Algorithm 7 by trying all pairs of start and end indices and finding the pair that will minimize the confidence interval length.

Theorem 17. *Algorithm 7 satisfies ϵ-DP.*

Proof. Since the global sensitivity of a counting query is 1, the proof follows by the same Laplace mechanism and parallel composition arguments from before. □

Algorithm 7 Gap Mechanism for COUNT

INPUT: Database $D = (B_1, ..., B_m)$ of size n, block size B, privacy budget ϵ, time step T, confidence parameter p, query Q
for $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
  $b_t = \sum_{\frac{t}{2}+1}^{t} |B_i|$
  $\tilde{c}_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{x_j \in B_i} \mathbf{1}_{x_j \in Q} + Lap\left(\frac{1}{\epsilon}\right)$
  $(s, f) = sumOptimizer(t, \tilde{c}_t, n, p)$
  $k = \sum_{s}^{f} b_t$
  $\tilde{C}_t = n \sum_{i=s}^{f} \tilde{c}_t / k$
  if $s == t$ **then**
    $\alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{\frac{1}{2k} \ln(\frac{2}{\lambda(1-p)})} + \ln(\frac{1}{(1-\lambda)(1-p)}) \right)$
  else
    $\alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{\frac{1}{2k} \ln(\frac{2}{\lambda(1-p)})} + \frac{\sqrt{8(f-s+1)}}{\epsilon k} \ln(\frac{2}{(1-\lambda)(1-p)}) \right)$
  end if
  return interval $[\tilde{C}_t - n \cdot \alpha_t, \tilde{C}_t + n \cdot \alpha_t]$
end for

Lemma 3. *Given a query Q, let N_Q be the total number of element in D satisfying the conditions of Q. Let $X_1, ..., X_k$ be randomly sampled without replacement from a database D of size n. Let N_i be the indicator for when X_i satisfies the conditions of Q, then*

$$N_Q \in \left[\frac{n \sum_{i=1}^{k} N_i}{k} \pm n \sqrt{\frac{\ln\left(\frac{2}{\delta}\right)}{2k}} \right]$$

with probability at least $1 - \delta$.

Proof. Observe that $\mathbb{E}[\frac{\sum_{i=1}^{k} N_i}{k}] = \frac{N_Q}{n}$. Thus by the hoeffding inequality, we get that

$$\Pr\left[\left| \frac{N_Q}{n} - \frac{\sum_{i=1}^{k} N_i}{k} \right| \geq t \right] \leq 2 \exp\left(-2kt^2\right) = \delta \tag{5.12}$$

where $t = \sqrt{\frac{\ln\left(\frac{2}{\delta}\right)}{2k}}$, which gives us the desired bounds on N_Q. □

Theorem 18. *Algorithm 5 outputs a valid confidence interval*

Proof. Follows from lemma 3, the Laplace concentration and tail bound just like in section 1 except with $(b - a) = 1$ and by multiplying by n to get the counts rather than the ratio of elements satisfying Q. □

5.4 Sum Query

A simple approach for deriving private confidence intervals for sum using our current framework is the following. Given a $\epsilon/2$-DP mechanism that outputs a $(1 - \frac{1-p}{2})$-CI, $[\mu_l, \mu_r]$ for the AVG and a $\epsilon/2$-DP mechanism that outputs a $(1 - \frac{1-p}{2})$-CI $[C_l, C_r]$, for COUNT, output $[\mu_l \cdot C_l, \mu_r \cdot C_r]$. By union bound, this is a p-confidence interval for sum and satisfies ϵ-DP by sequential composition.

By carefully analyzing the Gap mechanisms for both AVG and COUNT, you can see that there is some overlap as both use the noisy current count $\tilde{c}_t = c_t + Lap(\frac{3}{\epsilon_1})$. Thus, SUM queries can be answered with no extra overhead in privacy loss when AVG and COUNT are simultaneously being estimated. The algorithm is provided below without proof as the proofs are essentially the same as the ones in the previous sections.

Algorithm 8 Gap Mechanism for SUM with WHERE condition

INPUT: Database $D = (B_1, ..., B_m)$, block size B, privacy budget ϵ, time step T, confidence parameter p, query Q
for $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
  $b_t = \sum_{\frac{t}{2}+1}^{t} |B_i|$
  $(\epsilon_1, \epsilon_2) = BudgetOptimizer(t, \tilde{c}_{t-1}, \epsilon)$
  $c_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} \mathbf{1}_{x_j \in Q}$
  $s_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{j \in B_i} x_j \mathbf{1}_{x_j \in Q}$
  $\tilde{s_t} = s_t + Lap\left(\frac{b-a}{\epsilon_2}\right)$
  $\tilde{c}_t = c_t + Lap\left(\frac{1}{\epsilon_1}\right)$
  $\tilde{c}_{lb} = \tilde{c}_t - \ln(\frac{3}{1-p})/\epsilon_1$
  $\tilde{c}_{ub} = \tilde{c}_t + \ln(\frac{3}{1-p})/\epsilon_1$
  $k = \sum_{\frac{t}{2}+1}^{t} b_t$
  $\tilde{C}_t = n \sum_{\frac{t}{2}+1}^{t} \tilde{c}_t / k$
  $\tilde{\mu}_r = \tilde{s_t}/\tilde{c_{lb}}$
  $\tilde{\mu}_l = \tilde{s_t}/\tilde{c_{ub}}$
  $\tilde{\alpha_t} = (b-a)\sqrt{\frac{1}{2\tilde{c_{lb}}} \ln\left(\frac{6}{1-p}\right)} + \frac{(b-a)}{\epsilon_2 \tilde{c_{lb}}} \ln(\frac{3}{1-p})$
  $\gamma_t = \sqrt{\frac{1}{2k} \ln(\frac{6}{1-p})} + \ln(\frac{2}{1-p})$
  return interval $[(\tilde{\mu_l} - \tilde{\alpha_t}) \cdot (C_t - n\gamma_t), (\tilde{\mu_r} + \tilde{\alpha_t}) \cdot (C_t + n\gamma_t)]$
end for

Remark. *Let $[l, r]$ be the interval returned by Algorithm 8 on line 17. Note that it can be written in the form $\mu_t \pm \alpha_t$ by taking $\mu_t = \frac{l+r}{2}$ and taking $\alpha_t = \frac{r-l}{2}$.*

Chapter 6

Optimizations

6.1 Privacy Amplification

The following lemma of Ullman [30] and Borja et al. [2] state that when a private algorithm is run on a subsample of the database obtained by sampling without replacement, in many cases the privacy loss is much smaller than what a simple analysis would indicate. Since a linear scan of our randomly permuted database is equivalent to sampling without replacement, we may apply the bound to our mechanisms developed in the previous sections.

Lemma 4 (Privacy Amplification)**.** *Let $m \leq n$. If algorithm $\mathcal{A} : \mathcal{D}^m \to \mathcal{R}$ satisfies ϵ-DP then given a database $D \in \mathcal{D}^n$ and a random m-element sub-sample D' of the database. $\mathcal{A}' := \mathcal{A}(D')$ satisfies $\frac{(e^\epsilon - 1)m}{n}$-Differential Privacy.*

The privacy amplification bound for sampling without replacement may be used to improve the error from Laplace noise especially in the early stages of the algorithms of Chapter 5. To demonstrate this, consider the basic Gap mechanism for AVG (Algorithm 3). Let $(O_1, ...O_T)$ be the output sequence of algorithm 3 where each output i satisfies ϵ_i-DP. By Privacy Amplification (lemma 4) and sequential composition of DP, the first t outputs actually have a total privacy loss of

$$\frac{(e^{\sum_{i=1}^{t} \epsilon_i} - 1) \sum_{i=1}^{t} |B| 2^i}{n} = \frac{|B|(e^{\sum_{i=1}^{t} \epsilon_i} - 1)(2^t - 1)}{n}. \tag{6.1}$$

From this, we can pick an appropriate set of ϵ_i such that the above (6.1) is at most ϵ. For the remaining values $t + 1 \leq i \leq T$ we set $\epsilon_i = \epsilon$ which follows from the previous

parallel composition arguments from the privacy proof of Algorithm 3. A simple approach is to set $\epsilon_i = \epsilon'$ for each $i \leq t$ and then take $\epsilon' = \frac{1}{t}\ln(\frac{n\epsilon}{|B|(2^t-1)})$. Picking the correct t value will depend on the user's preference for accuracy. Due to the sequential composition and sample size, picking a larger t will lead to less overall amplification. Future work will leverage the user-specified weight functions to pick the optimal t.

Example: When $|B| = 100$, $T = 29100$, $n = 2910000$ and $\epsilon = 0.01$, by setting $t = 5$ we may set $\epsilon' = 0.468$ and if $t = 10$ we may set it to 0.025 which are both improvements from the default ϵ. We illustrate the improvements empirically for the $t = 5$ case in Chapter 7 (figure 7.7).

6.2 Improved Laplace Concentration Bound

The proof of the concentration bound on sums of independent Laplace distributons [8] uses the classic Chernoff method but uses the slightly loose bound $\frac{1}{(1-x)} \leq e^{2x}$ when $0 < x < \frac{1}{2}$. This bound is actually true for a larger range of values, namely up to $x \leq 0.7968$. Which allows us to improve the $0 < \lambda < \frac{2\sqrt{2}\nu^2}{b_{\max}}$ to $0 < \lambda \leq \frac{3.57\nu^2}{b_{\max}}$ in Theorem 19. Alternatively, we may use the bound $\frac{1}{(1-x)} \leq e^{1.4x}$ when $0 < x \leq \frac{1}{2}$ and get everything else the same except $\Pr[|Y| > \lambda] \leq \exp\left(-\frac{\lambda^2}{8\nu^2}\right)$ is now improved to $\Pr[|Y| > \lambda] \leq \exp\left(-\frac{1.3\lambda^2}{8\nu^2}\right)$.

Theorem 19. *Let $Y = \sum_i Y_i$ where each $Y_i \sim Lap(b_i)$ are independent. Let $b_{\max} = \max_i b_i$, $\nu \geq \sqrt{\sum_i b_i^2}$ and $0 < \lambda < \frac{2\sqrt{2}\nu^2}{b_{\max}}$ suppose $0 < \delta < 1$, then*

$$\Pr[|Y| > \lambda] \leq \exp\left(-\frac{\lambda^2}{8\nu^2}\right).$$

Being more general we derive the following theorem and corollary which can lead to more optimal constants.

Theorem 20. *Given $a > 1$ and bound $\frac{1}{1-x} \leq e^{ax}$ which holds when $x \leq x^* < 1$. Let $Y = \sum_i Y_i$ where each $Y_i \sim Lap(b_i)$ are independent. Let $b_{\max} = \max_i b_i$, $\nu \geq \sqrt{\sum_i b_i^2}$ and $0 < \lambda \leq \frac{\sqrt{x^*}2a\nu^2}{b_{\max}}$ suppose $0 < \delta < 1$, then*

$$\Pr[|Y| > \lambda] \leq \exp\left(-\frac{\lambda^2}{4a\nu^2}\right).$$

Proof. For each Y_i, the moment generating function is $\mathbb{E}[\exp(hY_i)] = \frac{1}{1-h^2b_i^2}$, where $|h| < \frac{1}{b_i}$. Using the inequality $\frac{1}{1-x} \leq e^{ax}$ for $x \leq x^*$, we have that $\mathbb{E}[\exp(hY_i)] \leq \exp(ah^2b_i^2)$, if $|h| \leq \frac{\sqrt{x^*}}{b_i}$. By taking $0 < h \leq \frac{\sqrt{x^*}}{b_{\max}}$, we have

$$\begin{aligned}
\Pr[Y > \lambda] &= \Pr[\exp(hY) > \exp(h\lambda)] && (6.2)\\
&\leq \exp(-h\lambda)\,\mathbb{E}[\exp(hY)] && (6.3)\\
&= \exp(-h\lambda)\prod_i \mathbb{E}[\exp(hY_i)] && (6.4)\\
&= \exp(-h\lambda + ah^2\nu^2) && (6.5)\\
&\leq \exp\left(-\frac{(2a-a)\lambda^2}{(2a)^2\nu^2}\right) && (6.6)\\
&= \exp\left(-\frac{\lambda^2}{4a\nu^2}\right) && (6.7)
\end{aligned}$$

where the first inequality follows by Markov inequality and the last follows from the assumption of $0 < \lambda \leq \frac{\sqrt{x^*}2a\nu^2}{b_{\max}}$ by setting $h := \frac{\lambda}{2a\nu^2} \leq \frac{\sqrt{x^*}}{b_{\max}}$. □

Remark. *We can recover the original bound (Theorem 7) by taking $a = 2$ and $x^* = \frac{1}{2}$.*

Corollary 3. *Let Y, ν, $\{b_i\}_i$, $b_{\max}$, a and x^* be defined as in the previous theorem. Suppose $0 < \delta < 1$ and $\nu \geq \max\{\sqrt{\sum_i b_i^2}, b_{\max}\sqrt{\frac{1}{x^*a}\ln(\frac{2}{\delta})}\}$. Then, $\Pr[|Y| > 2\nu\sqrt{a\ln(\frac{2}{\delta})}] \leq \delta$.*

Proof. By the symmetry of the Laplace distribution, union bound and the above theorem,

$$\Pr\left[|Y| > \lambda\right] \leq 2\exp\left(-\frac{\lambda^2}{4a\nu^2}\right) = \delta$$

by taking $\lambda = 2\nu\sqrt{a\ln(\frac{2}{\delta})} \leq \frac{\sqrt{x^*}2a\nu^2}{b_{\max}}$ which holds as $\nu \geq b_{\max}\sqrt{\frac{1}{x^*a}\ln(\frac{2}{\delta})}$ by assumption. □

To see how to use the corollary, for simplicity suppose $b_i = b$, then for a given (a, x^*) we take $\nu \geq \max\{b\sqrt{k}, b\sqrt{\frac{1}{x^*a}\ln(\frac{2}{\delta})}\}$. Notice that when k is sufficiently large, i.e., $k > \frac{1}{x^*a}\ln(\frac{2}{\delta})$, taking a smaller a leads to a smaller λ value. Thus we can obtain tighter concentration bounds by minimizing λ over the set $\{(x^*, a) : \frac{1}{1-x} \leq e^{ax}$ when $0 < x \leq x^*\}$.

6.3 Improved Concentration Bounds for Sampling With Replacement

As an alternative to using Hoeffding and lemma 2, one can directly use the Hoeffding-Serfling bound for sampling without replacement and even obtain some improvements.

Theorem 21 (Hoeffding-Serfling)**.** *Let $\mathcal{D} = x_1, ..., x_N$ be a set of N values in $[a, b]$. Let $X_1, ..., X_n$ denote a random sample without replacement from $\mathcal{D}$. Then, for any $t > 0$, we have*

$$\Pr\left[\max_{n \le k \le N-1} \frac{\sum_{i=1}^{k}(X_i - \mathbb{E}\, X_i)}{k} \ge t\right] \le \exp\left(-\frac{2nt^2}{(1 - n/N)(1 + 1/n)(b-a)^2}\right)$$

and

$$\Pr\left[\max_{1 \le k \le n} \frac{\sum_{i=1}^{k}(X_i - \mathbb{E}\, X_i)}{k} \le -t\right] \le \exp\left(-\frac{2nt^2}{(1 - (n-1)/N)(b-a)^2}\right)$$

Hoeffding-Serfling may be used to improve all the algorithms of section 5 and to demonstrate this we provide an improved mechanism for count queries. As before, we implement *sumOptimizer* on line 5 of Algorithm 9 by trying all pairs of start and end indices and finding the pair that will minimize the confidence interval length. The proof of privacy and accuracy are essentially the same but we derive the confidence interval using Theorem 21 instead.

We implement and evaluate algorithm 9 in Chapter 7 (figure 7.8). Our analysis shows that using the Serfling variant provides a significant improvement in the later time steps of the Gap mechanism.

Remark. *Both Hoeffding and Hoeffding-Serfling inequalities result in an error term around the estimates of magnitude $O(\frac{b-a}{\sqrt{m}})$. However, given prior knowledge of the variance or a private procedure to estimate the variance, we may further improve the confidence intervals by using the Bernstein-Serfling bounds from [3]. The resulting error is of magnitude $O(\frac{\sigma}{\sqrt{m}} + \frac{b-a}{m})$ which is a noticeable improvement when the variance bound is small.*

Algorithm 9 Hoeffding Serfling based Gap Mechanism for COUNT

INPUT: Database $D = (B_1, ..., B_m)$ of size n, block size B, privacy budget ϵ, time step T, confidence parameter p, query Q
for $t = 1, 2, 2^2, 2^3 \ldots, T$ **do**
    $b_t = \sum_{\frac{t}{2}+1}^{t} |B_i|$
    $\tilde{c}_t = \sum_{i=\frac{t}{2}+1}^{t} \sum_{x_j \in B_i} \mathbf{1}_{x_j \in Q} + Lap\left(\frac{1}{\epsilon}\right)$
    $(s, f) = sumOptimizer(t, \tilde{c}_t, n, p)$
    $k = \sum_{s}^{f} b_t$
    $\tilde{C}_t = n \sum_{i=s}^{f} \tilde{c}_t / k$
    if $s == t$ **then**
        $\alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{\frac{\left(1 - \frac{k-1}{n}\right)}{2k} \ln(\frac{2}{\lambda(1-p)})} + \ln(\frac{1}{(1-\lambda)(1-p)}) \right)$
    else
        $\alpha_t = \min_{\lambda \in (0,1)} \left(\sqrt{\frac{\left(1 - \frac{k-1}{n}\right)}{2k} \ln(\frac{2}{\lambda(1-p)})} + \frac{\sqrt{8(f-s+1)}}{\epsilon k} \ln(\frac{2}{(1-\lambda)(1-p)}) \right)$
    end if
    return interval $[\tilde{C}_t - n \cdot \alpha_t, \tilde{C}_t + n \cdot \alpha_t]$
end for

Chapter 7

Evaluation

In this chapter, we provide several experiments for the differentially private online aggregation mechanisms developed for AVG, COUNT, and SUM queries. We will discuss the data set and experiment setup, performance metrics, and the results of this evaluation. Our goal is to determine which mechanism produces the smallest confidence intervals and to understand the impact of the parameter ranges on the confidence interval lengths. Our main discovery is that the Gap mechanisms perform the best overall when answering the queries in table 7.1. Furthermore, we demonstrate and explain the various optimizations from Chapters 5 and 6.

7.1 Dataset and Queries

Our experiments were performed on the consumer travel report released by the US Department of Transportation's Bureau of Transportation Statistics [1]. The report contains

```
Q1: SELECT AVG(arr_del15) FROM flights;
Q2: SELECT AVG(arr_del15) FROM flights WHERE airport="PHX" AND month=1;
Q3: SELECT AVG(arr_del15) FROM flights WHERE airport="PHX" ;
Q4: SELECT AVG(arr_del15) FROM flights WHERE month=1;
Q5: SELECT COUNT(arr_del15) FROM flights WHERE airport="PHX";
Q6: SELECT SUM(arr_del15) FROM flights WHERE month=1;
```

Table 7.1: SQL queries for the flight delay data set

WHERE Condition	AVG	COUNT	SUM
none	74.71617392382139	2911301	217521272
airport="PHX" AND month=1	209.35321100917432	2180	456390
airport="PHX"	185.1253698224852	27040	5005790
month=1	76.00113098479454	238730	18143750

Table 7.2: True AVG, COUNT, and SUM values of the SQL queries

information on the number of on-time, delayed, cancelled, and diverted flights operated by larger air carriers. We duplicated the records 10 times to give us roughly 3 million tuples. Furthermore, we add an index column to each row and uniformly randomly shuffle the tuples. Each tuple has 21 attributes but we will focus on"arr_del15" which is the arrival delay,"month", and"airport". For the experiments, we are concerned about aggregating arrival delay of flights and in this dataset, the values lie in the range $[0, 6337]$.

We evaluate our private mechanisms over the queries found in Table 7.1. The first query is over the entire data set but the ones involving WHERE conditions were carefully selected and represent roughly 10%, 1%, and 0.1% of the data satisfying the condition. Specifically, when the WHERE conditions are 'airport="PHX" AND month=1', 'airport="PHX"', and 'month=1', the counts are 2180, 27040, and 238730 respectively. We do this to understand how sparse queries can be while still getting meaningful aggregates.

7.2 Experiment Setup

Our main experiments involve testing the private mechanisms of Chapter 5 over a wide range of parameters. We mostly focus on AVG queries as the algorithms for COUNT and SUM are similar. To get a good understanding of the error from private noise addition, we use privacy budgets $\epsilon = 0.01, 0.1, 1$. Also, the various block sizes $B = 100, 1000, 10000$ are used but the confidence probability is fixed to be $p = 0.95$. For the most part, the total time steps T is chosen based on B so that the entire data set is processed by the end.

For the first set of experiments we will answer SELECT AVG(arr_del15) FROM flights; using Baseline 1, Baseline 2, Single, Multi, and Hybrid Gap mechanisms and the goal is to determine the best overall strategy. The goal of the second set of experiments is to understand the role of budget optimization. We study various AVG queries with WHERE conditions (Q2-Q4 in Table 7.1) and we do so with two variants of the Single Gap mechanism. The first Gap mechanism naively splits the privacy budget in half and the second

Gap mechanism optimizes how we split the budget ϵ to reduce the confidence interval lengths. Next, we look at COUNT and SUM queries both with WHERE conditions and we compare them to the non-private algorithms. Lastly, we take a look at the impact of some optimizations from sections 6.1 and 6.3. Specifically, how Hoeffding-Serfling bound and privacy amplification improve our Gap mechanism.

Implementation: All experiments are done on a computer with specifications of 8GB RAM, Intel Core i5-8250U CPU @ 1.60GHz CPU, and Ubuntu 18.04.2 operating system. The source code for our experiments is written in Python and uses the DuckDB database management system to process SQL queries.

7.3 Metrics

We present a few metrics to test our main hypothesis that the Gap mechanisms perform the best overall. These metrics are also used to test our hypothesis that privacy amplification and budget optimization are effective for improving the Gap mechanism.

CI Lengths/Alpha Values: The main metric we use is to track the alpha values which are essentially half the confidence interval length. For a private mechanism, the alpha values are obtained by combining the sampling error and the error from Laplace noise addition via union bound. The benefit of comparing the alpha values of the mechanisms is that it can be done privately and often can be computed or at least approximated before the query has been run. Furthermore, smaller confidence interval lengths imply small estimation error so this is a better metric overall.

Score: The objective function of Section 4 seeks to minimize the weighted confidence interval lengths given an increasing set of weight functions. We refer to this quantity as the score of a mechanism. We do not directly present results and plots for various score functions except for when $w_t = 1$ for all $t \in [T]$. However, we will discuss and justify which mechanism will perform the best for a wide array of score functions.

Estimation Error: We plot the true value, non-private aggregate values and the private mechanisms aggregate values for certain queries. This allows us to see which mechanism has the smallest error by taking the difference from the true value, and also the effect of private noise addition by comparing the private to the non-private aggregate.

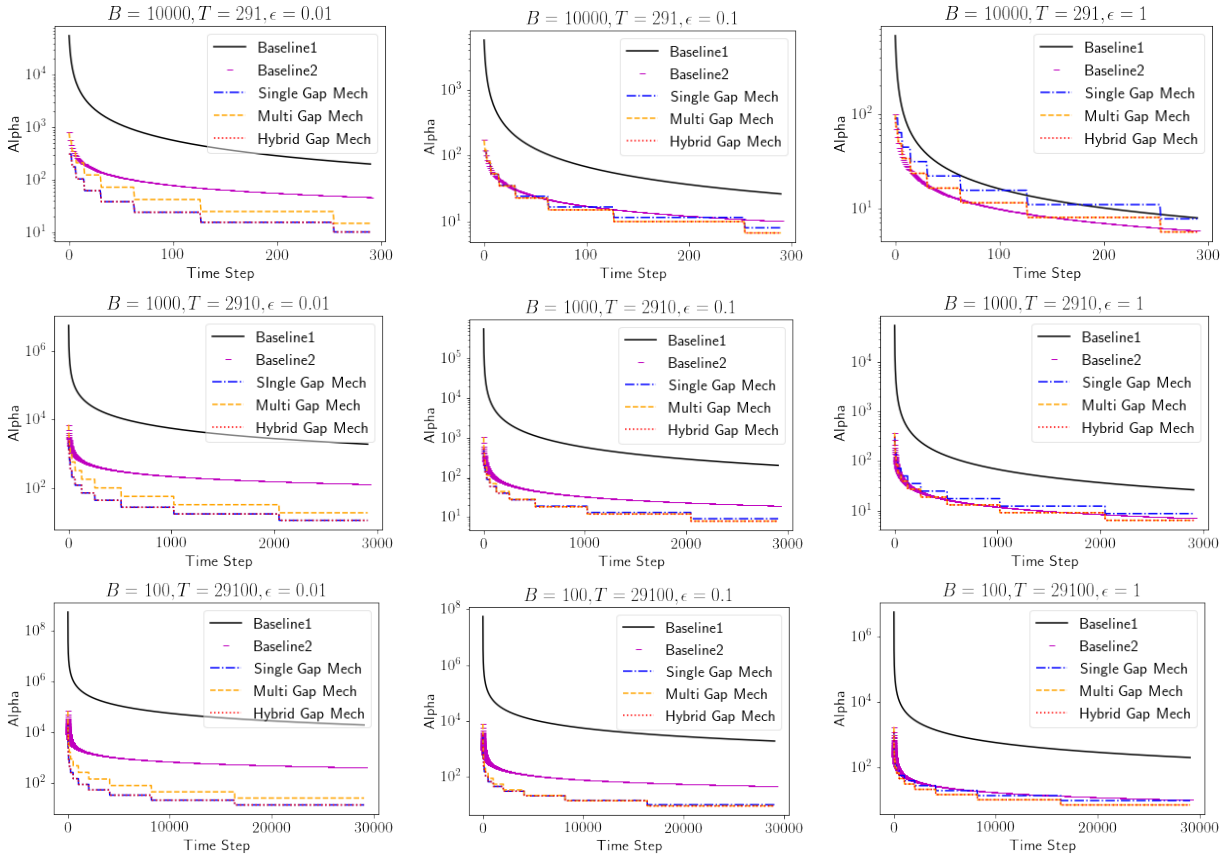

Figure 7.1: AVG Arrival Delays over the entire data set. As ϵ decreases the need for Gap mechanisms is clear.

7.4 Results

In this section, we present the results of our experiments.

7.4.1 Impact of Mechanism Choice

Overall it is evident that Algorithm 5 (Hybrid Gap Mech) outperforms the rest. Essentially it interpolates between Algorithm 3 (Single Gap Mech) and Algorithm 4 (Multi Gap Mech) so will always dominate the two. It is interesting to note that the simple Baseline 2 mechanism is very close to the Gap strategies when $\epsilon = 0.1$ and even outperforms them in

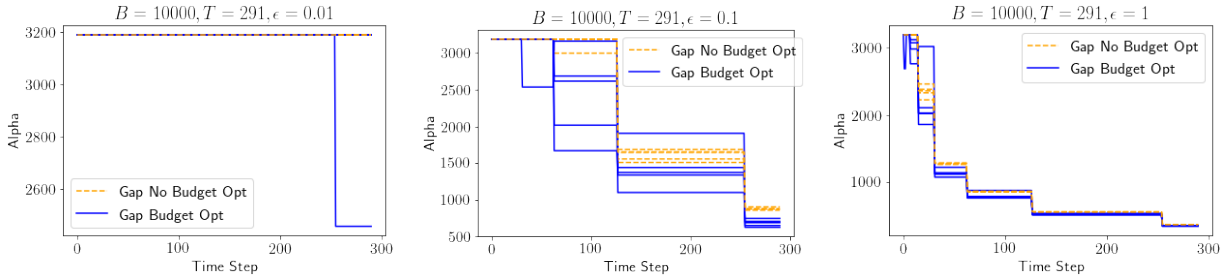

Figure 7.2: Average delay WHERE airport="PHX" and month=1". The query is sparse and so budget optimization is crucial.

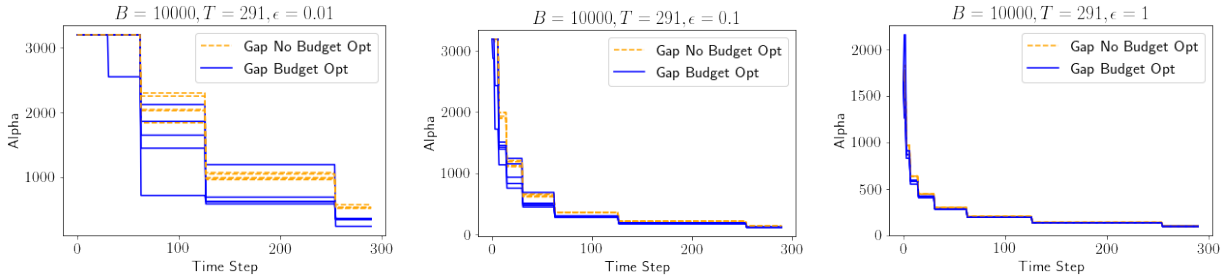

Figure 7.3: Average delay WHERE airport="PHX". Except when $\epsilon = 0.01$, budget optimization is not crucial for queries that make up $\sim 1\%$ of the database.

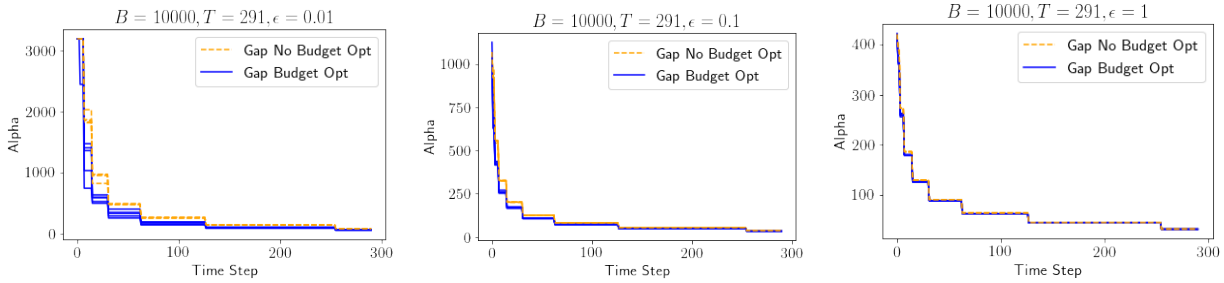

Figure 7.4: AVG delay WHERE month=1. Roughly %10 of the tuples satisfy this condition and it is clear that budget optimization is not necessary for dense queries.

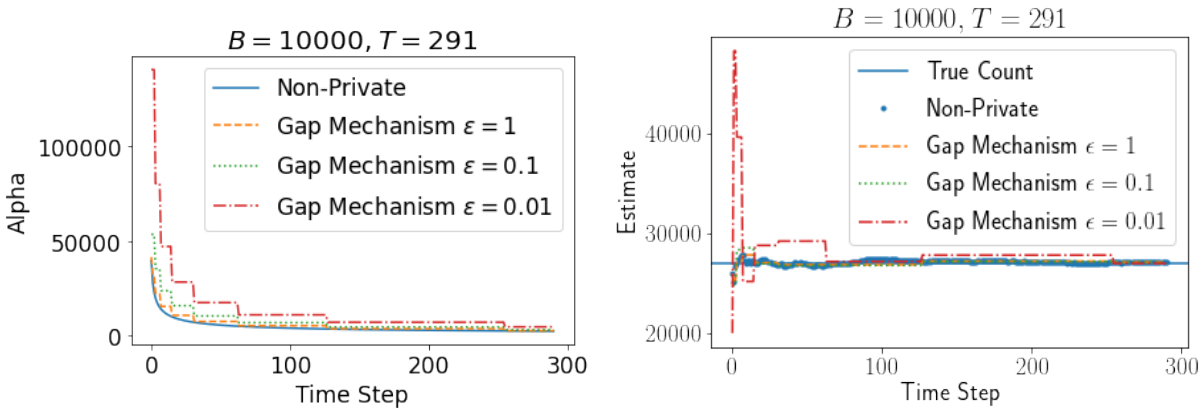

Figure 7.5: COUNT query WHERE airport="PHX". Gap based mechanisms preform relatively well across all ranges of ϵ when the tuples satisfying the query is dense enough (approximately 1% of database).

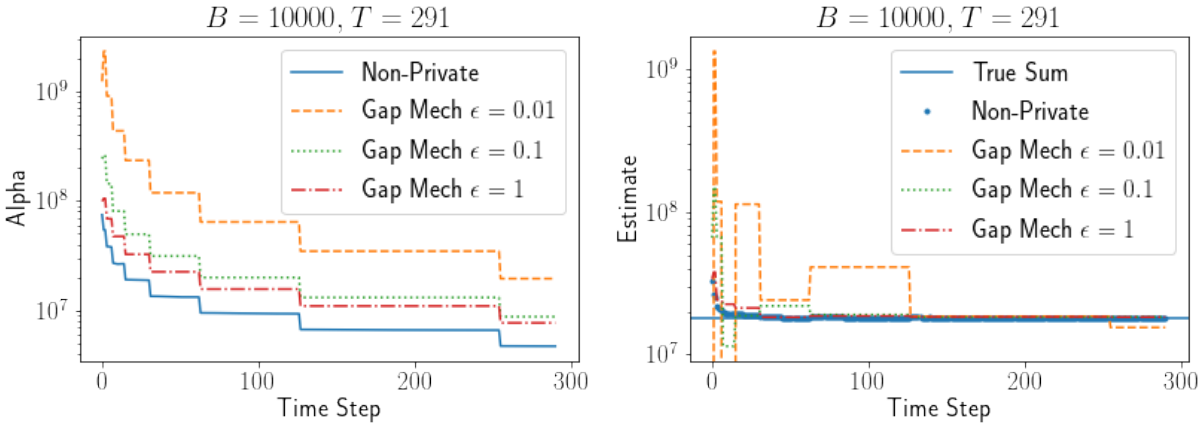

Figure 7.6: SUM Query WHERE month=1.

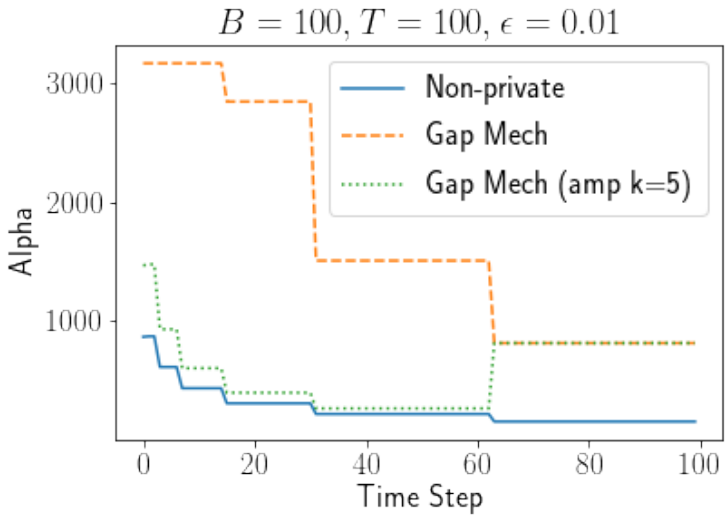

Figure 7.7: Single Gap mechanism for AVG with no WHERE conditions. Privacy amplification is applied to first $k = 5$ releases of the mechanism which corresponds to the first $2^5 - 1 = 63$ time steps. Amplification provides significant improvement in early stages even when $\epsilon = 0.01$.

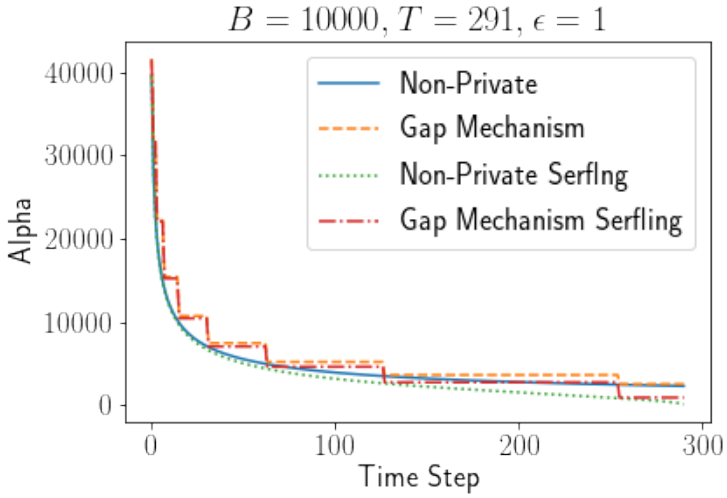

Figure 7.8: COUNT query WHERE airport="PHX". It can be seen that the Hoeffdig-Serfling based estimator provides significant improvements in the later time steps but is comparable to classic Hoeffding bound early on.

most time steps when $\epsilon = 1$. We will explain this phenomenon and go in-depth as to how the parameter choices change the performance of the algorithms.

In the first column of Figure 7.1, we see that the Single Gap Mech outperforms the Multi Gap Mech, the reason being that the error from Laplace noise dominates the error from sampling. Since our gap sizes are powers of 2, the Multi Gap Mech and even Baseline1 and Baseline2 use twice as many samples as Single Gap Mech. This leads to to a smaller sampling error but the additional $O(\sqrt{\log t})$, $O(T)$ and $O(\sqrt{t})$ factors on the error from Laplace noise make these a worse strategy compared to the $O(1)$ factor on the Single Gap Mech. In the second and third columns, we see a similar trend to the first but the larger privacy budget means that the Laplace noise quickly shrinks, and eventually the sampling error is the dominant term. Notice that the Baseline 2 strategy is a viable option in these high privacy budget cases.

It should also be noted that since the Hybrid Gap mechanisms alpha values are almost always smaller than the alpha values of the other mechanism, we essentially get that the score of the Hybrid Gap Mech given any increasing set of weights is going to be smaller than that of the other mechanisms. In figure 7.5, we again see those Gap based mechanisms perform well even for COUNT queries. It is fair to say Gap mechanisms perform the best overall.

7.4.2 Impact of Budget Optimization

Given a WHERE condition, we evaluate what is essentially the Single Gap Mech from the previous section modified as the total number of elements in the database satisfying the WHERE condition needs to be private. When splitting the privacy budget, we may simply split it in half for the noisy sum and the noisy count or we can optimize the budget appropriately. We perform experiments for the gap mechanism with and without budget optimization. Unlike the previous section where the alpha values will always be consistent, for this algorithm, a bad estimation of the count can lead to unreasonably large private alpha values. For that reason, we apply post-processing so that the maximum interval size corresponds to the range $[0, 6337]$. Furthermore, we run each algorithm a total of 5 times to get a better idea of the performance. To illustrate the performance based on the sparsity of the queries that satisfy the condition, we give three different conditions. Roughly, the first condition is 0.1% of the database the second is 1% and the third is 10%.

When taking the average with "WHERE month=1 AND airport=PHX" the total number of elements satisfying the condition is 2180. Unfortunately, this is too small of a count to get any meaningful aggregate with $\epsilon = 0.01$ as can be seen from the first chart in figure

7.2. Occasionally one can get an interval smaller than the full range with the optimized budget GAP mechanism but even then, on some iterations, one may get unlucky and get the largest possible alpha value. As ϵ gets larger we see that the budget optimization is less essential but still beneficial.

When using "WHERE airport = PHX", the total count is 27040. As can be seen in figure 7.3 both algorithms perform relatively well even when ϵ is small as we now have enough samples. As noted before, budget optimization provides more noticeable improvements for the smaller ϵ values. The same trend is noticed when we use "WHERE month = 1" which is a query that 238730 elements satisfy. The private alpha values are much smaller in this case as expected.

7.4.3 Impact of Privacy Amplification

In figure 7.7 we see the impact of privacy amplification on the early time steps when running the Gap mechanism for simple AVG queries. It should be noted that we post-process the alpha values so that the estimate always lies in the range $[0, 6337]$ which we did not do in figure 7.1. Notice that the alpha values for the amplified ranges are increasing rather than decreasing after time step 63. While this increase is brief and can be corrected through additional post-processing, it highlights that Laplace noise is the main source of error in the early steps. Even when the privacy loss is at most $\epsilon = 0.01$, by the privacy amplification results, for the first 5 gaps it suffices to use $\epsilon' = 0.468$. This improves the error from Laplace noise addition by a factor of 46. Contrast this with the results from figure 7.1 for the same parameter range of $\epsilon = 0.01$ and $B = 100$ and notice that in the chart, the initial alpha values correspond to intervals of size greater than 6337, where the size of an interval is the difference between the endpoints.

7.4.4 Impact of Improved Concentration Bound

In the early time steps, both the classic Hoeffding bound for sampling with replacement and the Serfling variant for sampling without perform similarly. This is because sampling a set of relatively small size from a large data set will result in very few duplicates essentially making them equivalent. In the later time steps where we have a much larger sample, sampling without replacement provides a noticeable improvement in reducing the sampling error.

Chapter 8

Conclusion and Open Questions

In this thesis, differentially private online aggregation is formulated for the first time. We identify how the desiderata for data analysis in the non-private online aggregation setting should be adapted for the differentially private online aggregation setting. Specifically, some requirements from the non-private setting such as wanting evenly paced updates are detrimental to private online aggregations' accuracy. We relax this requirement by introducing an objective function, which we call the score, that provides users with control over the accuracy and usability trade-offs.

Our main contribution is a family of differentially private algorithms we refer to as Gap mechanisms. Gap mechanisms can accurately answer common SQL queries such as AVG, COUNT, and SUM even in the presence of arbitrary WHERE conditions. Through our empirical study, we see that the Gap mechanisms perform the best overall in comparison to our baseline solutions and dominate especially when the privacy budget ϵ is small. Strategies for improving the Gap mechanisms by integrating privacy budget optimization, privacy amplification, and improved concentration bounds are discussed and empirically verified.

In future work, we will explore how distributional assumptions and techniques from private statistics [24, 6, 23, 10] can be leveraged to improve differentially private online aggregation. Furthermore, our work focused on single table queries but an interesting open problem is to adopt GROUP BY and JOIN queries to the private online aggregation setting.

References

[1] Airline on-time statistics and delay causes, Apr 2021.

[2] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by subsampling: Tight analyses via couplings and divergences, 2018.

[3] RÉMI BARDENET and ODALRIC-AMBRYM MAILLARD. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385, 2015.

[4] Rémi Bardenet and Odalric-Ambrym Maillard. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361 – 1385, 2015.

[5] Johes Bater, Yongjoo Park, Xi He, Xiao Wang, and Jennie Rogers. SAQE: practical privacy-preserving approximate query processing for data federations. *Proc. VLDB Endow.*, 13(11):2691–2705, 2020.

[6] Sourav Biswas, Yihe Dong, Gautam Kamath, and Jonathan R. Ullman. Coinpress: Practical private mean and covariance estimation. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[7] Adrian Rivera Cardoso and Ryan Rogers. Differentially private histograms under continual observation: Streaming selection into the unknown, 2021.

[8] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.

[9] Aloni Cohen and Kobbi Nissim. Linear program reconstruction in practice. *J. Priv. Confidentiality*, 10(1), 2020.

[10] Christian Covington, Xi He, James Honaker, and Gautam Kamath. Unbiased statistical estimation and valid confidence intervals under differential privacy, 2021.

[11] Rachel Cummings, Sara Krehbiel, Kevin A. Lai, and Uthaipon Tantipongpipat. Differential privacy for growing databases, 2018.

[12] Wei Dong and Ke Yi. A nearly instance-optimal differentially private mechanism for conjunctive queries, 2021.

[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[14] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 715–724. ACM, 2010.

[15] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[16] Hendrik Fichtenberger, Monika Henzinger, and Wolfgang Ost. Differentially private algorithms for graphs under continual observation. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pages 42:1–42:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[17] Simson L. Garfinkel, John M. Abowd, and Christian Martindale. Understanding database reconstruction attacks on public data. *Commun. ACM*, 62(3):46–53, 2019.

[18] Peter J. Haas and Joseph M. Hellerstein. Ripple joins for online aggregation. In Alex Delis, Christos Faloutsos, and Shahram Ghandeharizadeh, editors, *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA*, pages 287–298. ACM Press, 1999.

[19] Peter J. Haas, Jeffrey F. Naughton, S. Seshadri, and Arun N. Swami. Selectivity and cost estimation for joins based on random sampling. *Journal of Computer and System Sciences*, 52(3):550–569, 1996.

[20] Peter J. Haas, Jeffrey F. Naughton, S. Seshadri, and Arun N. Swami. Selectivity and cost estimation for joins based on random sampling. *J. Comput. Syst. Sci.*, 52(3):550–569, June 1996.

[21] Joseph M. Hellerstein, Peter J. Haas, and Helen J. Wang. Online aggregation. In Joan Peckham, editor, *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 171–182. ACM Press, 1997.

[22] Noah M. Johnson, Joseph P. Near, and Dawn Song. Towards practical differential privacy for SQL queries. *Proc. VLDB Endow.*, 11(5):526–539, 2018.

[23] Gautam Kamath, Vikrant Singhal, and Jonathan R. Ullman. Private mean estimation of heavy-tailed distributions. In Jacob D. Abernethy and Shivani Agarwal, editors, *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pages 2204–2235. PMLR, 2020.

[24] Vishesh Karwa and Salil P. Vadhan. Finite sample differentially private confidence intervals. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 44:1–44:9. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[25] Feifei Li, Bin Wu, Ke Yi, and Zhuoyue Zhao. Wander join and XDB: online aggregation via random walks. *SIGMOD Rec.*, 46(1):33–40, 2017.

[26] Stephen Macke, Maryam Aliakbarpour, Ilias Diakonikolas, Aditya G. Parameswaran, and Ronitt Rubinfeld. Rapid approximate aggregation with distribution-sensitive interval guarantees. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*, pages 1703–1714. IEEE, 2021.

[27] Darakhshan J. Mir, S. Muthukrishnan, Aleksandar Nikolov, and Rebecca N. Wright. Pan-private algorithms: When memory does not help. *CoRR*, abs/1009.1544, 2010.

[28] R. J. Serfling. Probability Inequalities for the Sum in Sampling without Replacement. *The Annals of Statistics*, 2(1):39 – 48, 1974.

[29] Yuchao Tao, Xi He, Ashwin Machanavajjhala, and Sudeepa Roy. Computing local sensitivities of counting queries with joins. In David Maier, Rachel Pottinger, AnHai Doan, Wang-Chiew Tan, Abdussalam Alawini, and Hung Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference*

2020, online conference [Portland, OR, USA], June 14-19, 2020, pages 479–494. ACM, 2020.

[30] Jonathan Ullman. Cs7880: Rigorous approaches to data privacy hw1, 2017.