Differentially Private Histogram Publication For Dynamic Datasets: An Adaptive Sampling Approach

Haoran Li
Emory University
Atlanta, GA
hli57@emory.edu

Li Xiong
Emory University
Atlanta, GA
lxiong@emory.edu

Xiaoqian Jiang
University of California, San Diego
La Jolla, CA
x1jiang@ucsd.edu

Jinfei Liu
Emory University
Atlanta, GA
jinfei.liu@emory.edu

ABSTRACT

Differential privacy has recently become a de facto standard for private statistical data release. Many algorithms have been proposed to generate differentially private histograms or synthetic data. However, most of them focus on "one-time" release of a static dataset and do not adequately address the increasing need of releasing series of dynamic datasets in real time. A straightforward application of existing histogram methods on each snapshot of such dynamic datasets will incur high accumulated error due to the composibility of differential privacy and correlations or overlapping users between the snapshots. In this paper, we address the problem of releasing series of dynamic datasets in real time with differential privacy, using a novel adaptive distance-based sampling approach. Our first method, DSFT, uses a fixed distance threshold and releases a differentially private histogram only when the current snapshot is sufficiently different from the previous one, i.e., with a distance greater than a predefined threshold. Our second method, DSAT, further improves DSFT and uses a dynamic threshold adaptively adjusted by a feedback control mechanism to capture the data dynamics. Extensive experiments on real and synthetic datasets demonstrate that our approach achieves better utility than baseline methods and existing state-of-the-art methods.

Categories and Subject Descriptors

H.2.7 [**Database Administration**]: [Security, integrity, and protection]; H.2.8DatabaseApplications[Data mining]

General Terms

Theory, Performance, Experimentation

Keywords

Differential privacy; adaptive sampling; dynamic dataset release

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
CIKM'15, October 19-23, 2015, Melbourne, VIC, Australia.
© 2015 ACM. ISBN 978-1-4503-3794-6/15/10...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806452.

1. INTRODUCTION

Sharing dynamic private data while providing privacy guarantee enables many important data mining and knowledge discovery applications. Consider the examples below:

Medical research: A hospital gathers data from individual patients every day. The dynamic datasets, e.g. the daily datasets of individual patients with fevers, coughs, and different demographic attributes can be shared with researchers for cohort discovery, medical research, and seasonal epidemic outbreak monitoring.

Traffic Monitoring: A GPS service provider gathers data from individual users about their locations, speeds, mobility, etc. The dynamic datasets, e.g., the numbers of users at different regions during each time period, can be mined for commercial interest, such as congestion patterns on the roads.

A common scenario of such applications is that a trusted server gathers data from a large number of individual subscribers. The aggregated data can be then continuously shared with other untrusted entities for various purposes. The trusted server, i.e. publisher, therefore must ensure that releasing the data does not compromise the privacy of any individual who contributed data. The goal of our work is to enable publishers to share a series of dynamic private datasets over individual users while guaranteeing their privacy.

The current state-of-the-art standard for privacy preserving data publishing is differential privacy [9, 27], which requires that the output released by a data provider be perturbed by a randomized algorithm $\mathcal{A}$, so that the output of $\mathcal{A}$ remains roughly the same even if any individual tuple in the input data is arbitrarily modified. Given the output of $\mathcal{A}$, an adversary will not be able to infer much about any individual tuple in the input, and thus privacy is protected.

Most existing works on differentially private data release focus on "one-time" release of static data (e.g. [20, 29, 26, 7, 17], etc). In this paper, we study the problem of releasing histograms for dynamic datasets while guaranteeing user-level differential privacy, i.e., protecting the presence of a user in the entire series of dynamic datasets. In the worst case, a user may be present in all datasets in the series. A straight-forward application of the standard differential privacy mechanism or existing histogram method to each snapshot of the dataset will lead to a very high perturbation error $O(N)$ in the order of the number of datasets or snapshots N in the series, due to the composition theorem [22].

A set of related works have studied the problem of releasing aggregate time series and stream statistics. The works in [12, 6] proposed differentially private continual counters over a binary stream. However, both works adopt an event-level differential privacy, which

protects the presence of an individual event, i.e. a user's contribution to the data stream at a single time point, rather than her presence or contribution to the entire series. The works in [25, 13, 14] studied the problem of releasing aggregate time-series with user-level differential privacy. Both works consider temporal correlations of the time-series. The paper [25] uses a Discrete Fourier Transform approach and is not applicable to real-time applications when data needs to be released at each time point. Other works [13, 14] take a model based approach which assumes original data is generated by an underlying process and uses the model based prediction to improve the accuracy of the released data. The limitation is that the model needs to be assumed or learned from public data with similar patterns and the method may not be effective when the real data deviates from the model.

The recent work [18] studies the problem similar to ours and represents the state-of-the-art. It proposed a novel w-event privacy framework by combining user-level and event-level privacy, which essentially guarantees user-level privacy within any window of w timestamps. When w is set to the number of time points in the series of data, or infinity for infinite data streams, it converges to user-level privacy. In addition, it proposed a sampling approach with various privacy budget allocation schemes to release data. However, in their schemes, privacy budgets may be exhausted prematurely or not fully utilized, still leading to suboptimal utility of the released data.

Our contributions. In this paper, we present a novel and principled adaptive distance-based sampling approach for releasing multiple histograms for a series of dynamic datasets in real time. We summarize the contributions and features of our approach below.

1). We propose a distance-based sampling approach to address the dynamics of evolving datasets under user-level differential privacy. Instead of generating a differentially private (DP) histogram at each time stamp, we only compute new histograms when the update is significant, i.e., the distance between the current dataset and the latest released dataset is higher than a threshold. Both the distance computation and threshold comparison are designed to guarantee differential privacy. The key observation is that datasets may be subject to small updates at times. Distance-based sampling allows us to release a new histogram only when the datasets have significant updates, hence saving the privacy budget and reducing the overall error of released histograms. In contrast to [18], we use an explicit threshold to determine the sampling points, inspired by the sparse vector technique [15] originally proposed for releasing DP counts only when the counts are greater than a threshold. The explicit threshold based sampling provides two advantages: 1) we can predefine a threshold based on the expected update rate of the data if there is prior domain knowledge, 2) we can dynamically adjust the threshold in a principled way based on data dynamics. Another important feature of our approach is that it is orthogonal to the histogram method used for each time point, i.e. it can use any of the state-of-the-art static differentially private histogram release method (e.g. [9, 28, 7, 26, 7, 17, 24, 20, 21, 30]) as a black box, which is efficient and effective for generating "one time" histograms.

2). We present two methods for defining the threshold. The first method, DSFT (*Distance*-based *Sampling* with *Fixed Threshold*), uses a predefined threshold T. The second improved method, DSAT (*Distance*-based *Sampling* with *Adaptive Threshold*), applies a feedback control mechanism to adaptively adjust the threshold T. Real world dynamic datasets may exhibit varying update behaviors across different settings. The adaptive threshold mechanism allows us to dynamically adjust the threshold T without having to rely on prior knowledge to tune the threshold. We use a PID (Proportional, Integral, and Derivative) controller [2] to detect the dynamics and adaptively adjust the threshold such that the privacy budget is not depleted prematurely due to high update and sampling rates or insufficiently utilized due to low update and sampling rates.

3). We present formal analysis of differential privacy guarantees, complexity, and utility for DSFT and DSAT. In our approach, each released DP histogram has either a perturbation error $O(C)$ at sampling points, where C is the maximum number of released DP histograms ($C << N$) or an update error with an upper bound (see Section 6). We also show a formal analysis of how to select optimal algorithmic parameters given a required utility guarantee.

4). In addition to standard user-level differential privacy, we further extend our methods under the framework of w-event privacy [18], so it can work with infinite series of evolving datasets.

5). Finally, we present extensive experiments using both synthetic and real datasets. Experiment results demonstrate that our methods significantly outperform the baseline approaches and existing state-of-the-art techniques [18].

We state the problem setting of releasing dynamic datasets under differential privacy in Section 3 and introduce w-event privacy and existing state-of-the-art solutions. We present our methods DSFT and DSAT while provide formal privacy analysis in Section 4, then follow by the utility analysis in Section 5. We extend our techniques to w-event privacy framework in Section 6. We include detailed experimental evaluation of our algorithms in Section 7 and conclude in Section 8.

2. RELATED WORK

Several mechanisms (e.g. [12, 6], etc) focus on event-level privacy in releasing counters, i.e. in publishing the number of event occurrences at every time point since the commencement of the system. These mechanisms consider the data stream as a bit string and at each time point they release the number of 1's seen so far. A set of related work focus on releasing aggregate time series or stream statistics under differential privacy as we discussed earlier [25, 13, 14]. The work in [25, 13] releases aggregate time-series with user-level differential privacy. Both works have some limitations as we discussed earlier. [31] releases dynamic transaction data under user-level privacy, and set an upper bound to limit the maximum number of updates to handle infinite updates. But it can only handle insertions updates.

The most recent work that is closely related to our work is Kellaris et al. [18] which deals with differentially private release of events or histograms for infinite stream. It proposes a w-event privacy framework by combining user-level and event-level privacy, which protects any event sequence occurring within any window of w timestamps. It is event-privacy with w = 1 and converges to user- level privacy with w = infinity. They also proposed two mechanisms, Budget Distribution (BD) and Budget Absorption (BA), to allocate the budget within one w-timestamp window. The key difference between our work and [18] is that our methods detect the data dynamics and adaptively adjust the distance threshold for sampling such that the privacy budget is not depleted prematurely due to high update and sampling rates or insufficiently utilized due to low update and sampling rates. In [18], privacy budgets may be depleted prematurely, especially when w is very large, or not fully utilized during the w timestamps. In addition, our method is independent of the histogram method used for each time point and can utilize any state-of-the-art histogram methods designed for static data release as a blackbox. In our experiments, we compare our methods with BD and BA in [18], since they represent the state-of-the-art and have been shown to perform better than other existing work.

Our distance threshold based sampling builds on top of the sparse vector technique [15] originally proposed for releasing differentially private counts only when the counts are greater than a threshold. The sparse vector technique has also been used in [19] for releasing top-k frequent itemsets given a static transaction dataset and a threshold derived from the kth frequent itemsets. In our work, we use the sparse vector technique in a novel way to enable differentially private distance based sampling for releasing dynamic datasets while adaptively adjusting the distance threshold.

3. PRELIMINARIES

In this section, we formally define the problem of releasing series of real-time dynamic histograms or datasets and introduce definitions on user-level differential privacy and w-event privacy. We summarize all frequently used notations in Table 1.

Table 1: Frequently used notations

Notation	Discription
$\mathbf{D}$	A series of original dynamic datasets
$\tilde{\mathbf{D}}$	A set of released DP datasets for $\mathbf{D}$
D_i or $\tilde{D}_i$	Snapshot of $\mathbf{D}$ or $\tilde{\mathbf{D}}$ at time point t_i
$\mathbf{H}$	A series of original dynamic histograms
$\tilde{\mathbf{H}}$	A set of released DP histograms for $\mathbf{H}$
H_i or $\tilde{H}_i$	Snapshot of $\mathbf{H}$ or $\tilde{\mathbf{H}}$ at time point t_i
N	Number of time points
C	Cutoff point (i.e. the upper bound of the number of released DP datasets)
U	Domain universe or number of histogram bins
ϵ	Overall privacy budget
ϵ_1	Privacy budget for the decision step
ϵ_2	Privacy budget for the sampling step
$d(D_i, \tilde{D}_j)$	The distance between D_i and $\tilde{D}_j$
Δ	The sensitivity of L_1 distance

3.1 Problem definition

Let N denote the total number of time points. Let $\mathbf{D}$ denote a series of original dynamic datasets and D_i be a dataset snapshot at time stamp t_i. We assume all snapshots have the same domain universe U, the product of domains of all attributes. For every t_i, we are to release a private dataset $\tilde{D}_i$. Over the N time stamps, the series of privately released dynamic datasets $\tilde{\mathbf{D}} = \{\tilde{D}_i : 1 \leq i \leq N\}$ should guarantee user-level ϵ-differential privacy.

In this paper, we call $\mathbf{H}$ as a series of original dynamic histograms (corresponding to $\mathbf{D}$) with H_i as a snapshot at t_i, and $\tilde{\mathbf{H}}$ as a series of released private dynamic histograms with $\tilde{H}_i$ as a private snapshot at t_i. Since a dataset can be transformed to a histogram, and a synthetic dataset can be constructed from a histogram, $\mathbf{D}$ and $\mathbf{H}$ are interchangeable in this paper.

3.2 Differential Privacy

Intuitively, a randomized mechanism $\mathcal{A}$ is differentially private if its outcome is not significantly affected by the removal or addition of any record. ϵ-differential privacy is formally defined as $Pr[\mathcal{A}(D) \in \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(D') \in \mathcal{O}]$, where $\mathcal{O}$ is any arbitrary set of possible outputs of $\mathcal{A}$, D and D' are two neighbouring datasets differing in at most one record (i.e. D can be obtained from D' by adding or removing at most one record). In our problem definition, an adversary should learn approximately the same information about any individual user given $\tilde{\mathbf{D}}$, irrespective of its presence or absence in $\mathbf{D}$, and one individual can be present in up to N snapshots in $\mathbf{D}$. Two series of dynamic datasets $\mathbf{D}$ and $\hat{\mathbf{D}}$ are user-level neighbors if one can be obtained by adding or removing one individual (including all its occurrences in the snapshots) from the other. Then user-level ϵ-differential privacy is defined as below.

DEFINITION 3.1 (USER-LEVEL ϵ-DIFFERENTIAL PRIVACY). *Let $\mathcal{A}$ be a randomized mechanism over two user-level neighbors $\boldsymbol{D}$, and $\hat{\boldsymbol{D}}$ which differ in one user's presence in the entire series, and let $\mathcal{O}$ be any arbitrary set of possible outputs of $\mathcal{A}$. Algorithm $\mathcal{A}$ satisfies ϵ-differential privacy iff the following holds*

$$Pr[\mathcal{A}(\boldsymbol{D}) \in \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(\hat{\boldsymbol{D}}) \in \mathcal{O}]$$

Laplace Mechanism. Dwork et al. [11] show that ϵ-differential privacy can be achieved by adding i.i.d. Laplace noise to query result $q(D)$, where D is a dataset. Formally, $\tilde{q}(D) = q(D) + (\nu_1, \ldots, \nu_M)'$, where $\nu_i \sim Lap(0, \frac{GS(q)}{\epsilon})$, for $i = 1, \ldots, M$, and M is the dimension of $q(D)$. ν_i follows a Laplace distribution with mean zero and scale $\frac{GS(q)}{\epsilon}$, where $GS(q)$ denotes the global sensitivity [11] of the query q. The global sensitivity is the maximum L_1 distance between the results of q from any two neighbouring datasets D and D', formally defined as $GS(q) = max_{D,D'} ||q(D) - q(D')||_1$. In our problem setting, the global sensitivity of any two user-level neighbors $\mathbf{D}$ and $\hat{\mathbf{D}}$ is formally defined as

$$GS(q) = max_{\mathbf{D},\hat{\mathbf{D}}} ||q(\mathbf{D}) - q(\hat{\mathbf{D}})||_N.$$

For a sequence of DP mechanisms, the sequential composition theorem [22] guarantees its overall privacy as follows:

THEOREM 3.1 (SEQUENTIAL COMPOSITION [22]). *For a sequence of n mechanisms $M_1, \ldots, M_n$ and each M_i provides ϵ_i-differential privacy, the sequence of M_i will provide $(\sum_{i=1}^{n} \epsilon_i)$ differential privacy.*

Hence, one way to achieve epsilon-differential privacy for the entire series of $\mathbf{D}$ is to apply Laplace mechanism for each D_i with noise $Lap(\frac{N}{\epsilon})$, which leads to $O(N)$ noise.

(α, σ)-usefulness. We use a formal utility metric (α, σ)-usefulness [3] to analyze the utility of each snapshot $\tilde{D}_i$ in $\tilde{\mathbf{D}}$.

DEFINITION 3.2 ((α, σ)-USEFULNESS). *A randomized mechanism $\boldsymbol{A}$ is (α, σ)-useful for queries in class $\mathcal{C}$ if with probability $1 - \sigma$, for every query $Q \in \mathcal{C}$ and a dataset D, $\boldsymbol{A}(D) = \tilde{D}$, $|Q(D) - Q(\tilde{D})| \leq \alpha$.*

3.3 w-event privacy

W-event privacy [18] is proposed as an extension of differential privacy to address release of infinite streams. It guarantees user-level ϵ-differential privacy for every sub sequence of length w (or over w timestamps) **anywhere** (i.e. it can start from any timestamp) in the original series of dynamic datasets. w-neighboring series of dynamic datasets, $\mathbf{D}_w$, and $\hat{\mathbf{D}}_w$, can be defined as the user-level neighbors under any sub sequence of length w anywhere. w-event privacy can be formally given as below:

DEFINITION 3.3 (W-EVENT ϵ-DIFFERENTIAL PRIVACY). *Let $\mathcal{A}$ be a randomized mechanism over two w-neighboring series of dynamic datasets $\boldsymbol{D}_w$, and $\hat{\boldsymbol{D}}_w$, and let $\mathcal{O}$ be any arbitrary set of possible outputs of $\mathcal{A}$. Algorithm $\mathcal{A}$ satisfies w-event ϵ-differential privacy (or, w-event privacy) iff the following holds*

$$Pr[\mathcal{A}(\boldsymbol{D}_w) \in \mathcal{O}] \leq e^{\epsilon} Pr[\mathcal{A}(\tilde{\boldsymbol{D}}_w) \in \mathcal{O}]$$

3.4 Baseline and Existing State-of-the-Art Solutions

Given our problem of releasing dynamic datasets under user-level privacy, we review some baseline and existing state-of-the-art

methods which will motivate our approach. We will also compare our approach with these methods in the experiment section.

Baseline method. A baseline method is to apply existing "one time" DP histogram release methods to the dataset at every time point. If each released DP histogram preserves $\frac{\epsilon}{N}$-differential privacy, the series of N dynamic datasets guarantee ϵ-differential privacy by sequential composition theorem. This results in an overall noise of $O(N)$ which can be extremely large for large N. In an unbounded setting with N being infinite, this method will not be useful.

Fixed-sampling method. Another potential solution is to release $\frac{N}{I}$ DP histograms periodically given a sampling interval I. Privacy budget $\epsilon/\frac{N}{I}$ is allocated to each dataset at the sampling time point, and the entire private dataset series preserve ϵ-differential privacy. Unfortunately, the pre-defined sampling interval may not accurately capture the update pattern in the original series of dynamic datasets, leading to either high perturbation errors if sampling too frequently or large update errors if sampling not frequently enough or at wrong time points.

Approaches in w-event privacy. [18] proposes a sampling approach which computes the noisy distance between the dataset at the current time point and the original dataset at the latest sampling point, and then compares the noisy distance with the perturbation noise to be added if current dataset is to be released. If the distance is greater than the perturbation noise, a noisy dataset is released at current time stamp. The perturbation noise is determined by their privacy budget allocation schemes, Budget Distribution (BD) and Budget Absorption (BA), that allocate the budget to different timestamps in the w-event window. BD allocates the privacy budget in an exponentially decreasing fashion, in which earlier timestamps obtain exponentially more budget than later ones. BA starts by uniformly distributing the budget to all w timestamps, and accumulates the budget of non-sampling timestamps, which can be allocated later to the sampling timestamps. A main drawback of their approach is that the privacy budget may be exhausted prematurely (sampling too frequently in the beginning) or not fully utilized during all w timestamps (sampling not frequent enough), leading to suboptimal utility of the released data.

4. ADAPTIVE SAMPLING APPROACH

We propose an adaptive distance-based sampling approach to address the dynamics of evolving datasets under user-level differential privacy. Instead of generating a differentially private histogram at each time stamp, we only compute new histograms when the update is significant, i.e., the distance between the current dataset and the latest released dataset is higher than a threshold. The key observation is that datasets may be subject to small updates at times. Distance-based sampling allows us to release a new histogram only when the datasets have significant updates, hence saving the privacy budget and reducing the overall error of released histograms. In contrast to [18], we use an explicit threshold for distance comparison to determine the sampling points, which provides two advantages: 1) we can predefine a threshold based on the expected update rate of the data if there is prior domain knowledge, 2) we can dynamically adjust the threshold in a principled way based on data dynamics.

In this section, we first present the basic method, DSFT, which uses a predefined fixed threshold. This will allow us to analyze its privacy property which also applies to our adaptive method and facilitate our description of the adaptive method. We then introduce our adaptive method, DSAT, which dynamically adjusts the threshold in a principled way to adapt to the data dynamics.

4.1 DSFT

DSFT (Distance-based Sampling with Fixed Threshold) uses a fixed threshold and is divided into two steps at each time point ti: decision and sampling. The decision step computes a noisy distance between the original dataset H_i at current time stamp and the latest released histogram $\tilde{H}_j$ and determines if it is larger than a noisy threshold $\tilde{T}$. If yes, the sampling step generates a new DP histogram $\tilde{H}_i$, otherwise it outputs the previous $\tilde{H}_j$. The overall privacy budget is divided between the decision (ϵ_1) and sampling (ϵ_2) steps which are designed to guarantee differential privacy as we will analyze later.

Algorithm 1 presents DSFT. Line 1-4 initializes the privacy budget for the two steps, computes the noisy threshold, and releases a DP histogram at the first time stamp. Line 5-11 carry out the decision (line 7-8) and sampling (line 8-9) steps for each time point t_i if the number of released histograms is below the cutoff point C, and releases the last histogram with all remaining budget. For the distance $d(H_i, \tilde{H}_j)$, we use the L_1 distance in our implementation and other distance metrics (e.g. KL divergence) can be also used.

Algorithm 1 Distance-based Sampling with Fixed Threshold Algorithm (DSFT)

Input: $\mathbf{D} = \{D_i | 1 \leq i \leq N, i \in Z\}$, T, C and ϵ.
Output: $\tilde{\mathbf{D}} = \{\tilde{D}_i | 1 \leq i \leq N, i \in Z\}$
1: Set $\epsilon_1 = k\epsilon$, $\epsilon_2 = \epsilon - \epsilon_1$, k is computed due to theorem 5.4;
2: Set $\tilde{T} = T + Lap(\frac{2\Delta}{\epsilon_1})$, Δ is computed due to lemma 4.1;
3: For D_1, release a DP dataset $\tilde{D}_1$ with $\frac{\epsilon_2}{C}$ privacy budget;
4: Set $count = 1$, and $j = 1$;
5: **for** each time point t_i with $i \geq 2$ **do**
6: **if** $count \geq C$, **then** set $\tilde{D}_i = \tilde{D}_j$ **continue**;
7: Set $\tilde{d}(D_i, \tilde{D}_j) = d(D_i, \tilde{D}_j) + Lap(\frac{2C\Delta}{\epsilon_1})$
8: **if** $\tilde{d}(D_i, \tilde{D}_j) \geq \tilde{T}$, **then** release $\tilde{D}_i$ at t_i with $\frac{\epsilon_2}{C}$ budget, and set $count = count + 1$, and $j = i$;
9: **else** use $\tilde{D}_j$ as the release of D_i;
10: **if** $i == N$ and $count < C$, **then** release $\tilde{D}_N$ with all remaining privacy budget;
11: **end for**

4.2 DSAT

In DSFT, a prior knowledge on **D** is needed for the user to determine an appropriate value T. Suppose there exists an optimal value of T which can enable the algorithm to exactly generate C DP histograms. If the threshold T is higher than the optimal value, there will be remaining privacy budgets that are not utilized. On the contrary, if T is smaller than the optimal value, the privacy budget will be exhausted prematurely, resulting in update errors for remaining time points. In this section, we present DSAT, Distance-based Sampling with Adaptive Threshold, that releases a series of DP dynamic histograms while adaptively adjusting the threshold T_i for each time point, based on data dynamics. With DSAT, we do not have to find an optimal value of T, which may be difficult in practice.

Figure 1 illustrates the framework of DSAT. Intuitively, we wish to have C sampling points over N time points, hence our target sampling rate is $\frac{C}{N}$. Suppose we have released C_i histograms at t_i. If $\frac{C_i}{i} < \frac{C}{N}$, we need to decrease the threshold to allow more sampling time points, and vice versa. For each t_i, we adjust the threshold based on the feedback error between the update ratio $\frac{C_i}{i}$ at t_i and the target ratio $\frac{C}{N}$, which is formally defined below.

DEFINITION 4.1 (FEEDBACK ERROR). *We define the feedback*

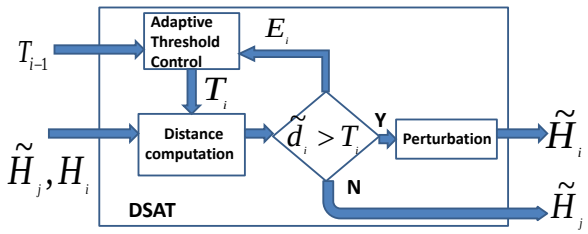


Figure 1: DSAT Framework

error E_i at t_i as follows:

$$E_i = |\frac{C_i}{i} - \frac{C}{N}| \tag{1}$$

where C_i means the number of sampling time points till t_i, C is the cutoff point, and N is the total number of time points.

DSAT adopts a PID (Proportional-Integral-Derivative) [2], a generic control loop feedback mechanism, to dynamically adjust the threshold T over time. Under our problem setting, we redefine the three correcting terms, *Proportional*, *Integral*, and *Derivative*, with the feedback error defined in Equation (1). These three terms are summed to compute the output u_i of PID controller at t_i. The final PID algorithm is defined as:

$$u_i = \underbrace{\theta_P \times e_i}_{proportional\ \ term} + \theta_I \times \underbrace{\sum_{\tau=t_i-w+1}^{t_i} e_\tau}_{integral\ \ term} + \underbrace{\theta_D \times \frac{e_i - e_j}{t_i - t_j}}_{derivative\ \ term} \tag{2}$$

where $\theta_P, \theta_I, \theta_D$ are respectively the proportional gain, the integral gain, and the derivative gain, e_τ is the error at t_τ, t_i is the current time point, t_j is the latest sampling time point.

Proportional term: The first proportional term produces an output value that is proportional to the current error e_i. The proportional term can be amplified by the *proportional gain* θ_P. In our context, the error e_i at the current time point t_i is calculated by

$$e_i = \frac{|E_i - \delta|}{\delta} \tag{3}$$

where E_i is the feedback error defined in equation (1), parameter δ is the set point for E_i. We assume δ is 5% in our empirical studies, i.e. the maximum tolerance for the feedback error is 5%. It can be determined by users according to specific applications. The proportional term is defined as $\theta_P \times e_i$.

Integral term: The integral term is to eliminate the cumulated offset through multiplying the sum of the instantaneous error over time by the integral gain. We define the integral term as $\theta_I \times \sum_{\tau=t_i-w+1}^{t_i} e_\tau$, where θ_I is the *integral gain* and w represents the integral time window denoting how many recent errors are taken.

Derivative term: The derivative term determines the slope of error over time and changes the PID output in proportion to this rate of change via the *derivative gain* θ_D. It is defined as $\theta_D \times \frac{e_i - e_j}{t_i - t_j}$. Given the PID error u_i, a new threshold T_i produced at the current time point t_i can be determined as follows:

$$T_i = T_{i-1} + sign(\frac{C_i}{i} - \frac{C}{N}) \times \theta \times u_i \tag{4}$$

T_{i-1} is the threshold produced at the previous time point t_{i-1}. Parameter θ determines the magnitude of impact of PID error on the T_i. $sign(.)$ is a sign function, indicating that if the update ratio $\frac{C_i}{i}$ is larger than the target ratio $\frac{C}{N}$, we need to increase T_j to generate less DP histograms and reduce the update ratio, and vice versa. Our DSAT uses only the proportional term in equation 2 in our experiment setting, for simplicity. That means, we set $\theta_P = 1$, $\theta_I = 0$, $\theta_D = 0$, and u_i is the same with e_i as defined in equation (3).

Algorithm 2 Distance-based Sampling with Adaptive Threshold Algorithm (DSAT)

Input: $\mathbf{D} = \{D_i | 1 \le i \le N, i \in Z\}$, T, C and ϵ.
Output: $\tilde{\mathbf{D}} = \{\tilde{D}_i | 1 \le i \le N, i \in Z\}$

1: Run step 1,2,3,4 in Algorithm 1;
2: Skip the first M timestamps;
3: **for** each time point t_i with $i > M$ **do**
4: **if** $count \ge C$, **then** set $\tilde{D}_i = \tilde{D}_j$
5: Set $\tilde{d}(D_i, \tilde{D}_j) = d(D_i, \tilde{D}_j) + Lap(\frac{2C\Delta}{\epsilon_1})$
6: Set $E_i = |\frac{count}{i} - \frac{C}{N}|$, $e_i = \frac{|E_i - \delta|}{\delta}$, and $u_i = \theta e_i$;
7: **if** $\frac{count}{t} - \frac{C}{N} \le 0$, **then** set $\tilde{T}_i = max\{0, \tilde{T}_{i-1} - u_i\}$
8: **else** set $\tilde{T}_i = min\{2, \tilde{T}_{i-1} + u_i\}$;
9: **if** $\tilde{d}(D_i, \tilde{D}_j) \ge \tilde{T}_i$ **then**
10: release a DP dataset $\tilde{D}_i$ at t_i with $\frac{\epsilon_2}{C}$ budget, and set $count = count + 1$, $j = i$;
11: **else**
12: release $\tilde{D}_j$;
13: **end if**
14: **if** $i == N$ and $count < C$ **then**
15: release $\tilde{D}_N$ with all remaining privacy budget;
16: **end if**
17: **end for**

Algorithm 2 presents DSAT. We use T_i to denote the produced threshold at t_i and other notations are the same as Algorithm 1. In Line 1, T_1 is set to be $T + Lap(\frac{\Delta}{\hat{\epsilon}_1})$. Different from Algorithm 1, $\hat{\epsilon}_1$ is a tiny privacy budget because the initial value T_1 is not significant in DSAT. We only need to bound it between 0 and 2, which is the domain of the L1 distance. Line 2 uses $\tilde{D}_1$ for the first M time points where M is a small integer number to allow a burn-in period and enough discrepancy to be accumulated, avoiding frequent updating of T_i during the beginning time periods. M can be user-specified and is not a sensitive parameter besides that it is much smaller than N. The algorithm from Line 3 to Line 12 is similar to Algorithm 1 except Line 6 to Line 8 which use the PID control to adaptively adjust and generate a new threshold T_i.

4.3 Privacy Analysis

Sensitivity analysis of L_1 Distance. In the sensitivity analysis, we use n_p (n_q) to denote the sum of all histogram bin counts of the histograms H_p (H_q). U is the number of histogram bins. Since the L_1 distance of Algorithm 1 and Algorithm 2 is computed using one private histogram and one original histogram, we only need to protect privacy for the original histogram.

LEMMA 4.1. *The sensitivity of L_1 distance $d(\tilde{H}_p, H_q)$ is $\Delta = \frac{2}{n_q - 1}$, where $\tilde{H}_p$ (H_q) is a DP histogram with the sum of all histogram bin counts as n_p (n_q). (Proof omitted due to space limitation)*

Privacy guarantee. Inspired by Hardt et al. [15], we formally provide the proof of privacy guarantee for the decision stage below. The intuition behind theorem 4.1 is that, the noises on both sides of $d(D_i, \tilde{D}_j) + Lap(\frac{2C\Delta}{\epsilon_1}) \ge T + Lap(\frac{2\Delta}{\epsilon_1})$ are necessary for the decision stage to be differentially private, even though T is publicly known.

THEOREM 4.1. *In algorithm 1, the decision stage guarantees ϵ_1-differential privacy.*

PROOF. $\mathbf{D}$ is a series of dynamic datasets with $\mathbf{D} = (D_1, \ldots, D_N)$ over N time points. $\hat{\mathbf{D}}$ is the user-level neighbor of $\mathbf{D}$, which is

$\hat{\mathbf{D}} = (\hat{D}_1, \ldots, \hat{D}_N)$. We say $\hat{\mathbf{D}}$ is the user-level neighbour of $\mathbf{D}$ if we can obtain $\hat{\mathbf{D}}$ by removing or adding only one individual user from $\mathbf{D}$ by the definition in section 3.2.

Let d_i denote $d(D_i, \tilde{D}_j)$ for every $i, j \in [N]([N] = \{1, \ldots, N\})$ beginning with $i = 2$, which is the true distance between D_i ad $\tilde{D}_j$, where $\tilde{D}_j$ is the private dataset released in the latest sampling time point t_j. Let $\tilde{d}_i$ denote $\tilde{d}(D_i, \tilde{D}_j)$, which is the DP L_1 distance.

For all pairs of user-level neighbours $\mathbf{D}$ and $\hat{\mathbf{D}}$, and the corresponding L_1 distance vectors $\mathbf{d} = (d_1, \ldots, d_N)$, we need to prove:

$$\log(\frac{\Pr_{\mathbf{D}}[d = \tilde{d}]}{\Pr_{\hat{\mathbf{D}}}[d = \tilde{d}]}) = \sum_{i=1}^{N} \log(\frac{\Pr_{\mathbf{D}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}{\Pr_{\hat{\mathbf{D}}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}) \leq \epsilon_1.$$

Because d_i is affected only by d_{i-1} at the previous time point, we have $\Pr_{\mathbf{D}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}] = \Pr_{\hat{\mathbf{D}}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}, \ldots, \tilde{d}_1]$. Let $S = \{i : \tilde{d}_i \geq \tilde{T}\}$ be the set of indices of $\tilde{d}_i$ at all sampling time points, and $S^C = \{i : \tilde{d}_i \leq \tilde{T}\}$ be the set of indices of $\tilde{d}_i$ at all non-sampling time points, we have $\log(\frac{\Pr_{\mathbf{D}}[d=\tilde{d}]}{\Pr_{\hat{\mathbf{D}}}[d=\tilde{d}]}) =$
$\sum_{i \in S} \log(\frac{\Pr_{\mathbf{D}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}{\Pr_{\hat{\mathbf{D}}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}) + \sum_{i \in S^C} \log(\frac{\Pr_{\mathbf{D}}[d_i = \emptyset | \tilde{d}_{i-1}]}{\Pr_{\hat{\mathbf{D}}}[d_i = \emptyset | \tilde{d}_{i-1}]})$.

Now we need to bound the two sums respectively. For the first sum, we can see that (1) independent Laplace noise with $Lap(\frac{2C\Delta}{\epsilon_1})$ is added to each distance with $\frac{\epsilon_1}{2C}$ differential privacy, (2) the computation of each L_1 distance needs to access the original histogram once, and (3) $|S| \leq C$ due to the algorithm, so we can obtain the following equation due to sequential composition theorem:

$$\sum_{i \in S} \log(\frac{\Pr_{\mathbf{D}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}{\Pr_{\hat{\mathbf{D}}}[d_i = \tilde{d}_i | \tilde{d}_{i-1}]}) = \sum_{i \in S} \log(\frac{\Pr_{\mathbf{D}}[\nu_i = \tilde{d}_i - d_i]}{\Pr_{\hat{\mathbf{D}}}[\nu_i = \tilde{d}_i - d_i]}) \leq \frac{\epsilon_1}{2}$$

For the second sum, let $A_Z(\mathbf{D})$ be the set of all values of the noise variables $(\nu_1, \ldots, \nu_{N-1})$ that cause $\tilde{d}_i \leq \tilde{T}$ for all $i \in S^C$, when the mechanism runs on $\mathbf{D}$, conditioning on $\tilde{T} = Z$ and $d_i = \tilde{d}_i$ for all $i \in S$. Since from $\mathbf{D}$ to $\hat{\mathbf{D}}$, all distances may be increased by at most Δ (i.e. $\Delta = \frac{2}{n-1}$ for the L_1 distance due to lemma 4.1), which will cause each distance to remain less than $\tilde{T}$ if we increase $\tilde{T}$ by Δ. But the distances larger than $\tilde{T}$ may become less than $\tilde{T} + \Delta$, so $A_{\tilde{T}+\Delta}(\hat{\mathbf{D}}) \subseteq A_{\tilde{T}}(\mathbf{D}) \subseteq A_{\tilde{T}+\Delta}(\hat{\mathbf{D}})$. Thus, we have: $\frac{\Pr_{\mathbf{D}}\{\tilde{T}=T+\nu_1\}}{\Pr_{\hat{\mathbf{D}}}\{\tilde{T}=T+\Delta+\nu_2\}} \leq \exp(\frac{\epsilon_1}{2})$. Therefore, let $Z_1 = T + \nu_1$ and $Z_2 = T + \Delta + \nu_2$, we have:

$$\begin{aligned}
&\prod_{i \in N^C} \Pr_{\mathbf{D}}(d_i = \emptyset | \tilde{d}_{i-1}) \\
=&\int_{-\infty}^{\infty} \Pr(\hat{d} = Z_1) \Pr((\nu_1, \ldots, \nu_k) \in A_Z(D)) dZ \\
\leq&\exp(\frac{\epsilon_1}{2}) \int_{-\infty}^{\infty} \Pr(\hat{T} = Z_1) \Pr((\nu_1, \ldots, \nu_k) \in A_{Z_1}(D)) dZ \\
\leq&\exp(\frac{\epsilon_1}{2}) \int_{-\infty}^{\infty} \Pr(\hat{T} = Z_2) \Pr((\nu_1, \ldots, \nu_k) \in A_{Z_2}(D')) dZ \\
=&\exp(\frac{\epsilon_1}{2}) \int_{-\infty}^{\infty} \Pr(\hat{T} = Z_1) Pr((\nu_1, \ldots, \nu_k) \in A_{Z_1}(D')) dZ \\
=&\exp(\frac{\epsilon_1}{2}) \prod_{i \in N^C} \Pr_{\hat{\mathbf{D}}}(d_i = \emptyset | \tilde{d}_{i-1})
\end{aligned}$$

Therefore, we have $\frac{\prod_{i \in N^C} \Pr_{\mathbf{D}}(d_i = \emptyset | \tilde{d}_{i-1})}{\prod_{i \in N^C} \Pr_{\hat{\mathbf{D}}}(d_i = \emptyset | \tilde{d}_{i-1})} \leq \frac{\epsilon_1}{2}$ □

THEOREM 4.2. *Algorithm 1 and 2 preserve ϵ-differential privacy.*

PROOF. For Algorithm 1, the decision stage preserves ϵ_1-differential privacy due to theorem 4.1. Since releasing at most C DP histograms guarantees ϵ_2-differential privacy, algorithm 1 preserves $\epsilon_1 + \epsilon_2 = \epsilon$-differential privacy due to theorem 3.1. For Algorithm 2, since adaptively adjusting threshold (Line 6 to Line 8) uses no raw data, it does not influence differential privacy guarantee, thus Algorithm 2 guarantees ϵ-differential privacy. □

5. UTILITY ANALYSIS

We analyze the utility of DSFT and DSAT using (α, σ)-usefulness in definition 3.2 and show the conclusions in theorem 5.1 and 5.2. Since we assume LPA as the DP histogram release method, the conclusions can be heuristically used as the upper bound when new methods better than LPA are employed. Here $d(H_i, H_j)$ denotes L_1 distance between H_i and H_j.

Error quantification of DSFT. The utility of released datasets at sampling time points are analyzed based on lemma 5.1. The error of datasets at non-sampling time points are obtained via the error bound of the decision stage in lemma 5.2.

LEMMA 5.1. *(Sum of Independent Laplace variables [6]) Suppose that $X_1, \ldots, X_n$ are independent Laplace random variables, with each X_i following a $Lap(b_i)$ distribution. Denote $Z = \sum_{i=1}^{n} X_i$ and $b_M = max_i b_i$. Then for all $\gamma \geq \sqrt{\sum_{i=1}^{n} b_i^2}$ and $0 < \lambda < \frac{2\sqrt{2}\gamma^2}{b_M}$, we have $Pr[Z > \lambda] \leq exp(-\frac{\lambda^2}{8\gamma^2})$.*

LEMMA 5.2. *In Algorithm 1, for any $0 < \sigma < 1$, we can obtain*

$$Pr\{d(H_i, \tilde{H}_j) \leq T + \frac{4\sqrt{2}\Delta(1 - log\sigma)}{\epsilon_1}\} \geq 1 - \sigma \quad (5)$$

This means, with probability greater than $1 - \sigma$, we can set t_i as non-sampling time points. (Proof omitted due to space limitation.)

THEOREM 5.1. *For a range count query covering m histogram bins on $\tilde{H}_k$, and $0 < \sigma < 1$, if k is a sampling time point, we have that $Pr\{|A_k - \tilde{A}_k| \leq -\frac{2\sqrt{2}Clog(\frac{\sigma}{2})}{n\epsilon_2}\} \geq 1 - \sigma$, and if k is a non-sampling time point, we have that $Pr\{|A_k - \tilde{A}_k| \leq T + \frac{4\sqrt{2}\Delta[1-log(1-\sigma)]}{\epsilon_1} - \frac{2\sqrt{2}Clog(\frac{\sigma}{2})}{\epsilon_2}\} \geq (1-\sigma)^2$, where A_k and $\tilde{A}_k$ are the query answers on the original histogram H_k and the DP histogram $\tilde{H}_k$. Therefore, each released histogram $\tilde{H}_k$ of our algorithms maintains (α, σ)-usefulness for range count queries.*

Error quantification of DSAT. We analyze the utility of DSAT based on theorem 5.2, and give the conclusion as below.

THEOREM 5.2. *For a range count query covering m histogram bins on $\tilde{H}_k$, and $0 < \sigma < 1$, if k is a sampling time point, the conclusion is the same as theorem 5.1 and if k is a non-sampling time point, we have $Pr\{|A_k - \tilde{A}_k| \leq T_k + \sum_{i=1}^{k} I_i u_i - \frac{2\sqrt{2}Clog(\frac{\sigma}{2})}{n\epsilon_2}\} \geq (1-\sigma)^2$, where I_i is a value being 1 or -1, and dependent on the data, and u_i is defined in equation (2). Therefore, each released histogram $\tilde{H}_k$ of our algorithms maintains (α, σ)-usefulness for range count queries. (The conclusion can be obtained via equation (5) and we omitted the full proof.)*

Lower bound of the data cardinality. Since the injected noise in the decision stage is related with data cardinality, we analyze the lower bound of data cardinality to guarantee a relatively small injected noise compared to the true L_1 distance. This lower bound can be used to maintain a high accuracy at the decision stage.

THEOREM 5.3. *In DSFT, in order to satisfy (α, σ)-usefulness and guarantee the utility of the decision stage, it requires that $n \geq \frac{16\sqrt{2}\alpha(1-log(1-\sigma))}{T\epsilon_1}$, where σ is defined in lemma 5.2. (Proof can be deducted from lemma 5.2)*

Select the value of k in DSFT. Our algorithm requires ϵ to be divided between ϵ_1 and ϵ_2 with $\epsilon_1 = k\epsilon$. We now analyze how to select k. Assume $\mathbf{H} = (H_1, \ldots, H_N)$ corresponds to $\mathbf{D}$. For each i, we analyze the incurred noise variance of L_1 distance between H_i and $\tilde{H}_i$ when i is (1) a sampling time point and (2) a non-sampling time point.

LEMMA 5.3. *The noise variance of the L_1 distance between H_j and $\tilde{H}_j$, is $\hat{\sigma}_1 = O(\frac{UC^2}{n^2\epsilon_2^2})$ for a sampling time point, and $\hat{\sigma}_2 = O(\frac{8\Delta^2}{\epsilon_1^2} + \frac{32C^2\Delta^2}{\epsilon_1^2} + \frac{UC^2}{n^2\epsilon_2^2})$ for a non-sampling time point.*

PROOF. We skip this proof due to space limitation. □

THEOREM 5.4. *If we use L_1 distance and LPA, the k value can be obtained as $k = min\{\sqrt[3]{\frac{n^2(8\Delta^2+32C^2\Delta^2)}{UC^2}}, 1 - \frac{C}{N}\}$*

PROOF. Since k is only used when analyzing the distance at non-sampling time points, we can obtain the upper bound of noise variance at a non-sampling time point due to lemma 5.3 with $\epsilon_1 = \frac{k\epsilon}{k+1}$ and $\epsilon_2 = \frac{\epsilon}{k+1}$ by $\hat{\sigma}_2 = \frac{8\Delta^2+32C^2\Delta^2}{\epsilon^2}\frac{(k+1)^2}{k^2} + \frac{UC^2}{n^2\epsilon^2}(k+1)^2$. Let $f(k) = \hat{\sigma}_2$, then the first-order derivative of $f(k)$ is as follows: $\nabla_k f(k) = -\frac{2(k+1)}{k^3}\frac{8\Delta^2+32C^2\Delta^2}{\epsilon^2} + 2(k+1)\frac{UC^2}{n^2\epsilon^2}$. By setting $\nabla_k f(k) = 0$, we can obtain the value of k as: $k = \sqrt[3]{\frac{(8\Delta^2++32C^2\Delta^2)n^2}{UC^2}}$. Since the second-order derivative of $f(k)$ with respect to k is no less than 0, $k = \sqrt[3]{\frac{(8\Delta^2++32C^2\Delta^2)n^2}{UC^2}}$ is the value of k when $f(k)$ arrives at the minimum. Simultaneously, we must require the privacy budget of each sampling time point to be no less than that of each time point in the baseline method, which leads to $\frac{\epsilon_2}{C} \geq \frac{\epsilon}{N}$, and $k \leq 1 - \frac{C}{N}$. Therefore, we can obtain that $k = min\{\sqrt[3]{\frac{n^2(8\Delta^2+32C^2\Delta^2)}{UC^2}}, 1 - \frac{C}{N}\}$. □

6. EXTENSIONS TO INFINITE STREAMS

DSAT under w-event privacy. Algorithm 3 presents DSAT under w-event privacy. For the first w time points, we run DSAT normally and record the privacy budget $\epsilon_{2,i}$ for every time point i, i.e. $\epsilon_{2,i} = \frac{\epsilon_2}{C}$ if i is a sampling point and $\epsilon_{2,i} = 0$ otherwise. For time points $w+1$ to N, if the remaining privacy budget ϵ_{rm} for the current w-window is larger than zero, we compare the distance between H_i and $\tilde{H}_j$, modify the threshold and release a private histogram when the private distance is larger than the threshold; if no privacy budget is left, we skip the current time point and go to the next one.

Privacy guarantee. The first w time points guarantees ϵ-differential privacy. The condition in Line 4 of Algorithm 3 guarantees that if there is no remaining privacy budget (i.e. $\epsilon_{rm} \leq 0$) for the current w window from time point t_{i-1} to t_{i-w+1}, no new private datasets will be released. Therefore, for any w-length window beginning with any time point, at most ϵ privacy budget will be used. This leads to the conclusion that Algorithm 3 satisfies w-event privacy.

7. EXPERIMENT

We implemented our methods on top of two static histogram methods, LPA in Matlab and PSD [7] in Python. All the experiments are performed on a PC with a 2.9GHz CPU and a 8GB memory. Table 2 summarizes the parameters and their default values in the experiments.

Algorithm 3 DSAT under w-event privacy

Input:
$\mathbf{D} = \{D_i | 1 \leq i \leq N, i \in Z\}$, T, C and ϵ.
Output:
$\tilde{\mathbf{D}} = \{\tilde{D}_i | 1 \leq i \leq N, i \in Z\}$
1: Run DSAT for the first w time points;
2: **for** $i = (w+1)$ to N **do**
3: $\epsilon_{rm} = \epsilon_2 - \sum_{m=i-w+1}^{m=i-1} \epsilon_{2,m}$
4: **if** $\epsilon_{rm} \leq 0$ **then**
5: Set $\tilde{D}_i := D_j$, where j is the time point of last release;
6: **else**
7: Set $count = \frac{\sum_{m=i-w+1}^{m=i-1} \epsilon_{2,m}}{\epsilon_2/C}$
8: Compute $\tilde{d}(D_i, \tilde{D}_j) = d(D_i, \tilde{D}_j) + Lap(\frac{2C\Delta}{\epsilon_1})$;
9: Compute $E_i = |\frac{count}{i} - \frac{C}{w}|$, $e_i = \frac{|E_i - \delta|}{\delta}$, and $u_i = \theta \times e_i$;
10: **if** $\frac{count}{i} - \frac{C}{w} \leq 0$, **then** $\tilde{T}_i = max\{0, T_{i-1} - u_i\}$;
11: **else** set $\tilde{T}_i = min\{2, T_{i-1} + u_i\}$;
12: **if** $\tilde{d}(D_i, \tilde{D}_j) \geq \tilde{T}_i$, **then** set $\epsilon_{2,i} = \epsilon_2/C$, $\tilde{D}_i := D_i + < Lap(1/\epsilon_{2,i}) >^U$, and $j = i$;
13: **else** set $\tilde{D}_i := D_j$;
14: **end if**
15: **end for**

Table 2: Experiment Parameters

Parameter	Description	Default value
N	Number of time points	500
d	Number of data dimensions	6
n	Number of tuples in D_i	500K
ϵ	Privacy budget	1.0
C	Cutoff point	$0.01 \times N$
r	Update rate	0.5
δ	Deviation tolerance	0.05
θ	Proportional gain	0.5

7.1 Experiment Setup

Datasets. We conducted our experiments with three datasets: the US census (`http://ipums.org`), the Taxi-Drive trajectory data (`http://research.microsoft.com/apps/`) and the Oldenburg traffic data [5].

The **US census dataset** contains six attributes, *Age*, *Gender*, *Education*, *Health insurance*, *Marital status* and *Income* with 3M tuples and domain sizes of 96, 2, 12, 2, 2, 3. Each tuple represents an individual user. In order to avoid the sparsity of histograms, we convert *Income* into a categorical attribute: values smaller than 0 (mapped to 1), values between 0 and 28K (mapped to 2), and values larger than 28K (mapped to 3). 28K is a median value. Values smaller than 0 means the tuples have ages smaller than 20. The number of histogram bins are the product of the domain sizes of all attributes.

We generate a series of dynamic datasets as follows. D_i is the original dataset at t_i. D_1 has 500K tuples randomly sampled from the original 3M tuples. A public pool is initiated using the remaining tuples. D_i ($i \geq 2$) is obtained by deleting m tuples from D_{i-1} while inserting m tuples randomly selected from the public pool to simulate the user updates. m is sampled from $N(\mu, \sigma^2)$, where μ is $\frac{r \times |D_{i-1}|}{2}$, and σ^2 is set to $100K$. Here, r is the *update rate*, $|D_i|$ is the data cardinality of D_i and datasets at all time points have the same data cardinality. The time points are partitioned into 10 periods with different values of m to simulate varying update patterns. All experiments use US census data by default since we can generate various datasets under different parameter settings.

The **Taxi trajectory dataset** has a one_week trajectories of $10,357$ taxis during the period of Feb. 2 to Feb. 8, 2008 within Beijing. We

transfer the time dimension to 168 time points with 24×7. The total number of points in this dataset is about 15 million and the total distance of the trajectories reaches 9 million kilometers. We partition the longitude and latitude into 10×10 grids. We amplify the number of taxis to $110,357$ by sampling dummy points on extremely sparse time points and geographical areas while still keeping the patterns of original data.

We generated **Oldenburg traffic data** with the Brinkhoff generator [5]. The input of the generator is the road map of Oldenburg in Germany, and the output is a set of moving objects on the road network. We created the data set with 1000 discrete timestamps, with 500,000 objects at the beginning. A 2D grid with 1024×1024 cells is used to record the locations of the moving objects.

Comparison. We evaluate the utility of the private DP histograms of dynamic datasets by answering random range count queries. The query accuracy of DSAT is compared with three solutions described in Section 3: the baseline Laplace mechanism, the fixed-sampling method, and the state-of-the-art w-event privacy methods. LPA and PSD [7] are used to generate DP histograms at sampling time points. We note that our proposed sampling framework can utilize any state-of-the-art static histogram method at each sampling point. Here we just use, as an example, the standard LPA method as well as the PSD method [7] which is a state-of-the-art static histogram method that uses spatial partitioning. The goal is to compare our proposed methods and the three solutions. We also include the non-private methods to compare the update errors of DSAT and fixed-sampling.

Metrics. For the US census dataset, we generated random range-count queries with random query predicates on each attribute defined in the SQL format as "Select COUNT(*) from D, Where $A_1 \in I_1$ and $A_2 \in I_2$ and. . .and $A_m \in I_m$". I_i is a random interval generated from the domain of attribute A_i. For the traffic data, query rectangles with various sizes are randomly generated. In each experiment run, 5000 random queries are generated and the average absolute error over 10 runs is reported, which is defined as $E^a = \frac{1}{M \times N} \sum_{k=1}^{M} \sum_{i=1}^{N} |\tilde{A}_i^k - A_i^k|$, A_i^k is the true answer and $\tilde{A}_i^k$ is the noisy answer. Here we use the range-count query to measure the utility since it composes data histograms, and the range counts can be used for many significant mining tasks, e.g. dynamic stream clustering, outlier detection of time-series data, etc.

7.2 Results on user level privacy

In all experiments, we compare our methods with the baseline and fixed-sampling methods, which are denoted by "baseline" and "fixed" in figures. Unless specified, we use LPA by default as the underlying histogram method for the sampling point. We also use "DSAT-true" and "fixed-true" to denote the non-private versions of DSAT and fixed-sampling.

Absolute error vs. k. Figure 2 investigates how utility changes with various k values, which specify the budget allocation ratio between ϵ_1 and ϵ_2 for the decision and sampling stages respectively. With the value of C being 10, we compute k to be 0.0532 due to theorem 5.4. From Figure 2, we can observe that the empirical result matches the theoretical result well and the utility reaches the optimal value with k between 0.01 and 0.1. The error increases as k becomes larger or smaller than 0.1 or 0.01, respectively. This is reasonable because larger k may lead to more perturbation error while smaller k values result in more update error.

DSFT and DSAT. In this experiment, we compare our proposed two methods DSFT and DSAT. From figure 3, we can observe that the error of DSFT is very sensitive to the threshold value T. As T initially increases, the error decreases thanks to the decreased perturbation error. As T further increases, the error increases back up due to the increased sampling error which becomes the dominant error. Without prior knowledge, it is difficult to determine the optimal T. However, the average absolute error of DSAT is close to the lowest error of DSFT with the optimal threshold T value being around 0.025. Here the initial value of T for DSAT can be arbitrarily selected. Thus, the DSAT method with the PID control can effectively adjust T to an optimal one. In the remaining experiments, we only use DSAT to compare with other methods.

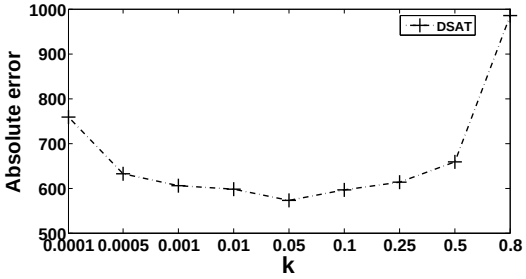


Figure 2: Absolute error vs. k

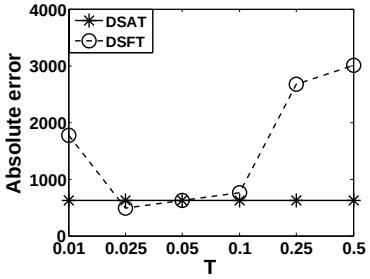
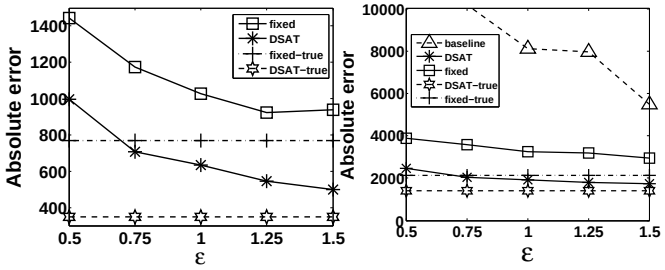


Figure 3: Absolute error vs. threshold value T

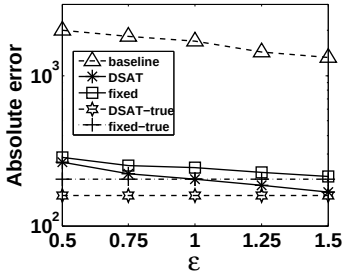
Absolute error vs. differential privacy. Figure 4 compares DSAT with other methods under various privacy budgets. The larger the privacy budget is, the closer the query accuracy is to non-private versions. Since the baseline performs one order of magnitude worse than other methods in most experiments, we do not include them for better readability of the graphs. The perturbation errors for fixed-sampling and DSAT are almost similar as the number of released DP histograms are the same. DSAT outperforms fixed-sampling because DSAT has much less update error, which can be seen from the comparison of non-private versions. Figure 4(b) uses the taxi trajectory dataset and Figure 4(c) uses PSD to release DP histograms with 3D US data. We can see that by using PSD, errors are generally improved compared to the ones using LPA. This further confirms that our methods can take advantage of any state-of-the-art static histogram methods for each sampling point.

Absolute error vs. update rate. We study the impact of the update rate r (defined in section 7.1) on the query accuracy for different methods, as shown in Figure 5. All methods remain stable for various update rates. The DSAT performs better than both non-private and private fixed-sampling methods. This is because the update error of non-private DSAT is much less than non-private fixed-sampling. This further verifies that our DSAT with PID controller succeeds in adaptively adjusting the threshold and the location of the sampling time point, leading to better performance.

Absolute error vs. dimensionality. Figure 6 examines the absolute error with various numbers of dimensions in the US dataset. DSAT again outperforms both non-private and private fixed-sampling



(a) US data (b) Taxi-drive data



(c) US data (log-scale) with PSD

Figure 4: Accuracy vs. differential privacy budget

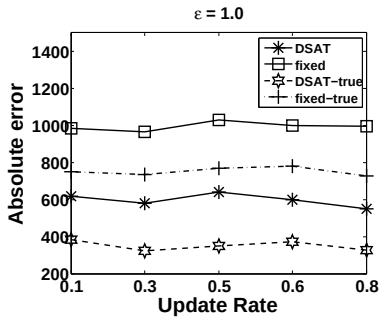


Figure 5: Absolute error vs. update rate

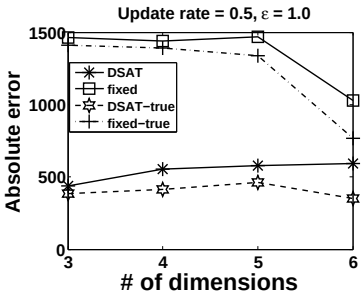
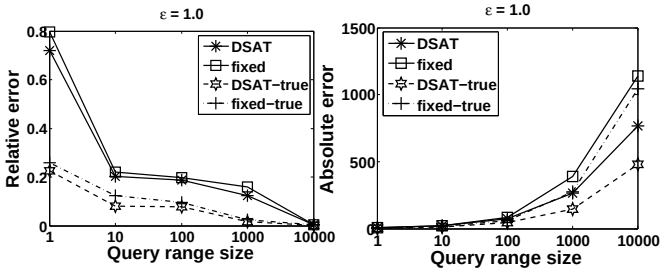


Figure 6: Absolute error vs. # of dimensions

methods with the dimensionality from 3 to 6. One interesting phenomena we observe is that the performances of non-private and private fixed-sampling methods improve sharply after five dimensions. This can be explained by the fact that a higher dimensionality results in a larger number of histogram bins. Given a threshold T, if the L_1 distance between two datasets D_i and D_{i-1} is below T, the previously released histogram will be used which incurs an update error. Given the same L_1 distance between two histograms, a larger number of bins would result in a smaller measured update error since the average difference for each histogram bin is smaller. Hence the fixed sampling methods show a dramatic drop in the error which is dominated by the update error. The DSAT methods are less sensitive to the number of dimensions because they already mitigate the update error by tuning the threshold adaptively. Hence the non-private DSAT shows a slight drop in the update error while the private DSAT shows a slight increase due to the dominating perturbation error.

Query accuracy vs. query range size. We study the impact of the query range size on the query accuracy for different methods. For each query range size, we randomly generated queries such that the product of the query ranges on each dimension equals the given size. Figure 7 presents the impact of various query range sizes on query accuracy in terms of relative error and absolute error. The relative error is defined as $E^r = \frac{1}{M \times N} \sum_{k=1}^{M} \sum_{i=1}^{N} \frac{|\tilde{A}_i^k - A_i^k|}{max(s, A_i^k)}$, where s is the sanity bound to mitigate the effect for $A_i^k = 0$. DSAT outperforms the private fixed-sampling method. The difference of relative errors between all methods is not obvious because of the large data cardinality in the US data. For all methods, the relative error gradually degrades as the query range size increases while the absolute error has the opposite trend. The reason is that when the query size is small, the true answer A_i^k is also small which may incur a small absolute error but large relative error. In this experiment, the sanity bound s is set to 1.



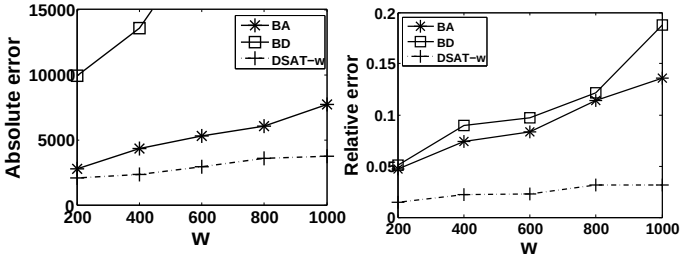
(a) Relative error (b) Absolute error

Figure 7: Query accuracy vs. query range size

7.3 Results on w-event privacy

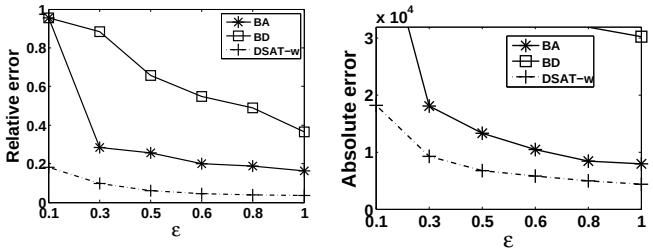
Query accuracy vs. parameter w. We use the Oldenburg traffic data in this experiment, since it contains 1000 timestamps that is sufficient to investigate the impact of w. We compare DSAT with BD and BA in [18] under w-event privacy framework while varying w values. BD and BA are implemented by using column partitioning technique and setting $\epsilon_1 = \frac{\epsilon}{U}$ as recommended in [18]. From figure 8, we can see that the gap between DSAT and BD or BA expands greatly as w increases. This is because our technique adaptively adjusts the threshold and allocates the privacy budget more appropriately. In contrast, BA and BD may not fully utilize or in advance exhaust most budget during w timestamps.

Query accuracy vs. differential privacy. In this experiment, we set w to be 800 using Oldenburg traffic data with 1000 timestamps. Figure 9 compares DSAT with BA and BD under various privacy budgets. We can see that BA degrades dramatically and the gap between BA and DSAT greatly expands as we reduce the privacy budget ϵ. This is because BA starts by uniformly distributing the budget to all w timestamps, and more perturbation error will be incurred when ϵ is small and w is large. Our DSAT performs well since the perturbation error of released datasets depends only on C.



(a) Absolute error (b) Relative error

Figure 8: Query accuracy vs. w



(a) Relative error (b) Absolute error

Figure 9: Query accuracy vs. differential privacy

8. CONCLUSIONS

In this paper, we have proposed an adaptive distance-based sampling approach to address the challenges of releasing a series of differentially private dynamic datasets in real time. With an upper bound to limit the number of DP data releases, our methods incur much smaller errors. We apply an adaptive control mechanism to dynamically adjust the threshold value. We also provide privacy and utility analysis for our method. Experiments on real and synthetic datasets show that our algorithm outperforms the baseline and existing state-of-the-art techniques. As future work, we would like to study update models and incorporate them into our sampling framework. We are also interested in applying the adaptive sampling framework for releasing other types of dynamic data with differential privacy, e.g. frequent patterns for dynamically changing transactional data and dynamic graph patterns in social networks.

Acknowledgment

This work is supported by the National Institute of Health (NIH) under award number R01GM114612, the Patient-Centered Outcomes Research Institute (PCORI) under award number ME-1310-07058, and the National Science Foundation (NSF) under award number 1117763, and partly supported by NLM (R00LM011392), NLM (R21LM012060), and NHLBI (U54HL108460). The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

9. REFERENCES

[1] G. Ács, C. Castelluccia, and R. Chen. Differentially private histogram publishing through lossy compression. In *ICDM*, 2012.

[2] K. H. Ang, G. Chong, and Y. Li. Pid control system analysis, design, and technology. *IEEE Trans. Contr. Sys. Techn.*, 13(4):559–576, 2005.

[3] K. L. Avrim Blum and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, 2008.

[4] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *PODS*, 2007.

[5] T. Brinkhoff. A framework for generating network-based moving objects. In *Geoinformatica*, pages 6(2), 153Ð180, 2002.

[6] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26, 2011.

[7] G. Cormode, C. M. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, 2012.

[8] G. Cormode, C. M. Procopiuc, D. Srivastava, and T. T. L. Tran. Differentially private summaries for sparse data. In *ICDT*, pages 299–311, 2012.

[9] C. Dwork. Differential privacy. *Automata, Languages and Programming, Pt 2*, 4052, 2006.

[10] C. Dwork. A firm foundation for private data analysis. *Commun. ACM.*, 2011.

[11] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating Noise to Sensitivity in Private Data Analysis. *Theory of Cryptography*, pages 1–20.

[12] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

[13] L. Fan, L. Bonomi, L. Xiong, and V. Sunderam. Monitoring web browsing behaviors with differential privacy. In *WWW Conference*, 2014.

[14] L. Fan and L. Xiong. An adaptive approach to real-time aggregate monitoring with differential privacy. *IEEE TKDE*, 26(9):2094–2106, 2014.

[15] M. Hardt and G. N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70, 2010.

[16] M. Hayy, V. Rastogiz, G. Miklauy, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. *VLDB*, 2010.

[17] G. Kellaris and S. Papadopoulos. Practical differential privacy via grouping and smoothing. In *VLDB*, 2013.

[18] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias. Differentially private event sequences over infinite streams. *PVLDB*, 7(12):1155–1166, 2014.

[19] J. Lee and C. W. Clifton. Top-k frequent itemsets via differentially private fp-trees. In *SigKDD, 2014*, pages 931–940, 2014.

[20] H. Li, L. Xiong, and X. Jiang. Differentially private synthesization of multi-dimensional data using copula functions. In *EDBT*, pages 475–486, 2014.

[21] H. Li, L. Xiong, L. Zhang, and X. Jiang. Dpsynthesizer: Differentially private data synthesizer for privacy preserving data sharing. *PVLDB*, 7(13):1677–1680, 2014.

[22] McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD*, New York, NY, USA, 2009. ACM.

[23] W. H. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *ICDE*, 2013.

[24] W. H. Qardaji, W. Yang, and N. Li. Understanding hierarchical methods for differentially private histograms. *PVLDB*, 6(14):1954–1965, 2013.

[25] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD*, pages 735–746, 2010.

[26] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, pages 225–236, 2010.

[27] Y. Xiao and L. Xiong. Protecting locations with differential privacy under temporal correlations. In *CCS*, 2015.

[28] Y. Xiao, L. Xiong, L. Fan, S. Goryczka, and H. Li. Dpcube: Differentially private histogram release through multidimensional partitioning. *Transactions on Data Privacy*, 7(3):195–222, 2014.

[29] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett. Differentially private histogram publication. *VLDB J.*, 2013.

[30] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: private data release via bayesian networks. In *SIGMOD*, pages 1423–1434, 2014.

[31] X. Zhang, X. Meng, and R. Chen. Differentially private set-valued data release against incremental updates. In *DASFAA (1)*, 2013.