Adaptively Sharing Time-Series with Differential Privacy

Liyue Fan
Math&CS Dept. Emory University
Atlanta, GA
liyue.fan@emory.edu

Li Xiong
Math&CS Dept. Emory University
Atlanta, GA
lxiong@mathcs.emory.edu

ABSTRACT
Sharing real-time aggregate statistics of private data has given much benefit to the public to perform data mining for understanding important phenomena, such as Influenza outbreaks and traffic congestions. We propose an adaptive approach with sampling and estimation to release aggregated time series under differential privacy, the key innovation of which is that we utilize feedback loops based on observed (perturbed) values to dynamically adjust the estimation model as well as the sampling rate. To minimize the overall privacy cost, our solution uses the PID controller to adaptively sample long time-series according to detected data dynamics. To improve the accuracy of data release per timestamp, the Kalman filter is used to predict data values at non-sampling points and to estimate true values from perturbed query answers at sampling points. Our experiments with three real data sets show that it is beneficial to incorporate feedback into both the estimation model and the sampling process. The results confirmed that our adaptive approach improves accuracy of time-series release and has excellent performance even under very small privacy cost.

1. INTRODUCTION
Sharing real-time aggregate statistics of private data has given much benefit to the public to perform data mining for understanding important phenomena. Consider the following examples of data aggregation and mining applications:

Disease Surveillance A health care provider, such as an Emergency Department, gathers data from individual visitors. The collected data, e.g., daily number of Influenza cases, is then shared with a third party, for instance, researchers, in order to monitor and to detect possible seasonal epidemic outbreaks at the earliest.

Traffic Monitoring A GPS service provider gathers data from a set of individual users about their locations, speeds, mobility, etc. The aggregated data, for instance, the number of users at each region during each time period, can be mined for commercial interest, such as popular places, as well as public interests, such as congestion patterns in roads.

In general, such aggregate data sharing applications have a similar scenario as shown in Figure 1. In this scenario, a central trusted component gathers data from a large number of individual subscribers. The collected data may be then aggregated and continuously shared with other untrusted entities for various purposes. The trusted server, i.e., publisher, is assumed to be bound by contractual obligations to protect the user’s interests, therefore it 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 the publisher to share useful aggregate statistics over individual users continuously (aggregate time series) while guaranteeing their privacy.

The current state-of-the-art paradigm for privacy-preserving data publishing is differential privacy. Differential privacy requires that the aggregate statistics reported by a data publisher be perturbed by a randomized algorithm \( A \), so that the output of \( A \) remains roughly the same even if any single tuple in the input data is arbitrarily modified. This ensures that given the output of \( A \), an adversary will not be able to infer much about any single tuple in the input, and thus privacy is protected.

Most existing work on differentially private data release deal with one-time release of static data \[8,9,11,23,25,26\]. In the applications we consider, high-volume data are acquired dynamically. In the aggregate time series, the data...
values at successive timestamps can be highly correlated, which makes solutions designed for static data problematic. A standard differential privacy mechanism can be applied to perturb the value at each timestamp. Due to the correlation between the values and the composition theorems of differential privacy [19], it can lead to an overall perturbation error of $\Theta(T)$, where $T$ is the length of the time series, which severely limits the utility of the published data if $T$ is very large.

Few recent works [5, 9, 23] studied the problem of releasing time series or continual statistics. Rastogi and Nath [23] proposed an algorithm which perturbs the Discrete Fourier Transform (DFT) of the entire time series and reconstructs a released version from the Inverse DFT. Since the entire time-series is required to perform those operations, the timeliness of publishing is greatly impacted, limiting its applicability for real-time disease surveillance and traffic monitoring applications. Moreover, this solution also suffers reconstruction error when calculating the Inverse DFT to recover the original time-series.

Dwork et al. [8] proposed a differentially private continual counter over a binary stream with a bounded error at each time step $k$ being $O\left(\frac{1}{\alpha}(\log k)^{1.5}\right)$ where $\alpha$ is the degree of differential privacy provided. Chan et al. [5] studied the same problem and concluded with a similar upper bound. Both works adopt an event-level privacy model, with the perturbation mechanism designed to protect the presence of an individual event, i.e. a user’s contribution to the data stream at a single time point, rather than the presence or privacy of a user.

**Our Contributions.** In this paper, we propose a novel adaptive approach with sampling and estimation for releasing time series under differential privacy. It uses sampling to query and perturb selected values in the time series with the differential privacy mechanism, and simultaneously uses prediction and estimation to dynamically predict the non-sampled values and correct the sampled values. Applying perturbation only to sampled values reduces the overall perturbation noise under a given differential privacy constraint. The prediction and estimation aims to reduce the prediction error at non-sampled points to reduce the impact of the perturbation noise at sampled points. The key innovation is that it utilizes feedback loops based on observed (perturbed) values to dynamically adjust the prediction/estimation model and the sampling rate. To this end, we examine two challenges in our system: predictability and controllability. The former raises the question: given a perturbed observation at each time point, can we formulate an estimate which is close to the true value and dynamically adjust the estimation model based on current observation? The latter imposes another question: suppose an accurate estimate can be derived at any time step, can we dynamically adjust the sampling rate according to the rate of data change? We propose a solution to address these two issues and we summarize our contributions below.

1) To improve the accuracy of data release at each timestamp, we propose to use the Kalman filter [13] which is widely adopted for signal recovery, to estimate the original data values based on observed values. By assuming a process model that generates the time series, one advantage of the Kalman filter is that it reduces the impact of perturbation errors introduced by differential privacy mechanism. This is achieved by linearly combining a prediction generated by the process model and the perturbed observation. The combined value, referred to as *aposteriori* estimate, is a minimum variance estimate, which provides an educated guess rather than a pure perturbed value. The estimate is then fed back to the system for future predictions and for dynamically adjusting the sampling process.

2) To minimize the overall privacy cost, hence, the overall perturbation error, we propose an adaptive sampling algorithm which adjusts the sampling rate using a PID controller. Without assuming a model for the sampling process, a PID controller, which is the most common form of feedback controller, is in place to detect data dynamics from estimates by the Kalman filter and to increase sampling frequency when data is going through rapid changes. Figure 2 illustrates the idea of adaptive sampling. We plot the original time-series, traffic count, as well as the number of queries issued by the adaptive sampling mechanism during each corresponding time unit. As is shown, the number of queries issued by our adaptive approach increases between day 50 and day 100, when the traffic count exhibits significant fluctuations, and it drops beyond day 100, when there’s little variation among the original data values.

3) We empirically study the accuracy and robustness of our approach with real time-series data sets. Our experiments show the proposed solution provides real-time accurate results and stability despite different data dynamics. We believe our solution is applicable to a wider range of applications.

The rest of the paper is organized as follows: Section
2. PRELIMINARIES

In this section we will discuss problem setup, differential privacy that we use as our privacy definition, and also review existing perturbation techniques to achieve differential privacy on time-series data.

2.1 Problem Statement

We consider time-series data consisting of aggregate values from a set of individuals (such as people visiting a particular hospital). Formally we define a time series \( X \) as follows:

Definition 1. [Aggregate Time Series] A univariate, discrete time series \( X = \{x_k\} \) is a set of values of a variable \( x \) observed at discrete time \( k \), with \( 0 \leq k < T \), where \( T \) gives the lifetime of the series.

In suggested applications, \( X \) is an aggregate count series, such as, the daily total of patients diagnosed of Influenza, or the hourly count of drivers passing by a gas station. This assumption will hold true for all baseline algorithms as well as our proposed approach.

A trusted aggregator who has access to the entire time-series, when sharing it with others, needs to release a high quality version from the original series, yet without compromising the privacy of any individual participant. We measure the quality of a published series by average relative error:

Definition 2. [Average Relative Error] The average relative error, denoted by \( E \), of a published series \( R = \{r_k\} \) derived from original time-series \( \{x_k\} \) is given by the following equation:

\[
E = \frac{1}{T} \sum_{k=0}^{T-1} \frac{|r_k - x_k|}{\max\{x_k, \delta\}}
\] (1)

where \( \delta \) is a user-specified constant (also referred to as sanitary bound as in \cite{25}) to mitigate the effect of excessively small query results. Here we assume that the sanitary bound remains same throughout the entire time-series.

Clearly, the quality of a published series increases as each \( r_k \) approaches \( x_k \), the extreme case of which would have \( r_k = x_k \), for each \( k \). However, a privacy-preserving release is likely to perturb original data values in order to protect individual privacy. Thus, a publishing mechanism that guarantees user privacy and yields high utility is desired.

2.2 Differential Privacy and Background

Informally, a mechanism is differentially private if its outcome is not significantly affected by the removal or addition of a single user. It ensures a user that any privacy breach will not be a result of participating in the database since anything that is learnable from the database with his record is also learnable from the one without his record.

The formal definition of differential privacy \cite{2}, also referred to as Unbounded Differential Privacy by \cite{14}, is given as follows. Here the parameter, \( \alpha \), specifies the degree of privacy offered.

Definition 3. [Differential Privacy] A non-interactive privacy mechanism \( A \) gives \( \alpha \)-differential privacy if for any dataset \( D_1 \) and \( D_2 \) differing on at most one record, and for any possible anonymized dataset \( \tilde{D} \in \text{Range}(A) \),

\[
\Pr[A(D_1) = \tilde{D}] \leq e^{\alpha} \times \Pr[A(D_2) = \tilde{D}]
\] (2)

where the probability is taken over the randomness of \( A \).

Laplace Mechanism. Dwork et al. \cite{8} show that differential privacy can be achieved by adding i.i.d. noise to the result of each query. The magnitude of the noise added conforms to a Laplace distribution with the probability density function \( p(x) = \frac{1}{2\lambda} e^{-|x|/\lambda} \), where \( \lambda \) is determined by both the desired privacy level \( \alpha \) and the global sensitivity \( \lambda \) of a query which is defined below.

Definition 4. [Global Sensitivity] For any function \( f : D \rightarrow \mathbb{R}^d \), the Global Sensitivity of \( f \) is

\[
GS(f) = \max_{D_1, D_2} ||f(D_1) - f(D_2)||
\] (3)

for all \( D_1, D_2 \) differing in at most one record. For instance, the sensitivity of count query is 1.

Dwork et al. \cite{8} prove that adding Laplace noise of magnitude \( \lambda = GS(q) / \alpha \) to the true answer of query \( q \) guarantees \( \alpha \)-differential privacy.

Composition. The composition properties of differential privacy provide privacy guarantees for a sequence of computations. Any sequence of computations that each provides differential privacy in isolation also provides differential privacy in sequence, which is known as sequential composition \cite{19}.

Theorem 1. \cite{19} Let \( A_i \) each provide \( \alpha_i \)-differential privacy. A sequence of \( A_i(D) \) over the dataset \( D \) provides \( (\sum_i \alpha_i) \)-differential privacy.
Algorithm 1 Laplace Perturbation Algorithm (LPA)

Input: Raw time-series $X$; privacy budget $\alpha$
Output: Released time-series $R$

1: for each $i \in [0, 1, ..., T - 1]$ do
2: draw noise from $\text{Lap}(T/\alpha)$;
3: $r_k = x_k + \text{noise}$;
4: return $R$.

In a special case called parallel composition [19], the computations operate on disjoint subsets of the data $D$. The ultimate privacy guarantee depends only on the worst guarantee among individual computations, rather than the sum. However, it is not applicable in our context since the data $D$ at successive timestamps can be highly correlated.

Given the sequential composition, an overall privacy requirement $\alpha$ can be considered as a privacy budget, and needs to be allocated among all the queries in a data release mechanism in order to guarantee $\alpha$-differential privacy of the released data.

User-level privacy vs. event-level privacy. The work [9] proposed a differentially private continual counter with the notion of event-level privacy, where the neighboring databases differ at $u_i$, a user $u$’s contribution at timestamp $i$. In our study, we provide a stronger privacy guarantee, user-level privacy, where the neighboring databases differ at the user $u$, i.e. $u$’s contribution at all timestamps, thus protecting sensitive information about user $u$ at any time.

2.3 Existing Solutions

Here we discuss baseline Laplace perturbation algorithm and a recently proposed DFT algorithm. Empirical studies of them comparing to our proposed solution are included in Section 6.

Laplace Perturbation Algorithm. The standard Laplace mechanism can be applied to perturb data values at each timestamp. In other words, $T$ recurring queries, assuming count queries, can be issued at each timestamp. Due to the composition theorem, if each individual query is $\alpha/T$-differentially private, the sequence of queries guarantees $\alpha$-differential privacy, which leads to a total perturbation error of the magnitude of $\theta(T)$. Figure 3(a) illustrates the workflow of this naive Laplace Perturbation Algorithm. The detailed algorithm is shown in Algorithm 1.

Discrete Fourier Transform. Rastogi and Nath [23] proposed the Fourier Perturbation Algorithm $\text{FPA}_k$ that transforms the raw series into frequency domain, perturbs the first $k$ coefficients, and then reconstructs the series with the perturbed $k$ coefficients. Figure 3(b) illustrates the main idea of their method. The outline is shown in Algorithm 2. It begins by computing $F^k$, which is composed of the first $k$ Fourier coefficients in the Discrete Fourier Transform (DFT) of $X$, with the $j^{th}$ coefficient is given as: $DFT(X)_j = \sum_{i=0}^{T-1} e^{2\pi i k j/T} x_i$. Then it perturbs $F^k$ using LPA algorithm with privacy budget $\alpha$, getting a noisy estimate $\tilde{F}^k$. This perturbation is to guarantee differential privacy. Denote $\text{PAD}^T(\tilde{F}^k)$ the sequence of length $T$ by appending $T-k$ zeros to $\tilde{F}^k$. The algorithm finally computes the Inverse Discrete Fourier Transform (IDFT) of $\text{PAD}^T(\tilde{F}^k)$ to get $R$. The $j^{th}$ element of the inverse is given as: $\text{IDFT}(X)_j = \frac{1}{T} \sum_{i=0}^{T-1} e^{-2\pi i j/T} x_i$.

3. OVERVIEW OF OUR SOLUTION

In this section, we propose a novel solution to sharing time-series data with differential privacy. It allows for fully automated adaptation to changing data dynamics and highly accurate time-series prediction/estimation.

3.1 Framework

It is intuitive that a good adaptation scheme is likely to issue more queries to the original time-series when data value is going through rapid variations and fewer queries when the data curve is relatively flat. Furthermore, this information of data trend can be inferred from historical queries at no extra cost. Therefore, we propose a feedback control solution which utilizes the past query results to make decisions about the future sampling strategy. In addition, a Kalman filter is used to predict the data values at non-sampling points and correct the predictions based on the observed (perturbed) values at sampling points.

The framework of our solution is shown in Figure 4. In a feedback loop presentation, the framework is composed of input, perturbation mechanism, estimation based on Kalman filter, adaptive sampling based on PID controller, and output.

- The input is a streaming time-series with one aggregated value at a time. It is sampled by the PID controller. As a result, not every data value is queried from the stream.

- The Laplace mechanism perturbs each data value that actually is sampled by the system in order to guarantee differential privacy.
where $M$ minimizes the sampling interval with a predefined minimum value $T$. For each time step after the first $T$, the error, between the prior estimate and the posterior estimate, is introduced by the Kalman filter. At each timestamp, the Kalman filter makes an forward prediction and then uses feedback control: the filter estimates the process state at sampling points and at non-sampling points, both generated by the Kalman filter. The error, between the prior estimate and the posterior estimate, is then fed through the adaptive sampling module with PID controller to determine a new sampling rate.

The output is a streaming time-series with posterior estimates at sampling points and apriori estimates at non-sampling points, both generated by the Kalman filter.

There are two types of error which we would like to balance in our solution: perturbation error and prediction error. The perturbation error is introduced by Laplace perturbation mechanism at sampling points, while the prediction error is introduced by the Kalman filter Prediction procedure at non-sampling points. Clearly, the more we sample, the larger perturbation error is introduced, but the prediction error is reduced due to increased feedback and vice versa. The baseline Laplace perturbation algorithm Algorithm 1 introduces high perturbation error by querying at every timestamp. The DFT algorithm Algorithm 2 mitigates the perturbation error at the cost of reconstruction error when recovering the time-series from the Inverse DFT. In contrast, our goal is to balance the trade-off between these two types of error by adaptively adjusting the sampling rate.

### 3.2 Algorithm

Here we present an overall description for our solution in Algorithm 3. We prove that it is $\alpha$-differentially private. More details about Kalman filter and PID controller will be further discussed in the next two sections.

Queries will be issued through the Laplace mechanism for the first $T$ timestamps to collect enough feedback for the PID controller (Line 3-6 in Algorithm 3). Line 7 initializes the sampling interval with a predefined minimum value $minI$. For each time step after the first $T$, a prior prediction is generated by KFPredict procedure.

The feedback control loop is implemented by Line 8 to 17. If the current time step is a sampling point ($k \% interval = 0$) and there’s budget left for more queries ($queryCount < M$ where $M$ is a predefined bound for maximum number of queries allowed), a noisy measurement is retrieved from Laplace perturbation (Line 11), and it is used by KFCorrect to obtain an updated estimate posterior. Line 13 publishes the posterior. A new interval is determined by the PIDControl output from Line 14. Note we always keep the interval above the minimum length. If either condition in Line 10 evaluates to false, Line 17 publishes the prediction prior to save the privacy budget.

**THEOREM 2.** Algorithm 3 satisfies $\alpha$-differential privacy.

PROOF. Each query issued through the Laplace mechanism is $\alpha/M$-differentially private. The maximum number of queries allowed is $M$. By the composition rule (Theorem 1), Algorithm 3 satisfies $\alpha$-differential privacy.

### 4. ESTIMATION

In this section, we present the formal Kalman filter model we adopt for estimating data values at each timestamp and describe the details of the Kalman filter prediction and correction procedures.

#### 4.1 The Kalman Filter Model

The Kalman filter was introduced in 1960 by R. E. Kalman as a recursive solution to the discrete data linear filtering problem. Since then, it has found application in the fields of data smoothing, process estimation, and object tracing, to name a few. The Kalman filter addresses the general problem of trying to estimate the internal state of a discrete-time controlled process that is governed by a linear stochastic difference equation. Since the internal state of a process is usually unavailable, the Kalman filter estimates the process using feedback control: the filter estimates the process state according to the linear equation at some time and it obtains feedback in the form of (noisy) measurements. It aligns perfectly with our time-series sharing scenario where only perturbed data values are available. We are then inspired to model the time-series data with a process model, and
to treat observations out from Laplace perturbation mechanism as noisy measurements. We now introduce the formal Kalman filter model in our context.

**Process Model.** In the time-series publishing scenario, we adopt a constant process model for the time-series which is given by the following equation:

\[ x_{k+1} = x_k + \omega \]  

(4)

where \( k \) is the discrete time index. This constant system model states that adjacent data values from the original time-series should be consistent except for a white Gaussian process model noise \( \omega \):

\[ p(\omega) \sim N(0,Q) \]  

(5)

where \( Q \) is the covariance of \( \omega \).

**Measurement Model.** The observation, obtained by Line 11 in Algorithm 3, is perturbed data out from the Laplace mechanism, and can be modeled by:

\[ z_k = x_k + \nu \]  

(6)

where \( \nu \), the measurement noise, is a Laplacian noise which follows:

\[ p(\nu) \sim \text{Lap}(0,\lambda) \]  

(7)

where \( \lambda \) is the magnitude parameter determined by differential privacy mechanism.

For computational efficiency and by the guidance of common practice, we will use a small, white Gaussian error to approximate the actual noise distribution \( \text{Lap}(\lambda) \). Thus we define the distribution of \( \nu \) as

\[ p(\nu) \sim N(0,R) \]  

(8)

where \( R \) is the covariance of \( \nu \).

**Prior and Posterior Estimates.** At time step \( k \), the *apriori* state estimate \( \hat{x}_k^- \) is made based on the system model (4) and is related to the *aposteriori* state estimate of last step:

\[ \hat{x}_k^- = \hat{x}_{k-1}. \]  

(9)

The *aposteriori* state estimate \( \hat{x}_k \) is based on a linear combination of *apriori* state estimate \( \hat{x}_k^- \) and a weighted prediction error \( \psi_k \). The *aposteriori* estimate is calculated as follows:

\[ \hat{x}_k = \hat{x}_k^- + K_k \psi_k. \]  

(10)

This error \( \psi_k \) is called *innovation*, which captures the difference between actual measurement and measurement prediction. It is given as

\[ \psi_k = z_k - \hat{x}_k^- \]  

(11)

The value of the weight \( K_k \) is called Kalman Gain which is adjusted with each measurement. In order to reduce uncertainty of the *aposteriori* estimate \( \hat{x}_k \), \( K_k \) is chosen to minimize the *aposteriori* error covariance \( P_k \), which is defined as

\[ P_k = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]. \]  

(12)

Symmetrically, we define the *apriori* error covariance \( P_k^- \) as

\[ P_k^- = E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T]. \]  

(13)

![Figure 5: A complete picture of the Kalman filter](image)

Applying the least-square method to (12), we get

\[ K_k = P_k^- (P_k^- + R)^{-1}, \]  

(14)

By (10,12,13), we get

\[ P_k = (1 - K_k)P_k^- \]  

(15)

The projection of the *apriori* error covariance \( P_k^- \) can be derived given (4,9,12,13), and \( P_{k-1} \), resulting

\[ P_k^- = P_{k-1} + Q. \]  

(16)

### 4.2 Prediction and Correction

The sheer advantage of the Kalman filter lies in the fact that it maintains and updates the best estimate of the internal state by properly weighing and combining all available data (state prior and noisy observations by the differential privacy mechanism) to form an educated guess. It accomplishes that by repeating the following two mechanisms [23]:

- **Prediction:** This mechanism projects forward in time the current state and error estimates to obtain the *apriori* estimates for the next step. At time step \( k \), the filter predicts the values of the internal state and the error covariance at time step \( k + 1 \).
- **Correction:** This mechanism is responsible for the feedback, i.e., for incorporating a new measurement into the *apriori* estimate to obtain an improved *aposteriori* estimate. At time step \( k + 1 \) when an actual measurement is available, the filter corrects itself based on the innovation.

Figure 3 gives a high-level diagram of the two operations of the Kalman filter. After each prediction and correction pair, the process is repeated with the previous *aposteriori* estimate \( \hat{x}_{k-1} \) used to project or predict the new *apriori* estimate. The *prediction* mechanism is implemented by KFPredict in Algorithm 4. Note that we always derive *apriori* state estimate using the most recent published value \( r_{k-1} \). The *correction* mechanism is implemented by KFCorrect in Algorithm 5. Note that we obtain measurement \( z_k \) as the noisy observation from Laplace perturbation mechanism.

Since each noisy observation from Laplace mechanism comes with a cost (privacy budget spent), we are motivated to sample data values through the differential privacy interface only when needed in our overall solution. As a result, a noisy observation, which is crucial to the *correction* step, may not be available at all times. Therefore, we propose to release the prior estimate when the observation is absent and to correct the released value when the observation is available.

The detailed sampling strategy is described in next section.
5. SAMPLING STRATEGY

In this section, we present our adaptive sampling strategy for issuing queries. To motivate and facilitate our discussion, we first present a fixed-rate sampling strategy, and then describe the details of the adaptive sampling strategy based on PID control which aims to achieve near optimal sampling interval based on the data dynamics.

5.1 Fixed Rate Sampling

A naive approach is to use fixed-rate sampling and to predict at non-query points. Given an interval \( I \), the fixed-rate algorithm samples the time series periodically and publishes the perturbed value per \( I \) time units. As for the time points between two adjacent queries, an estimate of the data value is published.

Algorithm 4 shows a sketch of the algorithm. The variable \( M \) represents the total number of queries allowed and the individual budget for each query is equivalent to \( \alpha/M \). Line 10 indicates that for the non-sampling points, we use the prior estimate given by the Kalman filter.

A sampling strategy making \( M \) queries in total leads to an overall perturbation error of \( \Theta(M) \), rather than \( \Theta(T) \) (\( T > M \)) by the baseline Laplace perturbation algorithm. Although we are not able to give a bound for the prediction error as the time-series we consider is very generic, we may consider that it is in relation to the number of non-sampling points \( T - M \).

The challenge of fixed-rate sampling is in defining the sampling rate, i.e. the total number of samples allowed, \( M \). Increasing the sampling rate, i.e. when \( M \) is high, an extreme case of which is to issue a query at each time step as in the baseline Algorithm 4, the overall perturbation error will grow with \( M \). On the other hand, when we decrease the sampling rate, i.e. when \( M \) is low, the perturbation at each sampling point will drop, but the published series will not reflect up-to-date data values, resulting large prediction error. 

Algorithm 4 KFPredict(\( k \))

| Input: | Previous published value \( r_{k-1} \); apriori error covariance \( P_{k-1} \) |
| Output: | Apriori state estimate \( \hat{x}_k^- \), error covariance \( P_k^- \) |
| 1: | \( \hat{x}_k^- \leftarrow r_{k-1} \); |
| 2: | \( P_k^- \leftarrow P_{k-1} + Q \); |
| 3: | return \( (\hat{x}_k^-, P_k^-) \); |

Algorithm 5 KFCorrect(\( k \))

| Input: | Apriori state estimate \( \hat{x}_k^- \); measurement \( z_k \); apriori error covariance \( P_k^- \) |
| Output: | Aposterior state estimate \( \hat{x}_k \), aposterior error covariance \( P_k \) |
| 1: | \( K_k \leftarrow P_k^- (P_k^- + R)^{-1} \); |
| 2: | \( \hat{x}_k \leftarrow \hat{x}_k^- + K_k (z_k - \hat{x}_k^-) \); |
| 3: | \( P_k \leftarrow (1 - K_k)P_k^- \); |
| 4: | return \( (\hat{x}_k, P_k) \); |

5.2 Adaptive Sampling with Control

We adopt a controller in our overall solution to adaptively adjust the rate of sampling. Here we introduce the idea of feedback control and present details of the PID controller for the time-series application.

Feedback Control. A typical feedback control system starts with a measurement of the system output. The measurement is then compared with a desired value to generate the tracking error. The error is fed through a controller to generate a control action which will be further implemented into the input to the system. Then the system output will be monitored by the sensor to start another iteration.

The notation System is the process to regulate. In the adaptive sampling module, it represents the sampling process. The Controller component specifies the manner with which the error information is utilized to adjust the sampling rate. We describe the error feedback and the PID controller used below.

Error Feedback. The feedback to the controller in our solution is the relative error between the aposteriori estimate and the apriori estimate at each time step. At time step \( k_n \) (\( 0 \leq k_n < T \)), where the subscript indicates the \( n \)-th sampling point (\( 0 \leq n < M \)), we define this error \( E_{k_n} \) as follows:

\[
E_{k_n} = \frac{|\hat{x}_{k_n} - \hat{x}_{k_n}^+|}{\max(\hat{x}_{k_n}, \delta)}
\]  

(17)

Note that the posterior estimate is only available when given a noisy observation at time step \( k_n \). Thus no error is defined at non-sampling point.

The model error defined above is based on the amplified innovation error \( K_k \psi_k \) according to Equation (10). Clearly,
the model error measures how well the internal state model describes current data dynamics, suppose the \textit{aposteriori} estimate $\hat{x}_{kn}$ is close to the true value. Since the state prior is given by a constant state model, we may infer that data is going through rapid changes if the error $E_{kn}$ increases with time. In response, the controller in our system will detect the errors and adjust the sampling rate accordingly. We will take a closer look at how the controller takes action correspondingly.

**PID Controller.** The PID control algorithm is the most common form of feedback control and the foundation of almost all basic control applications [15]. King [15] defines three types of action by a PID controller: Proportional, Integral, and Derivative. For our application, we design a PID controller that measures the performance of our sampling process over time in terms of a compound error. Now we further re-define its three components.

- **Proportional** error is to keep the controller output ($\Delta$) in proportion to the current error $E_{kn}$ with $k_n$ being the current time step and subscript $n$ being the sampling point index

\[ \Delta_p = C_p E_{kn} \]  

where $C_p$ denotes the proportional gain which amplifies the current error.

- **Integral** error is to eliminate offset by making the rate of change of control output proportional to the error.

With similar terms, we define the integral control as follows:

\[ \Delta_i = \frac{C_i}{T_i} \sum_{j=n-T_i+1}^{n} E_{kj} \]  

where $C_i$ denotes integral gain amplifying the integral error and $T_i$ represents the integral time indicating how many recent errors are taken.

- **Derivative** error attempts to prevent large errors in the future by changing the output in proportion to the rate of change of error.

\[ \Delta_d = C_d \frac{E_{kn} - E_{kn-1}}{k_n - k_{n-1}} \]  

where $C_d$ is derivative gain amplifying the derivative error.

The full PID algorithm we have developed so far is thus

\[ \Delta = C_p E_{kn} + \frac{C_i}{T_i} \sum_{j=n-T_i+1}^{n} E_{kj} + C_d \frac{E_{kn} - E_{kn-1}}{k_n - k_{n-1}} \]  

Control gains $C_p$, $C_i$, and $C_d$ denote how much each of the proportional, integral, and derivative counts for the final calibrated PID error. We further constrain them to be non-negative and their summation is equal to 1:

\[ C_p, C_i, C_d \geq 0 \]  

\[ C_p + C_i + C_d = 1 \]

The three gains as well as integral time $T_i$ are system parameters to be set in our application.

---

**Algorithm 7** PIDControl($k_n$)

**Input:** Feedback errors $E_{kn-T_i+1}, ..., E_{kn}$

**Output:** New interval $I'$

1: $\Delta \leftarrow C_p E_{kn} + \frac{C_i}{T_i} \sum_{j=n-T_i+1}^{n} E_{kj} + C_d \frac{E_{kn} - E_{kn-1}}{k_n - k_{n-1}}$

2: $I' \leftarrow \max\{I + \theta(1 - e^{-\frac{\Delta}{\xi^2}}), \text{min}I\}$

3: return $I'$

Given the PID error $\Delta$, a new sampling interval $I'$ can be determined by the following equation:

\[ I' = I + \theta(1 - e^{-\frac{\Delta}{\xi^2}}) \]  

where $\theta$ and $\xi$ are pre-determined parameters.

A procedure that implements the PID controller is shown in Algorithm 7. Notice that we will keep the sampling interval above a minimum threshold $\text{min}I$. An overall algorithm with adaptive sampling was shown in Algorithm 3. As our approach might issue fewer than $M$ queries, the overall perturbation error introduced can be bounded by $O(M)$.

---

6. EXPERIMENT

We have implemented our approach in Java with JSC (Java Statistical Classes) for simulating the Laplace distribution. We empirically study the predictability and controllability of our proposed approach and compare with alternative methods in terms of utility.

Our study has been conducted with three real time-series data sets: flu, traffic, and unemployment.

- **Flu** is from the weekly surveillance data of Influenza-like illness provided by the Influenza Division of the Centers for Disease Control and Prevention\textsuperscript{2}. We collected the weekly outpatient count of the age group [5-24] from 2006 to 2010. This time-series consists of 209 data points.

- **Traffic** is a daily traffic count data set for Seattle-area highway traffic monitoring and control provided by the Intelligent Transportation Systems Research Program at University of Washington\textsuperscript{3}. We chose the traffic count at location I-5 143.62 southbound from April 2003 till October 2004. This time-series consists of 540 data points.

- **Unemployment** describes the monthly unemployment level of black or African American women of the age group [16,19] from ST. Louis Federal Reserve Bank\textsuperscript{4}. This data set contains observations from January 1972 to October 2011 with 478 data points.

Table 2 summarizes the default setting of parameters used in our experiments. Unless otherwise specified, the listed parameters take on their corresponding default values. This is not an optimal way of setting parameters. However we have been able to observe excellent performance by our approach. We will study the impact of individual parameters in next few sections.

\[ \text{http://www.jsc.nildram.co.uk} \]
\[ \text{http://www.cdc.gov/flu/} \]
\[ \text{http://www.jsc.nildram.co.uk} \]
\[ \text{http://research.stlouisfed.org/} \]
6.1 Accuracy and Robustness of Prediction

We first evaluate the accuracy of the Kalman filter estimation across all three data sets, compared against a simple linear predictor and pure Laplace perturbation from Algorithm 1. To this end, the experiment is set up in a way such that a noisy observation is obtained at each time step. The Kalman filter repeats the prediction-correction pair and releases the posterior estimate at each time step. The linear predictor generates a prediction based on the linear model but releases pure perturbed answers without correction. The relative error is calculated comparing the estimate/prediction against the true value and the average relative error, defined by Equation (1). The relative error is calculated over the entire series of plotted results. We measure the performance with respect to different scales for privacy budget \( \alpha \). The linear predictor is designed to fit a line with 2 data points as the data sets show little linearity in a larger scale. We set the process noise covariance \( Q \), defined in Equation (5), to be 0.001 and the measurement noise covariance \( R \), defined in Equation (8), to be 10 since we assume constant state model and small measurement noise. As shown in Figure 6, linear predictor is constantly worse than pure Laplace perturbation, because of the extra linear model error. The Kalman filter outperforms linear predictor as well as Laplace perturbation algorithm across all three data sets especially when given small privacy budgets (\( \alpha = 0.001, 0.001, 0.01 \)). With large budget (\( \alpha = 1 \)), which we note does not provide sufficient privacy protection, we observed no substantial advantage of using the Kalman filter, which can be explained by the nature of the \( \text{aposteriori} \) estimate defined by Equation (10); it only partially relies on the noisy measurement \( z_k \) hence does not fully reflect reduced perturbation error.

6.2 Effects of Kalman Filter Parameters

To understand the impact of process noise covariance \( Q \) and measurement noise covariance \( R \) on the accuracy of Kalman filter estimation, we vary their values independently when performing the prediction experiment. We present the results with the unemployment data set, since the other data sets show similar trends. Given \( \alpha \) is 0.01, results are presented in Figure 7. In Figure 7(a), we plot the average relative error with respect to different scales for \( Q \) and we observe similar trends among three \( R \) values. With \( R \) fixed, we can see that the accuracy of Kalman filter drops as \( Q \) increases. In Figure 7(b) we plot the average relative error with respect to different scales for \( R \) using three \( Q \) values. Given \( Q \) fixed, the accuracy improves as \( R \) increases. This can be explained by Equation (14.16) which define how the Kalman gain is calculated. Increasing \( Q \), the Kalman gain increases, resulting the \( \text{aposteriori} \) estimate, i.e., the final released value, favoring the noisy observation. Increasing \( R \), the Kalman gain decreases, resulting the \( \text{aposteriori} \) estimate favoring the \( \text{apriori} \) state prediction. Both results consistently state that it’s beneficial to rely more on the state prior than the noisy measurement with a small privacy budget, which means the observed values have larger noise.

6.3 Accuracy and Robustness of Adaptive Sampling

As the Kalman filter provides satisfactory estimations with limited privacy budget, we now examine the performance of adaptive sampling with respect to fixed-rate, given the privacy budget \( \alpha \) is 0.01. We vary the fixed sampling interval

---

### Table 2: Experiment Default Setting

| Parameter | Default Value |
|-----------|---------------|
| \( \delta \) | 1             |
| \( \alpha \) | 0.01          |
| \( Q \)   | 0.001         |
| \( R \)   | 10            |
| \( C_P \) | 0.9           |
| \( C_I \) | 0.1           |
| \( C_d \) | 0             |
| \( M \)   | 25\% \times T|
| \( T_s \) | 5             |
| \( \theta \) | 10           |
| \( \xi \) | 1000          |
| \( \text{min}\ I \) | 1          |
from 1 to 10, corresponding to the sampling rate from 100% to 10%. For our adaptive approach, we set $M$ to be $25\% T$.

The results with all three data sets are shown in Figure 8. As the sampling interval increases, i.e., from 1 to 5 in Figure 8(a), fixed-rate sampling shows reduced average relative error of various scales. This phenomenon can be interpreted by the decrease of total perturbation error resulting from less frequent queries. As the interval further increases, from 7 to 10 as in Figure 8(a) and from 7 to 10 as in Figure 8(b), the average relative error starts to rise again. This can be explained by the accumulation of prediction error, due to long-term prediction without correction. The optimal sampling interval is not known apriori and may differ from dataset to dataset. As in Figure 8, we found that the performance of adaptive sampling is comparable to the optimal fixed-rate sampling across all data sets, thanks to the adaptive strategy.

### 6.4 Effects of PID Parameters

We first examine the effect of $M$, the maximum allowable number of queries, with traffic data set shown in Figure 9(a). With the ratio of $M$ against $T$ increasing, the average relative error grows, which can be explained by the accumulation of perturbation error. Figure 9(b) studies the impact of control gains as opposed to the integration time. Fixing the case $C_p = 1, C_i = 0, C_d = 0$ as the baseline, the difference between the performance of every other setting and that of the baseline is plotted. In addition to the constraints from Equation (22), (23), we choose the control gains according to the common practice: proportional $> \text{integral} > \text{derivative}$. As seen in the plot, when the integration time increases, the resulting relative error shows no clear trend. Furthermore, between different control gain settings, the variation of error is small enough so that we believe it’s due to the randomness of Laplace perturbation error. Therefore we conclude that there's no extra “rule of thumb” beside the common practice for tuning the controller gains in our application.

We also study the impact of $\xi$ and $\theta$ as in Equation (24) on the accuracy of the published data series. Fixing either one of them, varying the value of the other does not result in substantial changes in terms of relative error. We consider both of them not influential and exclude the detailed figures.

### 6.5 Overall Performance vs. Alternative Approaches

To compare our method with respect to alternative approaches in Section 2, we also implemented Algorithm 1 as well as Algorithm 2. The number of DFT coefficients to preserve, $k$, is set to be 20, which is near optimal according to the study of 23. We plot the average relative error by our adaptive approach, Laplace perturbation algorithm, and Discrete Fourier Transform (DFT) algorithm with respect to different privacy budget scales. Results are shown in Figure 10. Again, the adaptive approach shows superior performance when the privacy budget $\alpha$ is limited. This confirms our hypothesis that with accurate estimate by the Kalman filter, the PID control mechanism can adjust the sampling rate as needed, thus improving the overall utility of the published series. Note that when the privacy budget is high and approaching 1, the baseline Laplace perturbation algorithm achieves smaller relative error because of the reduced perturbation error. Since the reconstruction error of the DFT approach and the prediction error of our adaptive approach both outweigh the perturbation error in this case, their released series contain larger relative error. However, a privacy budget greater than 1 does not provide sufficient privacy protection any more. We find that our adaptive approach outperforms the alternative methods under strong privacy guarantee.

### 7. RELATED WORK

Here we give a brief review over existing works related to differential privacy, time series, and the Kalman filter.

**Differential privacy on static data.** Dwork et al. [8] established the guideline to guarantee differential privacy for individual aggregate queries by calibrating the Laplace noise to the global sensitivity of each query. Since then, various mechanisms have been proposed to enhance the accuracy of differentially private data release. Blum et al. [2] proved the possibility of non-interactive data release satisfying differential privacy for queries with polynomial VC-dimension, such as predicate queries. Dwork et al. [10] further proposed more efficient algorithms to release private sanitized data with high accuracy. The work of Hay et al. [11] improved the accuracy of a tree of counting queries through consistency check, which is done as a post-processing procedure after adding Laplace noise. This hierarchical structure of queries is referred to as histograms by several techniques [11, 17, 27], where each level in the tree is an increasingly fine-grained summary of the data. The work by Xiao et al. [27] proposed a two-phase partitioning algorithm using kd-tree structure to improve the accuracy of released histograms. Li et al. [17] studied the feasibility of providing an optimal query strategy by analyzing a workload of counting queries apriori and estimating answers from the query strategy. Xiao et al. [26] proposed another approach based on the Harr wavelet which transforms original data summary before adding Laplace noise to it. Another recent study [25], aiming to reduce the relative error, suggests to inject different amount of Laplace noise based on the query result and works well with multidimensional data. Several other works studied differentially private mechanisms for particular kinds of data, such as search logs [10] and set-valued data [6]. When applied to highly self-correlated time-series data, all the above methods, designed to perturb static data, become problematic because of highly compound Laplace perturbation error.

**Time series.** Time series data is pervasively encountered in...
the fields of engineering, science, sociology, and economics. Various techniques [3], such as ARIMA modeling, exponential smoothing, ARAR, and Holt-Winters methods, have been studied for time-series forecasting. Papadimitriou et al. [22] studied the trade-offs between time-series compressibility property and perturbation. They proposed two algorithms based on Fast Fourier Transform (FFT) and Discrete Wavelet Transform (DWT) respectively to perturb time-series frequencies. But the additive noise proposed by them does not guarantee differential privacy, meaning it does not protect sensitive information from adversaries with strong background knowledge. Rastogi and Nath [23] proposed a Discrete Fourier Transform (DFT) algorithm which implements differential privacy when perturbing time-series data. However, the DFT algorithm cannot hide data on-the-fly in a streaming environment. We are also aware of the recent work by Dwork et al. [9] on continuous data streams. They defined the event-level privacy to protect an event, i.e., one user’s presence at a particular time point, rather than the presence of that user. If one user contributes to the aggregation at time point $t-1$, $t$, and $t+1$, the event-level privacy hides the user’s presence at only one of the three time points, resulting the rest two open to attack.

Kalman filter. R.E. Kalman published the seminal paper on the Kalman filter [13] in 1960. Since then, it has become widely applied to areas of signal processing [4] and assisted navigation systems [1]. It has also gained popularity in other areas of engineering. One particular application is to wireless sensor networks. Jain et al. [12] adopted a dual Kalman filter model on both server and remote sensors to filter out as much data as possible to conserve resources. But their main concern was to minimize memory usage and communication overhead between sensors and the central server by storing dynamic procedures instead of static data. Distributed Kalman filtering [20] by Olfati-Saber was deployed to a network of sensors to reach a consensus of estimate among neighboring sensor nodes. The complexity and stability of such deployment was formally studied in [21].

8. CONCLUSION

We have proposed an adaptive approach with sampling and estimation to release time series under differential privacy. The key innovation is that our approach utilizes feedback loops based on observed (perturbed) values to dynamically adjust the prediction/estimation model and the sampling rate. To minimize the overall privacy cost, the solution uses the PID controller to adaptively sample long time-series according to detected data dynamics. As to improve the accuracy of data release per timestamp, the Kalman filter is used to predict data values at non-sampling points and to estimate true values from perturbed query answers at sampling points. Our experiments with three real data sets show that it is beneficial to incorporate feedback into both the prediction model and the sampling process. The results confirmed that our adaptive approach improves accuracy of time-series release and has excellent performance even under very small privacy cost.

As for the future, we plan to develop an improved filter to handle a Laplacian measurement noise, as defined by Equation (7), instead of approximating it with a Gaussian noise. A good candidate is the Masreliez filter proposed in [18] and we need to overcome the difficulty of implementing the convolution operation involved in evaluating the score function of the Masreliez filter. Another potential direction is to
expand our solution to publish differentially private spatial-temporal statistics, for example, real-time traffic conditions at all intersections of a city.

9. ACKNOWLEDGMENTS

This research was supported in part by NSF grant CNS-1117763 and an Emory URC grant.

10. REFERENCES

[1] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. State Estimation in Discrete-Time Linear Dynamic Systems, pages 199–266. John Wiley & Sons, Inc., 2002.

[2] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In STOC, 2008.

[3] P. Brockwell and R. Davis. Introduction to time series and forecasting. Springer texts in statistics. Springer, 2002.

[4] R. G. Brown and P. Y. C. Hwang. Introduction to Random Signals and Applied Kalman Filtering, volume 2. John Wiley & Sons, 1997.

[5] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. In S. Abramsky, C. Gavoille, C. Kirchner, F. M. auf der Heide, and P. G. Spirakis, editors, ICALP (2), volume 6199 of Lecture Notes in Computer Science, pages 405–417. Springer, 2010.

[6] R. Chen, N. Mohammed, B. C. M. Fung, B. C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. The Proceedings of the VLDB Endowment (PVLDB), 4(11):1087–1098, August 2011.

[7] H. Cramér. Mathematical methods of statistics, by Harold Cramér. Princeton mathematical series. 1957.

[8] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In In Proceedings of the 3rd Theory of Cryptography Conference, pages 265–284. Springer, 2006.

[9] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In Proceedings of the 42nd ACM symposium on Theory of computing, STOC ’10, pages 715–724, New York, NY, USA, 2010. ACM.

[10] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In Proceedings of the 41st annual ACM symposium on Theory of computing, STOC ’09, pages 381–390, New York, NY, USA, 2009. ACM.

[11] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private queries through consistency. CoRR, abs/0904.0942, 2009.

[12] A. Jain, E. Y. Chang, and Y.-F. Wang. Adaptive stream resource management using kalman filters. In Proceedings of the 2004 ACM SIGMOD international conference on Management of data, SIGMOD ’04, pages 11–22, New York, NY, USA, 2004. ACM.

[13] R. E. Kalman. A new approach to linear filtering and prediction problems. 1960.

[14] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In Proceedings of the 2011 international conference on Management of data, SIGMOD ’11, pages 193–204, New York, NY, USA, 2011. ACM.

[15] M. King and M. King. Process Control: A Practical Approach. John Wiley & Sons, 2011.

[16] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, editors, WWW, pages 171–180. ACM, 2009.

[17] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In In PODS ’10: Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems of data, pages 123–134. ACM, 2010.

[18] C. Masreliez. Approximate non-gaussian filtering with linear state and observation relations. Automatic Control, IEEE Transactions on, 20(1):107 – 110, feb 1975.

[19] F. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In SIGMOD, 2009.

[20] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In Decision and Control, 2007 46th IEEE Conference on, pages 5492 –5498, dec. 2007.

[21] R. Olfati-Saber. Kalman-consensus filter : Optimality, stability, and performance. In CDC, pages 7036–7042, 2008.

[22] S. Papadimitriou, F. Li, G. Kollios, and P. S. Yu. Time series compressibility and privacy. In Proceedings of the 33rd international conference on Very large data bases, VLDB ’07, pages 459–470. VLDB Endowment, 2007.

[23] V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In SIGMOD, 2010.

[24] G. Welch and G. Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 1995.

[25] X. Xiao, G. Bender, M. Hay, and J. Gehrke. ireduct: differential privacy with reduced relative errors. In Proceedings of the 2011 international conference on Management of data, SIGMOD ’11, pages 229–240, New York, NY, USA, 2011. ACM.

[26] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. Knowledge and Data Engineering, IEEE Transactions on, 23(8):1200 –1214, aug. 2011.

[27] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. In W. Jonker and M. Petkovic, editors, Secure Data Management, volume 6358 of Lecture Notes in Computer Science, pages 150–168. Springer Berlin / Heidelberg, 2010.