Technical Report

TR-2011-014

Adaptively Sharing Time Series Data with Differential Privacy

by

Liyue Fan, Li Xiong

Mathematics and Computer Science

EMORY UNIVERSITY

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

1. INTRODUCTION

Sharing 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 Surveilence** A health care provider, such as an Emergency Department, gathers data from a number of 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 epidemipic 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 group of users at each region during each time period, can be mined for commercial interest ,such as popular places, as well as public interest, such as congestion patterns in roads.

As the examples above suggest, aggregate statistics of private data can be quite useful. 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 shared with other un-trusted entities for various purposes. The trusted server, i.e. publisher, is assumed to be bound by contractual obligations to protect the users' 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, while guaranteeing the privacy of, the data from individual users.

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 $\mathcal{A}$. so that the output of $\mathcal{A}$ remains roughly the same even if any single tuple in the input data is arbitrarily modified. This ensures that given the output of $\mathcal{A}$, an adversary will not be able to infer much about any single tuple in the input, and thus privacy is protected.

Previous studies, including [2] [3] [4] [8] [12] [11], work well with static data (except [3] [8]). However, the data generated by the above applications we consider are time series data. This means, data values at successive timestamps can by highly correlated, which makes those privacy solutions problematic. In order to publish long time-series, standard differential privacy mechanism, e.g. [2], require that noise be added at each timestamp according to the composition theorem, can result in an overall perturbation error of $\Theta(n)$, where n is the length of the time series, making the published version practically useless if n is very large.

Rastogi and Nath [8] studied a similar time-series sharing problem and proposed an algorithm based on the Discrete Fourier Transform and the Inverse DFT of the entire time-series. Since the entire time-series is required to perform those operations, the timeliness of publishing is greatly impacted, resulting the shared data useless for disease surveilence and traffic monitoring applications. Moreover, this solution also suffers reconstruction error when calculating the Inverse DFT to recover the original time-series.

Our Contributions. In this paper, we propose to model the time-series sharing problem as a control system, adaptively sampling the original series in order to minimize privacy cost as well as publishing error. 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 truth? The latter imposes another question: suppose an accurate estimate can be derived at any time step, can we reduce the number of queries by adjusting the sampling rate according to the rate of data change? We propose a solution to address those two issues which have not be studied by existing works. With differential privacy as the privacy requirement, we make the following contributions:

1) To improve the accuracy of data release per timestamp, we propose to use the Kalman filter [5] which is widely adopted for signal recovery, to estimate the original data value. 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

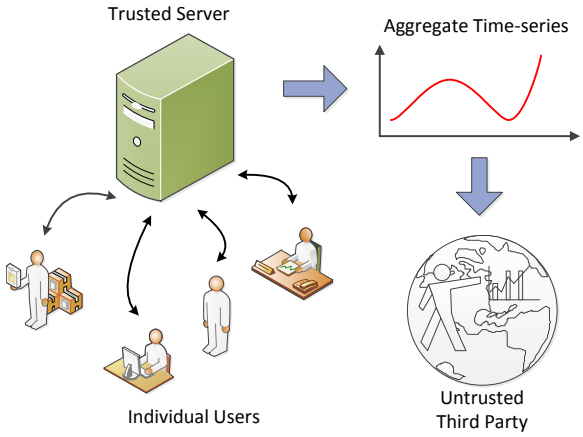

Figure 1: Aggregate data sharing scenario

an prediction generated by the process model and the perturbed observation. The combined value, referred to as *a posteriori* estimate, is a *minimum variance* estimate, which provides an educated guess rather than a pure perturbed value.

2) To minimize the overall privacy cost, hence, the overall perturbation error, we propose an adaptive sampling algorithm which adjusts the sampling rate using Feedback Control. A PID controller, which is the most common form of feedback controller, is in place to detect data dynamics and to increase query frequency when data is going through rapid changes. Figure 2 illustrate the idea the adaptive sampling.

3) We empirically study the *predictability* and *controllability* in terms of accuracy and robustness with real time-series data sets. Our experiments show the proposed solution provids accurate results and stability despite different data dynamics. We believe our solution is applicable to a wider range of applications with respect to that of existing works.

The organization of this paper is as follows: Section 2 provides the background for differential privacy and states two baseline approaches; Section 3 presents an overview of our proposed solution; Section 4 and 5 introduce the Kalman filter and feedback control respectively and system models as well as technical details; Section 6 evaluates the predictability and controbility empribically; Section 7 concludes the paper with possible future working directions.

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. A univariate, discrete time series $\mathbf{X} = \{X[k]\}$ *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. The average relative error, denoted by RE, of a published series $\mathbf{R} = \{R[k]\}$ *derived from original time-series* $\{X[k]\}$ *is given by the following equation:*

$$RE = \frac{1}{T}\sum_{k=0}^{T-1} |R[k] - X[k]|/max\{X[k], \delta\} \quad (1)$$

where δ is a user-specified constant (also referred to as *sanitary bound* as in [11]) 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 publish 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** [1] is given as follow. Here the parameter, α, specifies the degree of privacy offered.

Definition 3. A non-interactive privacy mechanism $\mathcal{A}$ *gives* α*-differential privacy if for any dataset* D_1 *and* D_2 *differing on at most one record, and for any possible anonymized dataset* $\widetilde{D} \in Range(\mathcal{A})$*,*

$$Pr[\mathcal{A}(D_1) = \widetilde{D}] \leq e^{\alpha} \times Pr[\mathcal{A}(D_2) = \widetilde{D}] \quad (2)$$

where the probability is taken over the randomness of $\mathcal{A}$*.*

Laplace Mechanism. Dwork et al.[2] 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 $p(x|\lambda) = \frac{1}{2\lambda}e^{-|x|/\lambda}$, where λ is determined by both the desired privacy level α and the **global sensitivity** [2] of a query which is defined below.

Definition 4. For any function $f : D \rightarrow \mathbb{R}^d$*, the Global Sensitivity of* f *is*

$$GS(f) = \max_{D_1, D_2} ||f(D_1) - f(D_2)||_1 \quad (3)$$

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

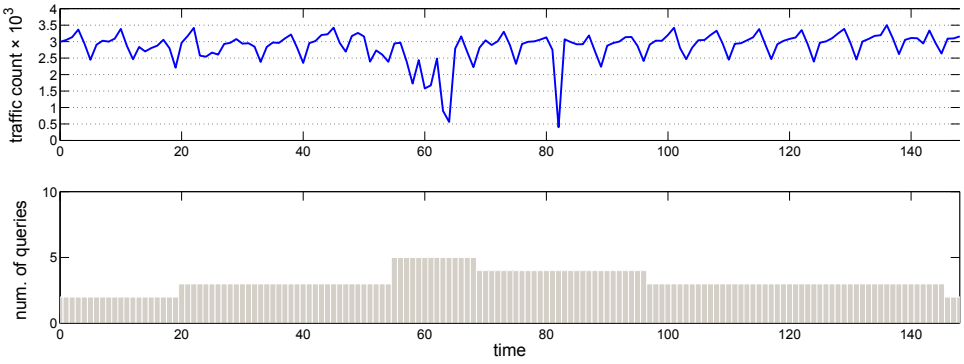

Figure 2: Adaptive sampling with traffic data: the number of queries issued per time unit increases between [50,100] and drops beyond 100

Table 1: Frequently Used Notations

Symbol	Description
$\mathbf{X}$	original time-series
$\mathbf{R}$	published time-series
T	length of $\mathbf{X}$
δ	sanitary bound
α	privacy level in differential privacy
$Lap(\lambda)$	Laplace distribution with magnitude λ
C_p	proportional gain
C_i	integral gain
C_d	derivative gain
M	total allowable number of queries
T_i	integration time
I	current sampling interval
θ, ξ	interval adjustment params
$minI$	minimum length of interval

Dwork et al.[2] prove that adding Laplace noise of magnitude $\lambda = GS(q)/\alpha$ to the true answer of query q leads to α-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* [7].

THEOREM 1. [7] *Let $\mathcal{A}_i$ each provide α_i-differential privacy. A sequence of $\mathcal{A}_i(D)$ over the dataset D provides $(\sum_i \alpha_i)$-differential privacy.*

2.3 Alternative Solutions

Here we discuss two alternative solutions and their detailed algorithms. Empirical studies of them comparing to our proposed solution are included in Section 6.

Laplace Perturbation Algorithm. With the theorem of composition, a time-series publishing problem can be treated as a sequence of T recurring queries, assuming *count* queries. If each individual query is α/T-differentially private, the sequence of queries guarantee α-differential privacy. The outline of this naive Laplace Perturbation Algorithm is listed in Algorithm 1:

Algorithm 1 Laplace Perturbation Algorithm(LPA)

Input: Raw time-series $\mathbf{X}$; privacy budget α
Output: Perturbed publish $\mathbf{R}$

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

Algorithm 2 Discrete Fourier Transform(DFT)

Input: Raw time-series $\mathbf{X}$; privacy budget α
Output: Perturbed publish $\mathbf{R}$

1: compute $\mathbf{F}^k = \mathbf{DFT}^k(X)$
2: compute $\widetilde{\mathbf{F}}^k = LPA(\mathbf{F}^k, \alpha)$;
3: compute $\mathbf{R} = \mathbf{IDFT}(\mathbf{PAD}^T(\widetilde{\mathbf{F}}^k))$;
4: **return** $\mathbf{R}$;

Discrete Fourier Transform. Rastogi and Nath[8] proposed the Fourier Perturbation Algorithm FPA_k that transforms the raw series into frequency domain, perturbs the first k coefficients, and reconstructs the series with the perturbed k coefficients. Their method is shown in Algorithm 2. It begins by computing $\mathbf{F}^k$, which is composed of the first k Fourier coefficients in the Discrete Fourier Transform(DFT) of $\mathbf{X}$, with the j^{th} coefficient is given as: $DFT(\mathbf{X})_j = \sum_{i=0}^{T-1} e^{\frac{2\pi\sqrt{-1}}{T}ji}\mathbf{X}_i$. Then it perturbs $\mathbf{F}^k$ using LPA algorithm with privacy budget α, getting a noisy estimate $\widetilde{\mathbf{F}}^k$. This perturbation is to guarantee differential privacy. Denote $\mathbf{PAD}^T(\widetilde{\mathbf{F}}^k)$ the sequence of length T by appending $T-k$ zeros to $\widetilde{\mathbf{F}}^k$. The algorithm finally computes the Inverse Discrete Fourier Transform(IDFT) of $\mathbf{PAD}^T(\widetilde{\mathbf{F}}^k)$ to get $\mathbf{R}$. The j^{th} element is given as: $IDFT(\mathbf{X})_j = \frac{1}{T}\sum_{i=0}^{T-1} e^{-\frac{2\pi\sqrt{-1}}{T}ji}\mathbf{X}_i$.

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 time-series prediction/estimation with desired accuracy.

3.1 Framework

It is intuitive that a good adaptation scheme is to issue more queries to the original time-series when data value is going through rapid variations and fewer queries when the 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 stratey.

The framework of our solution is shown in Figure 3. In a feedback loop presentation, the framework is composed of input, perturbation mechanism, Kalman filter, PID controller, and output.

- The **input** is a streaming time-series with one value at a time. It is sampled by the PID controller. As a result, not every data item is needed from the stream.
- The **Laplace mechanism** perturbs each item that actually enters the system in order to guarantee differencial privacy.
- Upon receiving a noisy observation from the Laplace perturbation, the **Kalman filter** *Correction* procedure is activated to generate the *a posteriori* estimate, which is achieved by correcting the *a priori* prediction(by *Prediction* procedure) with the obeservation.
- The prediction error, between the prior and the posterior, is then fed through the **PID controller** to determine a new sampling rate.

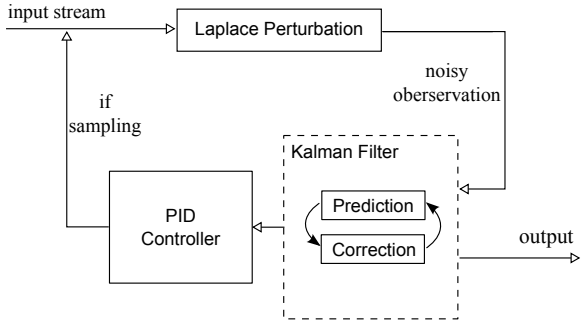

Figure 3: Closed chain control loop of our solution

With or without a noisy observation, the Kalman filter *Prediction* procedure produces the *a priori* prediction at each time step based on an internal state model. This *a priori* prediction will be published when a noisy observation is absent (a non-sampling point). More details about Kalman filter and PID controller will be further discussed in the next two sections.

3.2 Algorithm

Here we present the details about our solution in Algorithm 3. It is proved to be α-differentially private.

Queries will be issued through the Laplace mechanism for the first T_i timestamps to collect enough feedback for the PID controller(Line 3-5 in Algorithm 3). Line 6 initializes the interval with a predefined minimum value $minI$. Line 7 initializes the error covariance for $KFPredict$ for the next time step. For each time step after the first T_i, a *prior* prediction is generated by $KFPredict$ procedure.

Algorithm 3 Our solution (good name??)

Input: Raw time-series $\mathbf{X}$; privacy budget α
Output: Perturbed publish $\mathbf{R}$

1: $k, queryCount \leftarrow 0$;
2: **while** $queryCount < T_i$ **do**
3: draw $noise \sim Lap(M/\alpha)$;
4: $R[k] \leftarrow X[k] + noise$;
5: $k \leftarrow k+1$, $queryCount \leftarrow queryCount + 1$;
6: $interval \leftarrow minI$;
7: $P_{T_i-1} \leftarrow 100$; *initialize the error covariance*
8: **for** each $k \in (T_i, T_i+1, ..., T-1)$ **do**
9: $prior, P_k^- \leftarrow KFPredict(k)$;
10: **if** $k\%interval = 0$ **and** $queryCount < M$ **do**
11: draw $noise \sim Lap(M/\alpha)$;
12: $measurement \leftarrow X[i] + noise$;
13: $posterior, P_k \leftarrow KFCorrect(k)$;
14: release $R[i] \leftarrow posterior$;
15: compute $\Delta \leftarrow PIDError(k)$;
16: $interval \leftarrow max\{interval + \theta(1 - e^{\frac{\Delta-\xi}{\xi}}), minI\}$;
17: $queryCountl \leftarrow queryCount + 1$;
18: **else**
19: release $R[i] \leftarrow prior$;
20: **return** $\mathbf{R}$;

The feedback control loop is implemented by Line 8 to 19. If the current time step is a sampling point ($k\%interval = 0$) and there's budget left for more queries($queryCount < M$), a noisy measurement is retrieved from Laplace perturbation, and it is used by $KFCorrect$ to obtain an updated estimate *posterior*. Line 14 publishes the *posterior*. A new interval is determined by the $PIDError$ output from Line 15 and parameters θ and ξ from Line 16 which amplify the PID error. Note we always keep the interval above the minimum length. If either condition in Line 10 evaluates to false, Line 19 publishes the prediction *prior* to save on the privacy budget.

THEOREM 2. *Algorithm 3 satisfies α-differential privacy.*

PROOF. Each query issued through the Laplace mechanism is α/M-differentially private. The maximum number of queries allowed is M. By the composition rule, i.e. Theorem 1, the privacy budget spent is at most $M \times \alpha/M = \alpha$.

4. PREDICT WITH MINIMAL UNCERTAINTY

In this section, we will provide an introduction to the discrete Kalman filter, the advantage of using Kalman filter in our problem setting, and the application model.

4.1 The Kalman Filter

The Kalman filter was introduced in 1960 by R. E. Kalman [5] 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. The process model is represented by:

$$x_{k+1} = A_k x_k + \omega_k, \tag{4}$$

with a measurement that is

$$z_k = H_k x_k + \nu_k \tag{5}$$

where x_k is internal state of the process, A_k is a state transition matrix relating x_k to x_{k+1}, ω_k is a process model noise, z_k is a measurement, H_k is a matrix relating system state and external measurement, ν_k is a measurement noise, k is the discrete time index.

The process and measurement noise variables ω_k and ν_k are assumed to be independent of each other, white, and with normal probability distributions

$$p(\omega_k) \sim N(0, Q_k), \tag{6}$$

$$p(\nu_k) \sim N(0, R_k). \tag{7}$$

where Q_k and R_k are process noise covariance and measurement noise covariance respectively.

At time step k, the *a priori* state estimate $\hat{x_k}^-$ is made based on the system model (4) and is related to the *a posteriori* state estimate of last step:

$$\hat{x_k}^- = A_k \hat{x_{k-1}}. \tag{8}$$

The *a posteriori* state estimate $\hat{x_k}$ is based on a linear combination of *a priori* state estimate $\hat{x_k}^-$ and the weighted prediction error ψ_k. The *a posteriori* estimate is calculated as follows:

$$\hat{x_k} = \hat{x_k}^- + K_k \psi_k. \tag{9}$$

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

$$\psi_k = z_k - H_k \hat{x_k}^-. \tag{10}$$

The value of the weight K_k is called *Kalman Gain* which is adjusted with each measurement. In order to reduce uncertainty of the *a posteriori* estimate $\hat{x_k}$, K_k is chosen to minimize the *a posteriori* error covariance P_k, which is defined as

$$P_k = E[(x_k - \hat{x_k})(x_k - \hat{x_k})^T]. \tag{11}$$

Symmetrically, we define the *a priori* error covariance P_k^- as

$$P_k^- = E[(x_k - \hat{x_k}^-)(x_k - \hat{x_k}^-)^T]. \tag{12}$$

Applying the least-square method to (11), we get

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}, \tag{13}$$

$$P_k = (I - K_k H_k) P_k^-. \tag{14}$$

The projection of the *a priori* error covariance P_k^- can be derived given (8), (11), (12), and P_{k-1}, resulting

$$\hat{P_k}^- = A_k P_{k-1} A_k^T + Q_k. \tag{15}$$

4.2 Why Kalman Filter?

The Kalman filter estimates a process by using a form of feedback control: the filter estimtates the process state at some time and it obtains feedback in the form of (noisy) measurements. It aligns perfectly with our time-series publishing application where only perturbed data values are available. We are then inspired to model the time-series data with the process model (4), and to treat obersavations

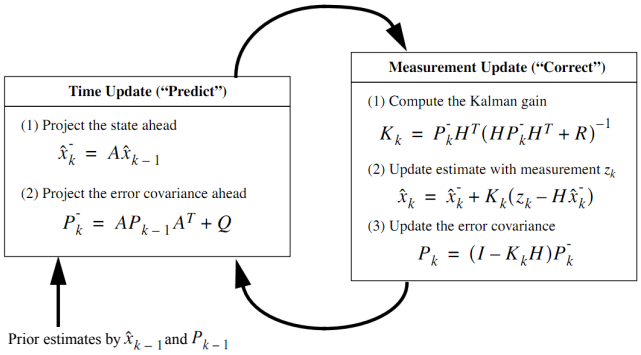

Figure 4: A complete picture of Kalman filter

out from Laplace perturbation mechanism as noisy measurements (5).

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 external measurements) to form an educated guess. It accomplishes that by repeating the following two mechanisms [10]:

- *Prediction:* This mechanism projects forward in time the current state and error estimates to obtain the *a priori* 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 *a priori* estimate to obtain an improved *a posteriori* estimate. At time step $k+1$ when an actual measurement is available, the filter corrects itself based on the innovation.

Welch and Bishop [10] give a high-level diagram of the two operations shown in Figure 4. After each prediction and correction pair, the process is repeated with the previous *a posteriori* estimates used to project or predict the new *a priori* estimates. Note that they simplify the problem setting by assuming the *process noise covariance* Q, *measurement noise covariance* R, and matrices A and H are time invariant. The constancy assumption will hold true in our application model.

4.3 Time-series Application Model

In the time-series publishing scenario, we adopt a constant system model which is given by the following equation:

$$x_{k+1} = x_k + \omega \tag{16}$$

meaning adjacent data values from the original time-series should be consistent except for a Gaussian process model noise ω:

$$p(\omega) \sim N(0, Q). \tag{17}$$

Our observation, which is perturbed data out from the Laplace mechanism, is modeled by:

$$z_k = x_k + \nu \tag{18}$$

where ν ,the measurement noise, is a Laplacian noise which follows:

$$p(\nu) \sim Lap(0, \lambda) \tag{19}$$

Algorithm 4 KFPredict(k)

Input: Previous publish $R[k-1]$; *a posteriori* error covariance P_{k-1}
Output: *A priori* 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: *A priori* state estimate $\hat{x_k}^-$; measurement z_k; *a priori* error covariance P_k^-
Output: *A posteriori* state estimate $\hat{x_k}$, *a posteriori* 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** $(\hat{x_k}, P_k)$;

where λ is the magnitude parameter determined by differential privacy mechanism.

In our application, we will use a small Gaussian error to approximate the actual noise distribution $Lap(\lambda)$. Thus we define the distribution of ν as

$$p(\nu) \sim N(0, R). \tag{20}$$

The *prediction* mechanism is implemented by *KFPredict* in Algorithm 4. Note that we always derive *a priori* state estimate using the most recent publish.

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 through Laplace mechanism comes with a cost (privacy budget spent), we are encouraged to devise a query/sampling strategy to wisely allocate the overall budget, issuing queries through the differential privacy interface only when needed. As a result, many times a noisy observation for the Kalman filter may be absent. In our overall solution, we propose to release prior estimates when the observation is absent and to correct when the observation is available.

5. QUERY STRATEGY

In this section we will discuss two strategies to query the input data stream: fixed-rate and adaptive sampling. We propose an adaptive sampling approach, guided by the PID control algorithm, to achieve near optimal query strategy.

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 6 shows a sketch of the algorithm. The variable *numQuery* represents the total number of queries allowed and the individual budget for each query is equivalent to $\alpha/numQuery$. Line 7 indicates that for the non-query

Algorithm 6 Fixed Rate Sampling

Input: Raw time-series $\mathbf{X}$; privacy budget α; fixed query interval I
Output: Perturbed publish $\mathbf{R}$

1: $numQuery \leftarrow \mathbf{X}.length/I$;
2: **for** each $k \in 0, 1, ..., T-1$ **do**
3: $\quad prior \leftarrow KFPredict(k)$;
4: $\quad$ **if** $k\%I = 0$ **do**
5: $\quad\quad$ sample $noise \sim Lap(numQuery/\alpha)$;
6: $\quad\quad posterior \leftarrow X[k] + noise$;
7: $\quad\quad R[k] \leftarrow KFCorrect(k)$;
8: $\quad$ **else**
9: $\quad\quad R[k] \leftarrow prior$;
10: **return R**;

points, we use the prior estimate given by the Kalman filter.

The challenge of fixed-rate sampling is in defining the sampling rate, ie. the time interval I between two adjacent sample points. If the sampling rate is high, an extreme case of which is to query and purturb at every time point, the individual budget for each query would very small thus resulting in high perturbation error. On the other hand when the sampling rate is low, the perturbation at each query point will decrease, but the published series will not reflect the data trend well. *A priori* knowledge of the data is required to find the optimal sampling rate in order to better describe the trend and minimize publish error. However, that is impractical for a real-time publishing scenario.

With no *a priori* knowledge of the time series, it is desirable and necessary to detect data dynamics and to adjust the sampling rate accordingly. When data value shows stability based on previous samples, the sampling rate should decrease to save the privacy budget for the future. On the other hand, if data is going through rapid changes, the sampling rate should increase to capture the data changes and to improve utility.

5.2 Adaptive Sampling with Control

We adopt a feedback control approach to achieve adaptively adjusting the rate of sampling. The feedback, i.e. error, to the controller in the time-series application is the relative error between the *a posteriori* estimate and the *a priori* estimate at any particular time step. We define the error E_k at time step k as follows:

Definition 5. At a given time step k, the error E_k is defined as

$$E_k = \frac{\hat{x_k} - \hat{x_k}^-}{max\{\hat{x_k}, \delta\}} \tag{21}$$

if the a posteriori estimate $\hat{x_k}$ is available.

Note that $\hat{x_k}$ is only available when given a noisy observation at time step k. Thus no error is defined at non-sampling point.

Clearly, the error E_k measures how well the internal state model describes current data dynamics, suppose the *a posteriori* estimate $\hat{x_k}$ 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_k increases with time. In response, the controller in

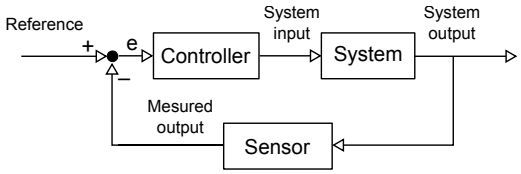

Figure 5: Generic feedback loop

our system will detect the errors and adjust the sampling rate accordingly. We will take a closer look at how the controller takes action correspondlingly.

5.2.1 Feedback Control Problem

In general, a control problem is formulated to ensure that a process operates at its design specifications, satisfying some predetermined dynamic performance criteria. The most common structure is feedback, as this structure can monitor variations in the process and successfully compensate for unwated excursions in a manner conssitent with the performance objectives.

The role of the feedback controller, according to Romagnoli and Palazoglu [9], is to react to the variantions in the process operation caused by the disturbances and recover the desired operation in a timely and smooth manner.

A block diagram showing the concept of the feedback loop is in Figure 5. It starts with a measurement of the system output. The measurement is then compared with a desired value to generate the tracking error e. The error is fed through the controller to generate an control action which will be further implemented into the input to the system. The system output will be monitored by the sensor to start another iteration.

The *System* in Figure 5 is the process to regulate. In the time-series sharing application, it represents the sampling process.

The *Controller* component in Figure 5 specifies the manner with which the error information is utilized. We choose *PID control* mechanism which will be analyzed in the next.

5.2.2 PID Controller

The PID control algorithm is the most common form of feedback control and the foundation of almost all basic control applications [6]. King [6] 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 data model over time in terms of a compound error. Now we further re-define its three components.

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

$$\Delta_p = C_p E_{k_n} \tag{22}$$

  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.

Algorithm 7 PIDControl(k_n)

Input: Feedback errors $E_{k_{n-T_i+1}}, ..., E_{k_n}$
Output: New interval I'

1: $\Delta \leftarrow C_p E_{k_n} + \frac{C_i}{T_i} \sum_{i=n-T_i+1}^{n} E_{k_i} + C_d \frac{E_{k_n} - E_{k_{n-1}}}{k_n - k_{n-1}}$;
2: $I' \leftarrow max\{I + \theta(1 - e^{\frac{\Delta - \xi}{\xi}}), minI\}$;
3: **return** I';

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

$$\Delta_c = \frac{C_i}{T_i} \sum_{i=n-T_i+1}^{n} E_{k_i} \tag{23}$$

  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.

$$\Delta_d = C_d \frac{E_{k_n} - E_{k_{n-1}}}{k_n - k_{n-1}} \tag{24}$$

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

The full PID algorithm we have developed so far is thus

$$\Delta = C_p E_{k_n} + \frac{C_i}{T_i} \sum_{i=n-T_i+1}^{n} E_{k_i} + C_d \frac{E_{k_n} - E_{k_{n-1}}}{k_n - k_{n-1}} \tag{25}$$

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:

$$C_p, C_i, C_d \geq 0 \tag{26}$$

$$C_p + C_i + C_d = 1 \tag{27}$$

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

Given the PID error Δ, a new sampling interval can be determined by the following equation:

$$I' = I + \theta(1 - e^{\frac{\Delta - \xi}{\xi}}) \tag{28}$$

where θ and ξ are predetermined 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 $minI$.

6. EXPERIMENT

We have implemented our apporach in Java with JSC (Java Statistical Classes [1]) for simulating the Laplace distribution. We empirically study the *predictability* and *controllability* of our proposed aprroach 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*.

[1] http://www.jsc.nildram.co.uk

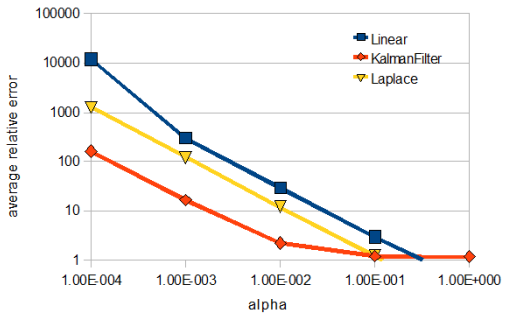

Figure 6: Prediction Accuracy with Flu data set

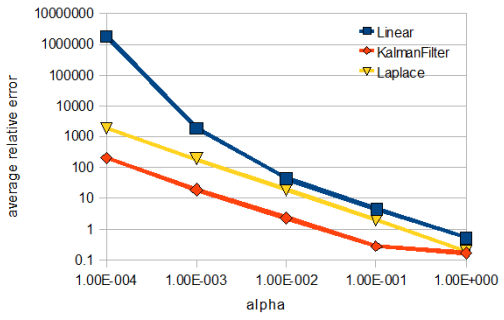

Figure 7: Prediction Accuracy with Traffic data set

- **Flu** is from the weekly surveillance data of Influenza-like illness provided by the Influenza Division of the Centers for Disease Control and Prevention [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 [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[4]. This data set contains observations from January 1972 to October 2011 with 478 data points.

6.1 Accuracy and Robustness of Prediction

We first evaluate the accuracy of the Kalman filter estimation accross all three data sets, compared against a simple linear predictor and pure Laplace noisy observations. To this end, the experiment is set up in a way such that a noisy observation is obtained at each time step and both Kalman filter and linear predictor give an estimate for the next step. This prediction is compared with the true value for an relative error and the average relative error over the entire series is plotted in the results. We experiment the performance with respect to different scales for privacy budget α. The linear predictor fits a line with 2 data points as the data sets show little linearity in a larger scale. We set the the process noisy covariance Q to be 0.001 and the measurement noise covariance R to be 10 since we assume constant state model and small measurement noise. As shown in Figure 6, 7, and 8, Kalman filter outperforms linear predictor across all three data sets especially when given small privacy budget.

6.2 Effects of Kalman Filter Parameters

To understand the impact of Q and R on the accuracy of Kalman filter estimation, we vary their values independently with *Flu* data set. Given α is 0.01, results are presented in Figure 9 and 10. In Figure 9, with R fixed, the accuracy of Kalman filter drops as Q increases. In Figure 10, given Q fixed, the accuracy improves as R increases. This can be explained by Equation (13) which defines how the Kalman gain is calculated. Increasing Q, the Kalman gain increases, resulting the *a posteriori* estimate, i.e., the final released value, favoring the noisy observation. Increasing R, the Kalman gain decreases, resulting the *a posteriori* estimate favoring the *a priori* state prediction. The results from both 9 and 10 consistently state that it's beneficial to rely more on the state prior than the noisy measurement.

[2]http://www.cdc.gov/flu/
[3]http://www.its.washington.edu/
[4]http://research.stlouisfed.org/

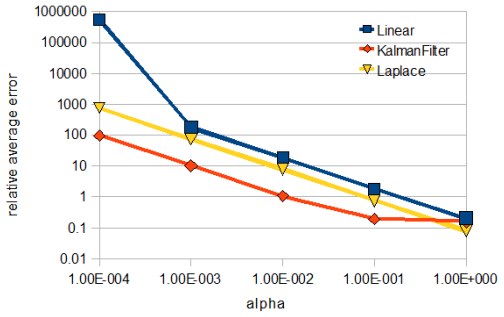

Figure 8: Prediction Accuracy with Unemployment data set

6.3 Accuracy and Robustness of Control

As the Kalman filter provides satisfactory estimations with limited privacy budget, we now examine the performance of

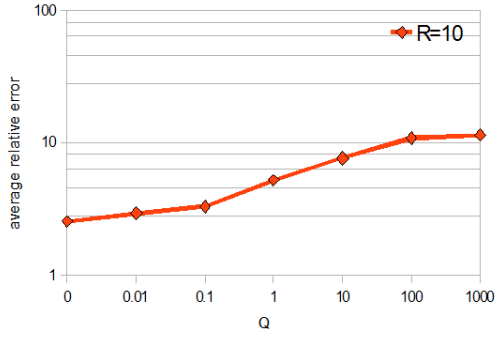

Figure 9: Impact of Q on Prediction Accuracy with Flu data set

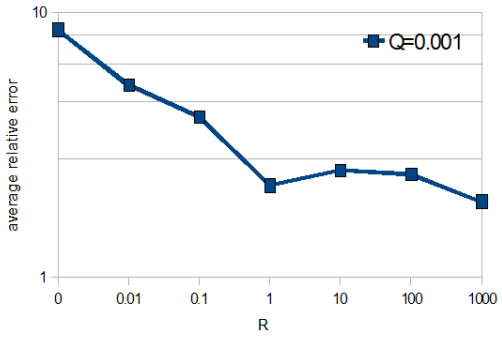

Figure 10: Impact of R on Prediction Accuracy with Flu data set

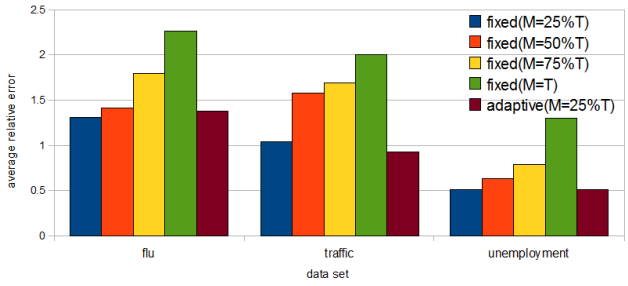

Figure 11: Accuracy of Answers

adaptive sampling with respect to fixed-rate, given the privacy budget α is 0.01. In order to set up a fixed sampling interval for Algorithm 6, we use the quatiles of T for the number of queries to issue and an extreme case where a query is issued at every time step. For our approach, we set M to be 25%T, (C_p, C_i, C_d) to be $(0.9, 0.1, 0)$, T_i to be 20, θ to be 5, and ξ to be 0.4. 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 the next section.

The results with all three data sets are shown in Figure 11. As the number of queries increases (the sampling interval drops), fixed-rate sampling suffers from a growing error. This phenominon can be interpreted by the accumulation of perturbation erros resulting from more queries. From this experiment, we found the optimal quantile for the fixed-rate sampling is 25% and adaptive sampling is comparable to the optimal fixed-rate accross all data sets.

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 12. With the ratio of M against T increasing, the average relative error grows, which, again, can be explained by the accumulation of growing perturbation errors. Figure 13 studies the impact of control gains as opposed to the integration time. We choose the control gains according to the common practice: $propotional > integral > derivative$. As seen in the plot, when the integration time increases, the resulting relative error shows no clear trend. Furthermore, we believe the variation of error is a result of the randomness of perturbation errors, as the case where $Ci = 0$ shows similar

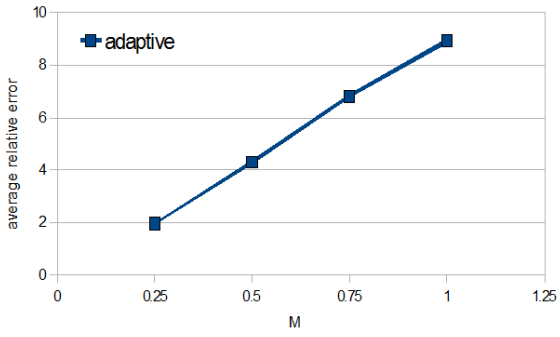

Figure 12: Impact of M on Publish Accuracy with Traffic data set

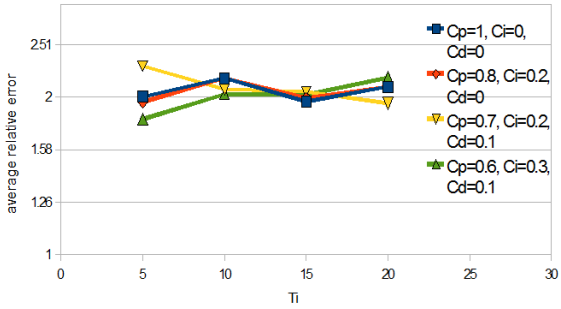

Figure 13: Impact of Cp, Ci, Cd, and Ti on Publish Accuracy with Traffic data set

variation as other cases. Therefore we conclude that there's no extra "rule of thumb" beside the common practice for tuning the controller gains in our application.

Figure 14 and 15 show the impact of ξ and θ as in Equation (28) on the accuracy. When θ is fixed to 5 and we vary the scale of ξ, the error shows little variation. The same happens when fixing ξ. Thus we consider both of them not influencial.

6.5 Overall Performance vs. Alternative Approaches

7. CONCLUSION

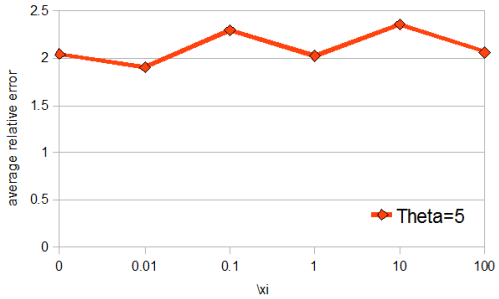

Figure 14: Impact of ξ on Publish Accuracy with Traffic data set

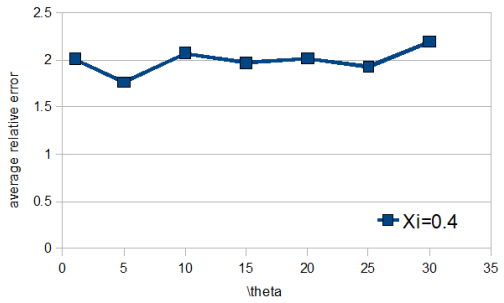

Figure 15: Impact of θ on Publish Accuracy with Traffic data set

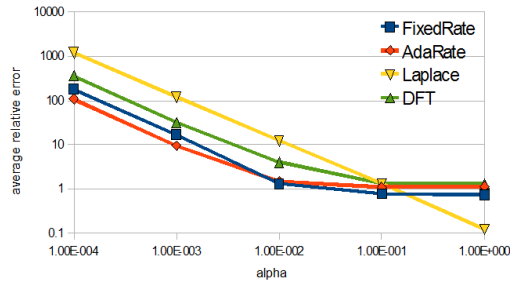

Figure 16: Comparison of Publish Accuracy with Flu data set

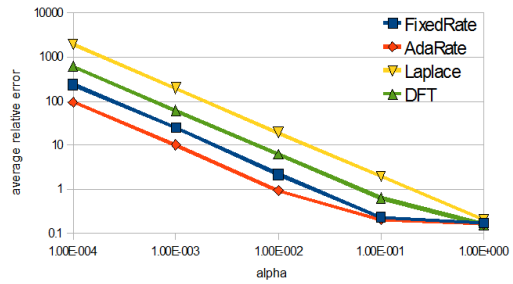

Figure 17: Comparison of Publish Accuracy with Traffic data set

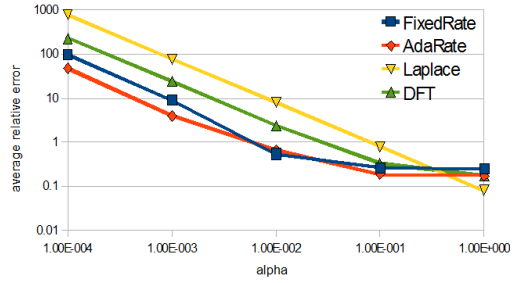

Figure 18: Comparison of Publish Accuracy with Unemployment data set

8. ACKNOWLEDGMENTS

9. REFERENCES

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

[2] 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.

[3] 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.

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

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

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

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

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

[9] J. A. Romagnoli and A. Palazoğlu. *Introduction to process control / José A. Romagnoli, Ahmet Palazoğlu*. Marcel Dekker/CRC, New York :, 2005.

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

[11] 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.

[12] 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.

9.1 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references).

APPENDIX