Monitoring Web Browsing Behavior with Differential Privacy

Liyue Fan, Luca Bonomi, Li Xiong, and Vaidy Sunderam
Dept. of Mathematics and Computer Science, Emory University
Atlanta, GA, USA
{ lfan3, lbonomi, lxiong, vss }@mathcs.emory.edu

ABSTRACT

Monitoring web browsing behavior has benefited many data mining applications, such as top-K discovery and anomaly detection. However, releasing private user data to the greater public would concern web users about their privacy, especially after the incident of AOL search log release where anonymization was not correctly done. In this paper, we adopt differential privacy, a strong, provable privacy definition, and show that differentially private aggregates of web browsing activities can be released in real-time while preserving the utility of shared data. Our proposed algorithms utilize the rich correlation of the time series of aggregated data and adopt a state-space approach to estimate the underlying, true aggregates from the perturbed values by the differential privacy mechanism. We evaluate our algorithms with real-world web browsing data. Utility evaluations with three metrics demonstrate that the quality of the private, released data by our solutions closely resembles that of the original, unperturbed aggregates.

Categories and Subject Descriptors

H.2.7 [**Database Management**]: Database Administration—*security, integrity, and protection*; H.2.8 [**Database Management**]: Database Applications—*data mining*

Keywords

Web Monitoring, Web Mining, Differential Privacy

1. INTRODUCTION

Web browsing behavior of individual users can be characterized by the sequence of visited web pages and the time of visits as they interact with the World Wide Web during each browsing session. These private browsing sessions have been extensively used by servers as well as third-party researchers to perform data mining and understand important phenomena, such as:

- finding the most popular web pages and how the trend changes over time [9, 19],
- detecting interesting, new contents on the web [13, 24] ,
- watching the web traffic for suspicious behavior or attacks [5, 12],
- ensuring websites uptime, performance, and functionality are as expected.

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2568038.

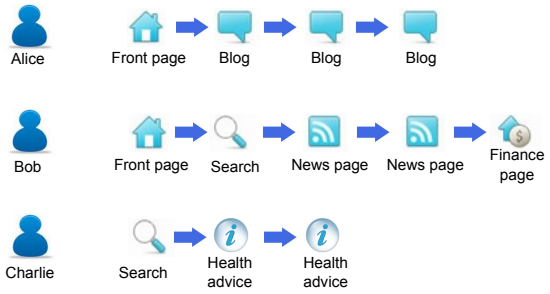

Figure 1: Examples of distinct browsing behavior

However, the release of private browsing data to third-parties or the greater public would raise user concerns from a privacy perspective. Users tend to interact with the web in an unrestrained manner, leaving behind confidential preferences and identifying information. The AOL data release in 2006 is an unfortunate privacy incident [1], where the searches of an innocent citizen were quickly re-identified by a newspaper journalist. Similarly, the web browsing behavior exhibits strong patterns and indicates sensitive information about users, such as their health, locations, connections, and finances. Therefore it is the responsibility of the service provider, i.e. data holder, to ensure that releasing this data will not compromise individual user privacy.

EXAMPLE 1. *Three browsing sessions, each associated with an individual user, are illustrated in Figure 1. Alice starts her session from the front page and successively navigates to a blog. In this session, we notice that Alice quickly leaves the front page and reaches the blog page where she persists for several consecutive time stamps, which potentially reveals that Alice may be interested in blogging activities. Bob's session shows another distinct browsing behavior compared to other users. In his session, Bob starts from the front page, performs a search, and then visits several news pages. His session ends finally after reaching the finance page. From this browsing sequence, we can infer that Bob is likely to be interested in news, especially finance related news.*

The open question is how to release web browsing data in a way that is simultaneously useful and private. Can we simply replace or remove user names or other unique identifiers? As in Figure 1, replacing user names with random identifiers would not hide the

distinctiveness or identifiability of each browsing pattern. We are in need of a strong, provable privacy guarantee.

The current state-of-the-art paradigm for privacy-preserving data publishing is *differential privacy* [2], which 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. Given the output of $\mathcal{A}$, an adversary will not be able to gain much more knowledge about any single tuple in the input, and thus privacy is protected. We will provide the formal definition of differential privacy in Section 3.

In this work, we take a first step towards sharing web browsing data with differential privacy. Rather than releasing a collection of browsing sessions, we consider the problem of releasing the number of visits to each web page at every time stamp. We adopt the formal differential privacy guarantee to protect each individual session and design two algorithms for releasing the counting data in real-time. Our contributions are summarized as follows:

(**1**) We propose a state-space approach to monitoring aggregated web browsing activities under differential privacy. The standard differential privacy mechanism injects perturbation noise into the true aggregates at every time stamp, resulting in high perturbation error in the released data. We consider the true aggregates at every time stamp as "hidden" states, and the perturbed aggregates obtained from differential privacy mechanism as "noisy" observations. Based on the known perturbation mechanism, we propose to release the posterior estimates of "hidden" states, which are statistically more accurate than the purely perturbed values, for web monitoring tasks.

(**2**) We design two algorithms based on time series state-space models to release differentially private aggregates in real-time. The first algorithm extends our previously proposed FAST framework [14] and establishes a univariate time series model for the count series of every web page. The second algorithm establishes a multivariate time series model for simultaneously monitoring the visits to all web pages and utilizes the Markov property of web navigation for accurate process modeling. The Kalman filter based posterior estimation technique is used by both algorithms to enhance the utility of released data. We propose to approximately model the Laplace perturbation noise as Gaussian and formally analyze the optimal choice of Gaussian variance in the specific web browsing setting.

(**3**) We demonstrate how to learn the model parameters for our proposed approaches from a small set of training data. We then select three utility metrics, including average relative error, precision for top-K mining, and KL-divergence, and show that the usefulness of private released data values by our methods closely resembles that of the original unperturbed data. The experiments with real-data collected from `msnbc.com` confirm that our proposed methods release useful aggregates in real-time without compromising individual privacy. We believe that they can be applied to enable a wide range of web monitoring tasks.

The rest of the paper is organized as follows: Section 2 briefly surveys the related works on privacy preserving (aggregated) user activity monitoring. Section 3 provides the problem formulation, background for differential privacy, and state-space modeling. Section 4 presents an overview of our framework, as well as the technical details of two proposed algorithms. Section 5 describes the data set and presents a set of empirical studies. Section 6 concludes the paper and states possible directions for future work.

2. RELATED WORK

In this paper we study the problem of privately monitoring web browsing activities, which is a special case of privacy-preserving time series analysis present in literature. Several works have been proposed in the last decade to tackle the private release of time series data and they differ in the privacy notion and data arrival pattern. Among these works two major research directions have been developed investigating the private release of numeric time series [23, 7, 11, 14, 22] and discrete series (event sequences) [16, 25] respectively. Furthermore, the processing of the time series can be done in an *online* and *off-line* manner. In the former setting the time series is processed in a streaming fashion, that is for every new data item arriving in the stream the algorithmic solution provides an output, while in the latter setting the computation is performed after the entire series is acquired.

2.1 Works in Differential Privacy

Differential privacy [10] has become the de facto standard for privacy preserving data analytics. Dwork et al. [10] 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 works have adopted this definition for publishing histograms [27], search logs [18], mining data streams [6], and record linkage [3].

The works of [7, 11] studied continual counting queries over data streams. These solutions provide important theoretical results in the stream setting by exploiting the relationship between the privacy level and the final utility. However, 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. Rastogi and Nath [23] also studied the problem of releasing time series of count data. They proposed an algorithm based on Discrete Fourier Transform (DFT), which guarantees differential privacy by perturbing the discrete Fourier coefficients. However, it is not applicable to real-time monitoring tasks due to the nature of Fourier transform. Furthermore, it introduces an approximation in representing the time series using only d largest Fourier coefficients.

With the technique of posterior estimation, also referred to as probabilistic inference in [26], we have proposed FAST framework in [14, 15] for releasing time series data with user-level differential privacy. The FAST approach presents two major components: sampling and filtering. The sampling component is to minimize the overall privacy cost by adaptively sampling long time series. The filtering component predicts data values at non-sampling points and corrects the perturbed values at sampling points. The FAST framework is designed to protect user-level privacy. Due to the problem setting in this paper, it is adapted to provide session-level privacy guarantee.

2.2 Other Privacy Notions

Many solutions have been developed to release time series data with privacy notions other than differential privacy. Papadimitriou et al. [22] studied the trade-off between time series compressibility and perturbation, developing two algorithms based on Fast Fourier Transform (FFT) and Discrete Wavelet Transform (DWT) respectively. The key idea of these approaches consists in processing the time series both in off-line and online fashion by perturbing the frequency of the series. Although the proposed techniques introduce additive noise to the released series, there's no formal definition or guarantee of privacy. Gőtz et al. [16] proposed a novel approach (MaskIt) for releasing user context streams with suppression. MaskIt allows user-defined sensitive contexts that are carefully checked in the released stream, with the goal of limiting the adversary's ability of learn about user sensitive contexts from the

Session	t_1	t_2	t_3	t_4	t_5	...
s_1	fp	fp				
s_2		fp	news	sports news		
s_3		tech	news	news	local	...
s_4			on-air	health	local	...
...	...	...	...	...	...	...

Table 1: Example browsing sessions

released stream. They also propose to model user movement patterns by a Markov chain. In the event stream domain, a similar notion of privacy based on suppression has been proposed in [25]. In this approach, the sensitive information is related to the presence of private patterns in the data stream. The suppression of these patterns is performed over the stream to maximize the utility for the useful non-sensitive patterns reported in the released series.

3. PRELIMINARIES

3.1 Problem Definition

Here we formally define the problem of monitoring web browsing activities with differential privacy. A browsing session is defined as a sequence of web pages browsed at consecutive, discrete time stamps, which can be obtained from the log file of page requests [1]. We consider all browsing sessions are established on a single server, where no data integration is needed across different servers. Furthermore, the sessions are dynamically established, possibly starting at different time points with variable lengths.

Let D denote the database of browsing sessions (illustrated in Table 1) and T denote the expected length of monitoring period. Without loss of generality, we assume that there are m web pages, i.e. $page_1, page_2, \ldots, page_m$, hosted on the server. An example of a browsing session of length 3 is s_2 in Table 1. In this session, the user starts from the front page ("fp" for short) at time t_2, then successively navigates to the news page at time t_3, and finally at time t_4 browses the sports news page and the session terminates there. The goal of our work is to release the number of sessions in D browsing $page_i$ at time k for each i and k, without disclosing the presence or absence of any private session. A formal definition of our problem is provided below.

PROBLEM 1 (PRIVATE WEB MONITORING). *Let x_k^i denote the number of sessions in D that browse the i-th page at time stamp k, $1 \leq k \leq T$. For every time stamp k, we are to release a sanitized count r_k^i for every i, such that the series of private releases $\{\{r_k^i\}_{i=1}^m$, for $k = 1, \ldots, T\}$ satisfies α-differential privacy.*

We further limit user browsing activities by introducing an additional parameter l_{max}, which indicates the maximum session length allowed in our problem setting. We assume $l_{max} < T$. Our consideration is three-fold: **1**) In practice, a typical browsing session would not contain unlimited number of web pages. For example, the average session length is 4.7 from data collected on `msnbc.com`, shown in Table 3. **2**) It is very common for web applications to specify a fixed time-out period such that the session automatically ends if the user does not refresh or request a page within that period. **3**) From privacy preservation point of view, if a session could contain an unbounded number of web pages, it would have an unlimited influence on the released aggregate values. As a consequence, the differential privacy mechanism, described below, would have to inject a large perturbation in order to mitigate such an influence. Based on these considerations, we set $l_{max} = 20$ later in our empirical studies.

[1]Between two successive requests, the user is assumed to browse the previous page

3.2 Differential Privacy

The privacy guarantee provided by our work is *differential privacy* [2]. Simply put, a mechanism is differentially private if its outcome is not significantly affected by the removal or addition of any record. An adversary thus learns approximately the same information about any individual record, irrespective of its presence or absence in the original database.

DEFINITION 1 (α-DIFFERENTIAL PRIVACY [2]). A non-interactive privacy mechanism $\mathcal{A}$ satisfies α-differential privacy if for any dataset D_1 and D_2 differing on at most one record, and for any possible anonymized dataset $\widetilde{D} \in Range(\mathcal{A})$,

$$Pr[\mathcal{A}(D_1) = \widetilde{D}] \leq e^{\alpha} \times Pr[\mathcal{A}(D_2) = \widetilde{D}] \quad (1)$$

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

The privacy parameter α, also called the *privacy budget* [21], specifies the degree of privacy offered. Intuitively, a lower value of α implies stronger privacy guarantee and a larger perturbation noise, and a higher value of α implies a weaker guarantee while possibly achieving higher accuracy. Two databases D_1 and D_2 that differ on at most one record are called *neighboring databases*. In our problem definition, a database "record" represents a browsing session and therefore our work is designed to protect the presence of absence of every individual session.

Laplace Mechanism. Dwork et al. [10] show that α-differential privacy can be achieved by adding i.i.d. noise to query result $q(D)$:

$$\tilde{q}(D) = q(D) + (\tilde{N}_1, \ldots, \tilde{N}_m)^\intercal \quad (2)$$

$$\tilde{N}_i \sim Lap(0, \frac{GS(q)}{\alpha}) \text{ for } i = 1, \ldots, m \quad (3)$$

where m represents the dimension of $q(D)$. The magnitude of $\tilde{N}$ conforms to a Laplace distribution with 0 mean and $GS(q)/\alpha$ scale, where $GS(q)$ represents the *global sensitivity* [10] of the query q. The global sensitivity is the maximum L1 distance between the results of q from any two neighboring databases D_1 and D_2. Formally, it is defined as follows:

$$GS(q) = \max_{D_1, D_2} ||q(D_1) - q(D_2)||_1 \ . \quad (4)$$

The following lemma establishes the sensitivity of counting web page visits in data set D for T time stamps, in order to protect the privacy of each individual session.

LEMMA 1 (SENSITIVITY FOR COUNTING PAGE VISITS). *Let D be the session database with maximum session length l_{max}. Then for a query $q(D) = \{\{x_k^i\}_{i=1}^m$, for $k = 1, \ldots, T\}$ which computes the number of sessions in D browsing $page_i$ at time k for every i and k, the sensitivity $GS(q)$ of q is at most l_{max}.*

PROOF. *For any session s in D, at most one page $page_i$ is visited by s at time k. Furthermore due to the session length constraint, over the entire time frame T, the session s can visit at most l_{max} pages (either the same page for multiple times or different web pages). Hence, by adding or removing any session s from the dataset D, we would change at most l_{max} counts of $q(D)$ output by one. Therefore, it follows that the sensitivity $GD(q)$ of q is at most l_{max}.* □

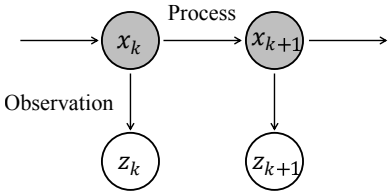

Figure 2: Illustration of State-Space Model

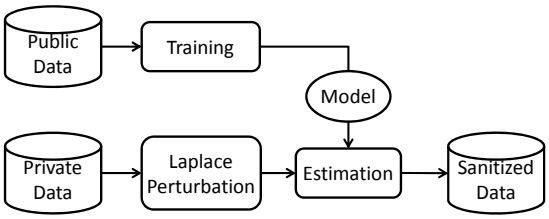

Figure 3: Proposed Framework

3.3 State-Space Model

State-space model is widely used for time series modeling and analysis. A simple illustration of the state-space model is shown in Figure 2. A time-series $\{z_k\}$ can be considered a sequence of observations made from the true, "hidden" states $\{x_k\}$ (indicated by the dark shade). The true states cannot be directly accessed and are often assumed to be generated by a known process. A linear *process model* that describes the transition between two consecutive states is as follows:

$$x_{k+1} = Ax_k + \omega_k \tag{5}$$

$$\omega_k \sim f_\omega(\cdot) \tag{6}$$

where A is the time-invariant coefficient and ω_k represents the noise of the linear model. Commonly, ω_k is assumed to be a white noise which follows a time-invariant distribution $f_\omega(\cdot)$. The process model relates the current system state x_{k+1} to the previous state x_k. The intuition behind the linear process model is that the current system state is expected to be linearly correlated to the previous state, except for some random process disturbance.

The observed time series $\{z_k\}$ are obtained from the true states and is considered to contain additive measurement noise. The following *measurement model* serves as an example of how an observation z_k can be obtained:

$$z_k = Hx_k + \nu_k \tag{7}$$

$$\nu_k \sim f_\nu(\cdot) \tag{8}$$

where H is the linear coefficient and ν_k represents the additive measurement noise. For simplicity, ν_k is often assumed to follow a time-invariant distribution with probability density function f_ν. The measurement model relates the observation z_k to the true state x_k. In traditional engineering context, f_ν depends on the measurement equipment/process and varies from system to system. In our problem setting, we consider the series of raw aggregates as the underlying states $\{x_k\}$ and the perturbed aggregates by the Laplace mechanism as the observations $\{z_k\}$. Therefore, we set $f_\nu = Lap(0, \frac{l_{max}}{\alpha})$ in privacy preserving monitoring scenario.

The state-space model of time series data is widely used to estimate the true state x_k from the observation z_k in the presence of noises, knowing the model parameters A and H, and the noise distributions f_ω and f_ν. However, there is no computationally efficient method for posterior estimation when either ω_k or ν_k is non-Gaussian. Below we outline two efficient filtering algorithms with Gaussian approximation of the measurement noise ν_k. We will examine the possibility of approximately modeling the Laplace measurement noise with a white, Gaussian noise in Section 4.

4. PROPOSED SOLUTIONS

The goal of our work is to enable the data holders to share useful aggregates for web monitoring applications, while preserving privacy. There are three key components to achieving this goal. First of all, the privacy of individual sessions should be protected with strong guarantee. Our solutions adopt differential privacy to protect every session, despite possible background knowledge known to the adversary. Secondly, the differential privacy mechanism injects perturbation noise into every released aggregate. Therefore, a post-processing, i.e. estimation, component is used in our system in order to recover the perturbed values and to improve the utility of released aggregates. We propose two approaches to estimate the aggregates for each web page separately or all web pages simultaneously. Thirdly, the dynamics of the underlying aggregate time series should be utilized in order to perform accurate estimation. Therefore, we propose to learn the state-space models with publicly available data, which can come from historical data or users who consent disclosure. We will demonstrate that our solutions learn accurate models and achieve high utility in the released data, even with a small training set.

The system framework of monitoring web browsing activities with differential privacy is depicted in Figure 3. At every time stamp, the statistics of browsing sessions can be obtained by aggregating the **private user data**. The raw aggregates will then be perturbed by the **Laplace Perturbation** mechanism to ensure the pre-defined level of differential privacy. The perturbed aggregates are then used by the **Estimation** module to derive posterior estimates of the original aggregates. Based on which approach is in use, the posterior estimate can be derived for each web page separately or for all web pages simultaneously. The state-space model used in posterior estimation can be learned through the **Training** component with publicly available data. Based on the method in use, we can learn a state-space model for each activity separately or an all-in-one state-space **model** for all web pages. The **sanitized output data** is the posterior estimates released in real-time, which are statistically more accurate than the purely perturbed values. The detailed algorithms are described below.

4.1 Univariate Time Series Approach

The first solution to monitoring web activities with differential privacy is to establish a univariate state-space model for the *count* series of each web page and estimate the true states from the perturbed values. We adapt our previously proposed FAST framework [14] to provide session-level privacy and apply the Kalman filter based estimation algorithm to each univariate *count* series.

For each web page i, we establish the following process model for the time series $\{x_k^i, \text{for } k = 1, \ldots, T\}$:

$$x_{k+1}^i = x_k^i + \omega_k^i \tag{9}$$

$$\omega_k^i \sim \mathbb{N}(0, Q^i) \tag{10}$$

where ω_k^i represents the process noise for the ith page. For simplicity, ω_k^i is often assumed to be a white Gaussian noise with variance Q^i. Similarly, the measurement model for the Laplace perturbed

value z_k^i is established below:

$$z_k^i = x_k^i + \nu_k^i \tag{11}$$

$$\nu_k^i \sim Lap(0, \frac{l_{max}}{\alpha}) \tag{12}$$

where ν_k^i represents the Laplace perturbation noise added to the count of page i at time k in order to protect individual sessions. Note that since ν_k^i is added to each page i in parallel at every time stamp k, it follows the same distribution regardless of i and k.

We have studied the posterior estimation challenge and found that the posterior distribution cannot be analytically determined when $f_\nu(\cdot)$ is non-Gaussian. It has been reported in [14] that it is sufficient to approximately model ν_k^i as a Gaussian noise for posterior estimation purpose and thus the Kalman filter algorithm can be adopted to efficiently compute the state estimates. Specifically, the following Gaussian distribution with mean 0 and variance R was proposed:

$$\nu_k^i \sim \mathbb{N}(0, R) \tag{13}$$

In this work, we adopt the same approximation in Equation (13) in order to utilize computational attractive structure of the Kalman filter for posterior estimation.

Estimation algorithm. The Kalman filter [17] is a recursive method that provides an efficient means to estimate the state of a linear Gaussian process, by minimizing the variance of posterior error. Below we outline the recursive Kalman filter mechanism for our proposed solution based on univariate time series models.

For each web page i, given the state-space models defined in Equation (9-11, 13), our algorithm recursively performs two steps: *Prediction* and *Correction*, at every time stamp k.

The $Prediction$ step is described in Algorithm 1. "ˆ" represents state estimates and the superscript " $^-$ " represents variables related to the prior estimate. In Line 1, a prior estimate $\hat{x}_k^{i-}$ is derived from the posterior estimate of time $k-1$, according to the constant process model in Equation (9). P_k^{i-} denotes the prior error variance and is defined as follows.

$$P_k^{i-} = E[(x_k^i - \hat{x}_k^{i-})(x_k^i - \hat{x}_k^{i-})^\intercal] \tag{14}$$

Line 2 updates P_k^{i-} according to the distribution of the process noise ω_k^i.

Upon receiving a perturbed value from the Laplace perturbation mechanism, the $Correction$ step, as in Algorithm 2, is performed to derive a posterior estimate of the true state. In Line 1, the Kalman gain K_k^i is updated. In Line 2, the posterior estimate $\hat{x}_k^i$ is obtained by linearly combining the prior estimate and the noisy measurement z_k^i. Line 3 further updates the posterior error variance P_k^i, which is defined as follows:

$$P_k^i = E[(x_k^i - \hat{x}_k^i)(x_k^i - \hat{x}_k^i)^\intercal] \tag{15}$$

Note that the Kalman gain K_k^i in Line 1 is derived by minimizing the posterior error variance P_k^i. We refer interested readers to the seminal work by R.E. Kalman [17] for details about the derivation of the Kalman gain.

The overall solution based on univariate time series state-space model is summarized in Algorithm 3. At every time stamp k, for each web page i, a prior estimate is derived from the prediction procedure according to the process model, while a posterior estimate is derived by combining the prediction and the noisy observation in the correction procedure. At the first time stamp, i.e. $k = 1$, the perturbed value z_1^i is released for initialization. The advantage of the univariate solution is its efficiency in computing the minimum variance estimate for linear Gaussian problems. As can be seen, the required computation time is linear of the number of web pages, i.e. $\mathcal{O}(m)$, per time stamp. The process can be easily parallelized, since each web page is modeled and processed separately. We will further study the run time performance of this approach in the experiment section.

Algorithm 1 Prediction(i,k)

Input: Previous posterior estimate x_{k-1}^i
Output: Prior estimate $\hat{x}_k^{i-}$

1: $\hat{x}_k^{i-} = \hat{x}_{k-1}^i$
2: $P_k^{i-} = P_{k-1}^i + Q^i$

Algorithm 2 Correction(i,k)

Input: Perturbed count z_k^i
Output: Posterior estimate $\hat{x}_k^i$

1: $K_k^i = P_k^{i-}(P_k^{i-} + R)^{-1}$
2: $\hat{x}_k^i = \hat{x}_k^{i-} + K_k^i(z_k^i - \hat{x}_k^{i-})$
3: $P_k^i = (1 - K_k^i)P_k^{i-}$

Parameters. The process noise variance Q^i is in general difficult to determine by observing the process. In practice, it is usually set by off-line tuning. We will show how to tune Q^i for each web page i in the experiment section. As for the approximate Gaussian noise variance R, we conducted analysis on posterior error variance and the result is stated below.

THEOREM 1 (OPTIMAL APPROXIMATION). Given the perturbation noise distribution $Lap(0, l_{max}/\alpha)$ at every time stamp, using an approximate Gaussian noise that follows $\mathbb{N}(0, R)$ leads to the following posterior error:

$$var(x_k^i - \hat{x}_k^i) = \frac{R^2[var(x_{k-1}^i - \hat{x}_{k-1}^i) + Q^i]}{(P_k^{i-} + R)^2} + \frac{2(P_k^{i-} l_{max})^2}{(P_k^{i-} + R)^2 \alpha^2} \tag{16}$$

and optimal posterior estimation requires $R \propto \frac{l_{max}^2}{\alpha^2}$.

PROOF. See Appendix A. □

On the other hand, the error covariance P_k^i and the Kalman gain K_k^i will stabilize quickly when Q^i and R are constant, regardless of the initial settings. Therefore, we adopt the following initialization $K_1^i = 0$ and $P_1^i = R$ in our experiments.

Privacy. The privacy guarantee of univariate algorithm is stated in the following theorem.

THEOREM 2 (PRIVACY GUARANTEE). *Algorithm 3 satisfies α-differential privacy.*

PROOF. *By definition of the Laplace perturbation mechanism and sensitivity analysis in Lemma 1, the perturbed data values $\{z_k^i$, for $i = 1, \ldots, m\}$ satisfy α-differential privacy. Since neither* **Prediction** *nor* **Correction** *interact with the raw aggregates, there is no extra privacy leakage incurred by those two procedures.* □

4.2 Multivariate Time Series Approach

As web browsing sequences exhibit strong navigation patterns between adjacent web page requests, it has been shown that user navigation patterns can be well captured by first-order Markov chain [4]. In order to incorporate this rich spatio-temporal correlation, we

Algorithm 3 Univariate Time-Series Algorithm(k)

Input: Raw counts $\{x_k^i$, for $i = 1, \ldots, m\}$, privacy budget α
Output: Private, released counts $\{r_k^i$, for $i = 1, \ldots, m\}$

1: **for** $i = 1, \ldots, m$, **do**
2: $prior \leftarrow$ **Prediction**(i, k)
3: $z_k^i \leftarrow$ perturb x_k^i by $Lap(\frac{l_{max}}{\alpha})$
4: $posterior \leftarrow$ **Correction**(i, k)
5: $r_k^i \leftarrow posterior$

propose the following method which establishes a multi-variate time series model and at each time stamp releases the counting histograms of all web pages at once.

The multivariate time series process model which utilizes the navigation Markov chain is described below:

$$\mathbf{X_{k+1}} = \mathbf{M}\mathbf{X_k} + \omega_\mathbf{k} \tag{17}$$

$$\omega_\mathbf{k} \sim \mathbb{N}(\mathbf{0}, \mathbf{Q}) \tag{18}$$

Each state in the multivariate time series is a vector, i.e. $\mathbf{X_k} = (x_k^1, \ldots, x_k^m)^\intercal$. The linear coefficient $\mathbf{M}$ is represented as a m-by-m Markov transition matrix and is defined as follows:

$$\mathbf{M} = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots \\ p_{2,1} & p_{2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{19}$$

where each element $p_{i,j}$ represents the probability of transitioning to the *ith* page from the *jth* page.

The process noise is also a vector, i.e. $\omega_\mathbf{k} = (\omega_k^1, \ldots, \omega_k^m)^\intercal$, where ω_k^i represents the process noise of the ith page at time stamp k. By assuming each ω_k^i follows a white, time-invariant Gaussian distribution and all ω_k^i's are mutually independent, we can derive that $\omega_\mathbf{k}$ is also white Gaussian and its covariance matrix $\mathbf{Q}$ is the diagonal, i.e.

$$\mathbf{Q} = \begin{pmatrix} Q^{1,1} & 0 & \ldots & 0 \\ 0 & Q^{2,2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & Q^{m,m} \end{pmatrix} \tag{20}$$

where each $Q^{i,i}$ is a positive scalar value that represents the variance of process noise ω_k^i in the multivariate model. Note that $Q^{i,i}$ might be different from Q^i as in the univariate model, since the multivariate model captures the spatio-temporal correlation of multiple aggregates.

Similarly, the multivariate measurement model is established as follows:

$$\mathbf{Z_k} = \mathbf{X_k} + \nu_\mathbf{k} \tag{21}$$

$$\nu_\mathbf{k} \sim \mathbb{N}(\mathbf{0}, \mathbf{R}) \tag{22}$$

where $\mathbf{Z_k}$ is the vector of perturbed values at time k, i.e. $\mathbf{Z_k} = (z_k^1, \ldots, z_k^m)^\intercal$. $\nu_\mathbf{k}$ stands for the vector of independent perturbation noise, i.e. $\nu_\mathbf{k} = (\nu_k^1, \ldots, \nu_k^m)^\intercal$, where each noise follows the time-invariant Laplace distribution $Lap(0, \frac{l_{max}}{\alpha})$. For efficiency, we approximately model $\nu_\mathbf{k}$ as a white Gaussian noise with covariance matrix $\mathbf{R}$.

Algorithm 4 Multivariate Time-Series Algorithm(k)

Input: Raw counts $\{x_k^i$, for $i = 1, \ldots, m\}$, privacy budget α
Output: Private, released counts $\{r_k^i$, for $i = 1, \ldots, m\}$

Prediction
1: $\hat{\mathbf{X}}_\mathbf{k}^- = \mathbf{M}\hat{\mathbf{X}}_{\mathbf{k-1}}$
2: $\mathbf{P}_\mathbf{k}^- = \mathbf{M}\mathbf{P}_{\mathbf{k-1}}\mathbf{M}^\intercal + \mathbf{Q}$
Perturbation
3: $\mathbf{Z_k} \leftarrow$ perturb $\mathbf{X_k}$ by $Lap(\frac{l_{max}}{\alpha})^m$
Correction
4: $\mathbf{K_k} = \mathbf{P}_\mathbf{k}^-(\mathbf{P}_\mathbf{k}^- + \mathbf{R})^{-1}$
5: $\hat{\mathbf{X}}_\mathbf{k} = \hat{\mathbf{X}}_\mathbf{k}^- + \mathbf{K_k}(\mathbf{Z_k} - \hat{\mathbf{X}}_\mathbf{k}^-)$
6: $\mathbf{P_k} = (\mathbf{I} - \mathbf{K_k})\mathbf{P}_\mathbf{k}^-$
7: $\{r_k^i\} \leftarrow \hat{\mathbf{X}}_\mathbf{k}$

The perturbation noises are added independently to web pages. Therefore, the covariance matrix $\mathbf{R}$ is also diagonal:

$$\mathbf{R} = \begin{pmatrix} R^{1,1} & 0 & \ldots & 0 \\ 0 & R^{2,2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R^{m,m} \end{pmatrix} \tag{23}$$

where each $R^{i,i}$ is a positive scalar value that represents the variance of measurement noise ν_k^i in the multivariate model.

Estimation algorithm. The estimation algorithm based on multivariate time-series model is summarized in Algorithm 4. As can been seen, similar computation structure based on the Kalman filter is adopted. At every time stamp k, a posterior estimate $\hat{\mathbf{X}}_\mathbf{k}$ is released for monitoring applications, which contains the estimated counts at time k for each web page.

From the computational aspect, the multivariate time series approach includes four matrix-matrix multiplications as well as one matrix inversion at every time stamp. The computation complexity at every time stamp is therefore $\mathcal{O}(m^3)$, due to the matrix computations. We will empirically study its runtime performance in the experiment section.

Parameters. In the Markov matrix $\mathbf{M}$, each entry $p_{i,j}$ represents the probability of transitioning to the ith page from the jth page. We propose to estimate each $p_{i,j}$ as follows:

$$p_{i,j} = \frac{\#(page_j, page_i)}{\#page_j} \tag{24}$$

where $\#page_j$ is the number of occurrences of page j and $\#(page_j, page_i)$ is the number of occurrences of page i immediately following page j. The number of occurrences can be obtained by counting support from the browsing sequences in the training set.

The noise covariance matrix $\mathbf{Q}$ can be also learned by off-line tuning. Unlike the univariate approach, the tuning of covariance matrix $\mathbf{Q}$ has an exponential search space, since we need to simultaneously set all diagonal elements. We adopt a similar approach to Yan et. al [28] which utilizes the computation structure of genetic algorithm (GA) for parameter optimization. The details are provided in the experiment section.

As for the approximate measurement noise $\mathbf{R}$, since the actual measurement noise is only determined by the Laplace perturbation mechanism, we set $R^{i,i} = R$ for every i as in the univariate model in our experiments.

Privacy. The privacy guarantee of multivariate algorithm is stated in the following theorem.

Symbol	Description	Default Value
α	Total Privacy Budget	1
R	Gaussian Noise Variance	$40,000$
m	Number of Web Pages	18
T	Monitoring Time	100
l_{max}	Max Session Length	20

Table 2: Default parameter settings

Dataset	MSNBC
sessions	989,818
categories	17
longest session length	14,975
average session length	4.7

Table 3: MSNBC data set summary

THEOREM 3 (PRIVACY GUARANTEE). *Algorithm 4 satisfies α-differential privacy.*

PROOF. *Similar to the proof of Theorem 2. Omitted.* □

5. EXPERIMENTS

Here we present a set of empirical studies conducted a simulation of dynamic web browsing behavior generated from real-world data. In each study, we compare the following four methods: 1) U−KF, i.e. our univariate Kalman filter approach as an extension of the FAST framework [14], 2) M−KF, i.e. our multivariate Kalman filter approach which incorporates the rich spatio-temporal correlation in the process model, 3) LPA, i.e. the baseline method that releases the Laplace perturbed value at every time stamp, which is applied to each univariate times series, and 4) DFT, i.e. the Fourier transformation based algorithm [23], applied to each univariate time series separately in an off-line manner.

The default settings of parameters required by the above methods are shown in Table 5, except for the process noise parameters Q^i and $Q^{i,i}$ for every i and the markov transition matrix $\mathbf{M}$, which can be learned from the training data. Note that the number of web pages m is 18, which includes all the actual web pages from MSNBC data set and an inactive status "$" introduced by us on purpose. We preserve $d = 20$ Fourier coefficients for the DFT method, as suggested by the authors [23].

5.1 Simulating Dynamic Browsing Sessions

We believe that empirical evaluations should be conducted in a practical setting in order to demonstrate the usability of proposed methods in solving real problems. In the absence of raw log files with the finest page and time granularity, we propose to simulate the dynamic browsing behavior with the Poisson process and anonymous, real-world session data.

As the data pool for our simulation, we consider the MSNBC anonymous web dataset at the UCI Machine Learning Repository. The MSNBC data, summarized in Table 3, contains nearly 1 million anonymous browsing sessions collected over a period of twenty-four hours on the msnbc.com domain. All web pages were classified into 17 categories, and each session in MSNBC data set records a sequence of category requests with variable length.

In our simulated data set, we consider a time frame of $T = 100$ time stamps, where at each time stamp a number of new sessions start and some existing ones may end. At time $t = 1$ we start by randomly sampling $S_{start} = 100000$ sessions from the candidate set. Successively at every new time stamp, we randomly sampled S_{new} sessions from the candidate set, where S_{new} is a random variable drawn from a Poisson distribution with mean 10000. This choice is motivated by the fact that the Poisson process has been commonly used in modeling user page request rate for web browsing [20, 8] and for video-on-demand systems [29]. In particular, we use the same methodology as in Yu et al. [29], where S_{new} is upper-bounded by $N = 20000$ which represents the maximum number of new sessions that the server can handle at any time. For every session drawn from the candidate set, prior to adding to our simulated data set, it is first truncated if needed to contain up to l_{max} web pages. Then it is padded with a special symbol "$", which indicates being inactive, at the beginning as well as in the end, such that the total number of symbols is T. For each session starting at time stamp k, $k-1$ $'s are added at the beginning. A proper number of $'s are added to the end of each session based on the actual session length. As a result, we generated $1,089,281$ dynamic browsing sessions, where the start of each session is indicated by the first non-$ symbol in the sequence.

To estimate model parameters for our proposed methods, we created a training data set with a small percentage, i.e. 5%, randomly sampled from the simulated data set. For evaluation purpose, we randomly sample 100 test data sets $\{D_1, D_2, ..., D_{100}\}$, each containing 10% of the simulated data set. Average results obtained from $\{D_1, D_2, ..., D_{100}\}$ are reported in our evaluations.

5.2 Learning Models

Below we describe how to estimate the model parameters from the training data. As we introduced $ in simulating the browsing behavior to indicate the inactive states, we treat $ as a web page and learn the process noise for its count series as well as the transition probabilities for our proposed methods.

For the univariate approach, we can learn the dynamics of the count series for each web page i, i.e. the process noise variance Q^i, from the training data set. For each web page i, we aggregated its count series from the training set and tuned Q^i to minimize the posterior estimate error. Since each Q^i is a real value and implies a large search space, we specified a search domain comparing only different orders of magnitude, i.e. $\{10^{-4}, 10^{-3}, \ldots, 10^{9}\}$, in order to speed up the training process. For every i and each setting of Q^i, we ran the univariate approach 50 times to overcome the randomness of the perturbation noise and the value which resulted in minimum average relative error, as defined in Equation (25), for posterior estimates was preserved for real-time monitoring.

For the multivariate approach, we can learn the transition probabilities $p_{i,j}$ for any web page pair i and j from the training set, as in Equation (24). Note that we can also learn the transition probability from the inactive status $ to any web page, which indicates the likelihood of starting a new session and which web page it starts from. As for the noise covariance matrix $\mathbf{Q}$, the same search domain as above was specified for each element $Q^{i,i}$. We implemented the genetic algorithm (GA) with random initial solutions and the population size 50, and ran it for 20 iterations. The fitness value is defined as the average relative error (over 50 runs) of the posterior estimates generated by the multivariate approach, compared to the raw count series aggregated from the training set. The best solution generated by GA algorithm was preserved for real-time monitoring.

5.3 Utility Evaluation

The goal of our work is to share useful statistics of on-line browsing behavior in order to perform monitoring tasks. We compare the utility of data released by our proposed methods, i.e. U−KF and M−KF, against existing approaches, i.e. LPA and DFT. Note that the DFT is an off-line method and therefore cannot be applied to

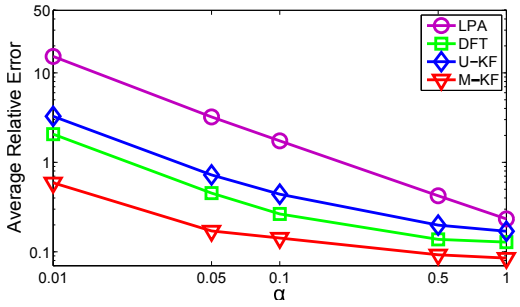

Figure 4: Comparison of average relative errors

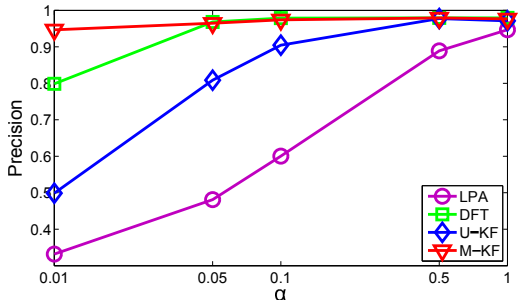

Figure 5: Comparison of top-K mining

real-time monitoring tasks. It is only included in our experiments for reference and comparison.

We adopt three different utility metrics in the following studies, including both generic metrics as well as application-specific metrics. For each set of evaluations, we further study the trade-off between utility and privacy for each method by varying the privacy budget, i.e α value. The usual range of α adopted by most other works in differential privacy is between 0.1 an 1. However, we choose a larger range in our experiments including smaller α values, i.e. 0.01 and 0.05, to demonstrate the applicability of all four methods when the privacy requirement is high.

5.3.1 Average Relative Error

In the first set of empirical evaluations, we consider to measure the Average Relative Error (ARE) between the released count series and the original count series. In short, ARE is a widely used metric which measures how well the released time series $\{r_k^i\}$ follows the original series $\{x_k^i\}$ for every $i = 1, \ldots, m$. It is a generic metric to evaluate data accuracy, disregarding the actual, domain-specific applications. More formally, we define the ARE error as follows:

$$ARE = \frac{1}{mT} \sum_{i=1}^{m} \sum_{k=1}^{T} \frac{|r_k^i - x_k^i|}{\max\{x_k^i, \delta\}} \tag{25}$$

where $\delta = 1$ by default, in order to handle the special case when x_k^i is zero.

As in the above definition, the ARE value provides an indication about the quality of the overall released time series, where smaller values of ARE imply higher similarity between the released and the original series, hence higher utility. We ran all four methods under different privacy budgets and the corresponding ARE values are reported in Figure 4. We observe that for every approach the ARE drops as the privacy budget increases. This is due to reduced perturbation error introduced by the differential privacy mechanism.

The baseline approach `LPA` which directly releases perturbed values results in the highest ARE error in every privacy setting, due to the perturbation noise. With relatively strong privacy requirement, i.e. $\alpha = 0.01$, the `LPA` algorithm results in large relative error which is more than 10 times higher than that of our proposed method `M-KF`. The `U-KF` method and the `DFT` method show similar results for all privacy settings. However, the former releases aggregates in real-time, while the latter requires an off-line processing of the time series due to the Fourier transform. Our proposed algorithm `M-KF` turns out to be superior and constantly outperforms all other methods, resulting in the lowest ARE error with real-time release of private data. `M-KF` yields to 59% error when $\alpha = 0.01$ and 8% error when $\alpha = 1$, while `DFT` results in more than 200% error when $\alpha = 0.01$ and 13% error when $\alpha = 1$.

5.3.2 Top-K Mining

A fundamental application of monitoring web browsing behavior is top-K mining, which aims to find the K most popular web pages visited at every time stamp. Therefore, the ability to preserve the most popular pages in the private, released data values is an important indicator of the solution usability. In the next set of experiments, we perform top-K mining at every time stamp on the released data by all the methods and report the Average Precision (AP) over the entire monitoring time period. We define the average precision as follows:

$$AP = \frac{1}{T} \sum_{k=1}^{T} TPR_k \tag{26}$$

where TPR_k represents the true positive rate of the top-K pages discovered from the private, released data at time k. Apparently, a higher value of AP indicates higher utility, since it reflects the capability for more accurate discovery of most visited web pages at any time stamp. We ran all four methods with different privacy budget values and plot the average precision for mining top-5 web pages in Figure 5. Similar trends can be observed when experimenting with different K values. Thus we omit those results here for brevity.

For all four methods, the average precision is raised as the privacy budget α increases. The baseline `LPA` again offers the worst mining utility in the shared data in every privacy setting, preserving only 33% of top-5 web pages when α is small, i.e.$\alpha = 0.01$, due to the random perturbation. Our univariate approach `U-KF` falls behind the off-line method `DFT` and the multivariate approach `M-KF` until the privacy budget is large enough, i.e. $\alpha \geq 0.5$, due to the individual state-space modeling for each web page. However, we observe that `U-KF` is still applicable and it yields 80% precision with $\alpha = 0.05$. The off-line `DFT` method yields 80% precision when $\alpha = 0.01$ and provides comparable utility to our multivariate approach `M-KF` when $\alpha \geq 0.05$. Again the `M-KF` method is proved to be superior to all the other methods, providing 95% precision in top-K mining even under very small privacy budget, i.e. $\alpha = 0.01$.

5.3.3 Distributional Similarity

In this set of experiments, we consider the count values at every time stamp, i.e. $\{x_k^i, \text{for } i = 1, \ldots, m\}$ and $\{r_k^i, \text{for } i = 1, \ldots, m\}$, as distributions of user sessions over the domain of web pages and propose to evaluate the distributional similarity between the released counts and the original counts at all time stamps. The intuition behind is that the session distribution over the domain of web pages enables understanding of the relative popularity of any web page at any time stamp. Thus measuring the distributional similarity would provide a comprehensive view of the utility of data released over the entire web page domain.

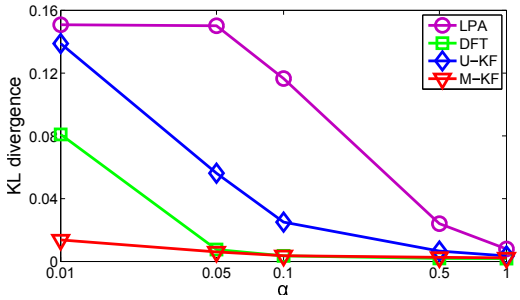

Figure 6: Comparison of distributional similarity

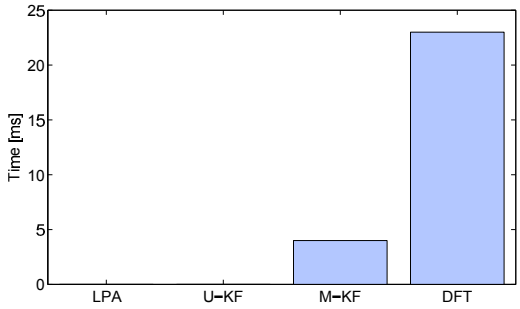

Figure 7: Comparison of runtime performance

A common metric widely used to measure the distance between two probability distributions is the KL-divergence. It is a non-symmetric measure that computes the information lost when a proposed distribution is used to estimate a true distribution. In our scenario, the estimate distribution at time stamp k comes from the released data values $\{r_k^i, \text{for } i = 1, \ldots, m\}$, while the true distribution comes from the original count values $\{x_k^i, \text{for } i = 1, \ldots, m\}$. We normalized the data values at every time stamp and denote the corresponding distributions as $\{\tilde{x}_k^i\}$ and $\{\tilde{r}_k^i\}$. Therefore, the average KL-divergence of the released time series $\mathbf{R}$ with respect to the original data series $\mathbf{X}$ can be defined as follows:

$$D_{KL}(\mathbf{X}\|\mathbf{R}) = \frac{1}{T}\sum_{k=1}^{T}\sum_{i=1}^{m} \ln\left(\frac{\tilde{x}_k^i}{\tilde{r}_k^i}\right)\tilde{x}_k^i \tag{27}$$

It reports the average KL-divergence of the released distributions $\{\tilde{r}_k^i\}$ with respect to the true distribution $\{\tilde{x}_k^i\}$ at all time stamps. Intuitively, the smaller the average KL-divergence is, the more similar the released distributions are to the original distributions.

All four methods were run under different privacy settings and the average KL-divergence results are shown in Figure 6. The baseline LPA yields highest KL-divergence among all methods in every privacy setting. Due to the randomness of the perturbation noise, the data released by LPA fails to preserve the distributional similarity with respect to the original data. Our univariate approach U−KF does not show clear advantage over LPA when $\alpha = 0.01$, due to the separate modeling of each web page. However, it can be seen that U−KF quickly catches up with DFT and the multivariate approach M−KF when $\alpha \geq 0.1$. Again, the M−KF method provides the best utility in every privacy setting, preserving the distributional properties in the private, released data values even when the privacy budget is small, i.e. $\alpha = 0.01$. We can see that M−KF yields a four times smaller divergence compared to the off-line DFT method when $\alpha = 0.01$, thanks to its accurate, multivariate model.

5.4 Runtime

An important aspect of the methods that release data for monitoring tasks is the processing time of shared data. Here we empirically compare all four methods in terms of the total running time for releasing private data values over T time stamps. Note that we exclude the training time for learning the model parameters since it is a one-time cost and usually done off-line. In addition, the processing time for each time stamp can be easily estimated from the total running time.

All four methods were run under the default parameter settings. Their testing time (training time excluded) was recorded and the average runtime in milliseconds over all test data sets is reported in Figure 7. As can be seen, the off-line method DFT, with overall complexity $\mathcal{O}(mT^2)$ [2], turns out to be the most expensive compared to other real-time methods. We observe that DFT takes 23 milliseconds to release aggregated data for $m = 18$ web pages over $T = 100$ time stamps. Our multivariate approach M−KF, which requires matrix multiplications, additions, and inversions, has overall complexity $\mathcal{O}(m^3T)$. As is shown, M−KF only takes 4 milliseconds, one sixth of DFT runtime, to release the same amount of data. We believe M−KF is highly applicable in our problem setting, especially when $m \ll T$. Both our univariate approach U−KF and the baseline LPA have linear complexity, i.e. $\mathcal{O}(mT)$. As in the empirical results, both LPA and U−KF take in-significant amount of time which is measured as zero. We conclude that compared to the baseline LPA, our U−KF method achieves good amount of utility improvement with no additional computational cost, while our M−KF method greatly improves utility with moderate additional computational cost. We believe that our proposed methods can be applied to sharing private statistics in real-time, without compromising the outcome of web monitoring applications.

6. CONCLUSION AND FUTURE WORKS

We took a first step towards releasing web browsing data for monitoring tasks with differential privacy and proposed two algorithms which release real-time statistics that guarantee session-level privacy. Our solutions utilize the rich correlation of the time series of aggregated data and establish univariate or multivariate state-space models to describe the underlying correlation. We have shown that the correlations can be learned from a small set of publicly available data and the learned models are accurate and applicable to a larger set of unknown, private data.

To improve the utility of released data, out solutions release the posterior estimates of the true aggregates, which are statistically more accurate than purely perturbed values. The posterior estimation algorithm is based on the Kalman filter and we formally analyzed the optimal choice for approximately modeling the Laplace perturbation noise with a Gaussian noise. We proved that both the univariate approach and the multivariate approach satisfy α-differential privacy. We further analyzed the time complexity and empirically compared the runtime efficiency.

We evaluated our solutions in comparison to a baseline method as well as an off-line method based on Fourier transform. Three utility metrics, including average relative error, precision for top-K mining, and KL-divergence, were selected to examine the utility of released aggregates. The results show that the utility of private, released data by our solutions closely resembles that of the original, unperturbed aggregates. We conclude that our solutions are highly

[2]In general, the Discrete Fourier Transform requires $\mathcal{O}(T^2)$ complex multiplications and additions for a time series of length T.

applicable to web monitoring tasks while providing a rigorous privacy guarantee.

Finally, there are a few aspects to our proposed approaches that can be explored in the future. The first potential aspect is the possibility to combine user browsing requests to different websites/servers. Browsing across platforms is very common in the real world, as users often switch between search engines and shopping sites or social networks. Combining the browsing requests to different servers which are made within a short time window could provide a broader view of user navigation patterns and thus enable more data mining applications. The second aspect, i.e. the scalability of the multivariate method, becomes an immediate challenge when the domain of web pages is large. As the m value increases, the multivariate state-space method yields high time complexity, i.e. $\mathcal{O}(m^3)$, due to matrix operations. Classic techniques that exploit matrix properties, such as sparsity, rank, and decomposability, can be utilized to reduce the computation requirement at the cost of accuracy. Last but not least, we can consider other probabilistic models and different inference methods, in order to find the most suitable techniques for web browsing behavior.

Acknowledgment

This research is supported by NSF under grant CNS-1117763 and AFOSR DDDAS program under grant FA9550-12-1-0240.

7. REFERENCES

[1] M. Barbaro and T. Zeller. A face is exposed for aol searcher no. 4417749. *The New York Times*, Aug. 2006.

[2] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 609–618, New York, 2008. ACM.

[3] L. Bonomi, L. Xiong, and J. J. Lu. Linkit: privacy preserving record linkage and integration via transformations. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 1029–1032, New York, NY, USA, 2013. ACM.

[4] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 280–284, New York, NY, USA, 2000. ACM.

[5] D. Canali and D. Balzarotti. Behind the scenes of online attacks: an analysis of exploitation behaviors on the web. In *NDSS 2013, 20th Annual Network and Distributed System Security Symposium, February 24-27, 2013, San Diego, CA, United States*, San Diego, UNITED STATES, 02 2013.

[6] T.-H. Chan, M. Li, E. Shi, and W. Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In S. Fischer-HÃijbner and M. Wright, editors, *Privacy Enhancing Technologies*, volume 7384 of *Lecture Notes in Computer Science*, pages 140–159. Springer Berlin Heidelberg, 2012.

[7] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. In *Proceedings of the 37th international colloquium conference on Automata, languages and programming: Part II*, pages 405–417, Heidelberg, 2010. Springer-Verlag.

[8] E. Chlebus and J. Brazier. Nonstationary poisson modeling of web browsing session arrivals. *Inf. Process. Lett.*, 102(5):187–190, May 2007.

[9] R. Cooley, B. Mobasher, and J. Srivastava. Web mining: information and pattern discovery on the world wide web. In *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*, pages 558–567, 1997.

[10] 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, Heidelberg, 2006. Springer-Verlag.

[11] 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*, pages 715–724, New York, 2010. ACM.

[12] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1065–1074, New York, NY, USA, 2008. ACM.

[13] M. Eirinaki and M. Vazirgiannis. Web mining for web personalization. *ACM Trans. Internet Technol.*, 3(1):1–27, Feb. 2003.

[14] L. Fan and L. Xiong. Real-time aggregate monitoring with differential privacy. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2169–2173, New York, 2012. ACM.

[15] L. Fan and L. Xiong. An adaptive approach to real-time aggregate monitoring with differential privacy. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2013.

[16] M. Götz, S. Nath, and J. Gehrke. Maskit: privately releasing user context streams for personalized mobile applications. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 289–300, New York, NY, USA, 2012. ACM.

[17] R. E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[18] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 171–180, New York, NY, USA, 2009. ACM.

[19] R. Kosala and H. Blockeel. Web mining research: a survey. *SIGKDD Explor. Newsl.*, 2(1):1–15, June 2000.

[20] R. Kumar and A. Tomkins. A characterization of online browsing behavior. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 561–570, New York, NY, USA, 2010. ACM.

[21] F. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. volume 53, pages 89–97, New York, 2010. ACM.

[22] S. Papadimitriou, F. Li, G. Kollios, and P. S. Yu. Time series compressibility and privacy. 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 *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 735–746, New York, 2010. ACM.

[24] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explor. Newsl.*, 1(2):12–23, Jan. 2000.

[25] D. Wang, Y. He, E. Rundensteiner, and J. F. Naughton. Utility-maximizing event stream suppression. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 589–600, New York, NY, USA, 2013. ACM.

[26] O. Williams and F. McSherry. Probabilistic inference and differential privacy. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2451–2459. 2010.

[27] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*, pages 32–43, Washington, DC, 2012. IEEE Computer Society.

[28] J. Yan, D. Yuan, X. Xing, and Q. Jia. Kalman filtering parameter optimization techniques based on genetic algorithm. In *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, pages 1717–1720, 2008.

[29] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng. Understanding user behavior in large-scale video-on-demand systems. In *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, EuroSys '06, pages 333–344, New York, NY, USA, 2006. ACM.

APPENDIX

A. PROOF OF THEOREM 1

Below we analyze the posterior error when the Laplace perturbation noise is approximately modeled with a Gaussian noise. Given any web page i and $\hat{x}_k^i$ derived by the Kalman filter based estimation algorithm, i.e. Algorithm 1 and 2, the posterior error variance can be calculated as follows:

$$var(\hat{x}_k^i - x_k^i) = E(\hat{x}_k^i - x_k^i)^2 - E^2(\hat{x}_k^i - x_k^i) \ . \qquad \text{(A.1)}$$

Note that both process noise and measurement noise are white and mutually, serially independent. By definition of $\hat{x}_k$, we get the following:

$$E(\hat{x}_k^i - x_k^i) = 0 \ .$$

Therefore, we will only need to estimate the first term in Equation (A.1). Substituting Line 2 in Algorithm 2 leads to

$$\begin{aligned} E(\hat{x}_k^i - x_k^i)^2 &= E[(1-K_k^i)(\hat{x}_k^{i-} - x_k^i) + K_k^i(z_k^i - x_k^i)]^2 \\ &= E[(1-K_k^i)(\hat{x}_{k-1}^i - x_k^i) + K_k^i\nu_k^i]^2 \\ &= E[(1-K_k^i)(\hat{x}_{k-1}^i - x_{k-1}^i) \\ &\quad + (1-K_k^i)(x_{k-1}^i - x_k^i) + K_k^i\nu_k^i]^2 \\ &= (1-K_k^i)^2 E(\hat{x}_{k-1}^i - x_{k-1}^i)^2 \\ &\quad + (1-K_k^i)^2 Q^i + (K_k^i)^2 \frac{2l_{max}^2}{\alpha^2} \end{aligned} \qquad \text{(A.2)}$$

where ν_k^i represents the Laplace perturbation noise and follows $Lap(0, \frac{l_{max}}{\alpha})$.

Substituting the Kalman gain, i.e. Line 1 in Algorithm 2, into Equation (A.2), we get

$$E(\hat{x}_k^i - x_k^i)^2 = \frac{R^2[E(\hat{x}_{k-1}^i - x_{k-1}^i)^2 + Q^i]}{(P_k^{i-} + R)^2} + \frac{2(P_k^{i-} l_{max})^2}{(P_k^{i-} + R)^2 \alpha^2} \ .$$

Applying the gradient descendant method to minimize the posterior error variance, we obtain the following result for R:

$$R = \frac{l_{max}^2}{\alpha^2} \frac{2P_k^{i-}}{E(\hat{x}_{k-1}^i - x_{k-1}^i)^2 + Q^i} \ .$$