Differentially Private Real-Time Data Publishing over Infinite Trajectory Streams*

Yang CAO†[a], Nonmember and Masatoshi YOSHIKAWA†, Member

SUMMARY Recent emerging mobile and wearable technologies make it easy to collect personal spatiotemporal data such as activity trajectories in daily life. Publishing real-time statistics over trajectory streams produced by crowds of people is expected to be valuable for both academia and business, answering questions such as “How many people are in Kyoto Station now?” However, analyzing these raw data will entail risks of compromising individual privacy. e-Differential Privacy has emerged as a well-known standard for private statistics publishing because of its guarantee of being rigorous and mathematically provable. However, since user trajectories will be generated infinitely, it is difficult to protect every trajectory under e-differential privacy. On the other hand, in real life, not all users require the same level of privacy. To this end, we propose a flexible privacy model of t-trajectory privacy to ensure every desired length of trajectory under protection of e-differential privacy. We also design an algorithmic framework to publish t-trajectory private data in real time. Experiments using four real-life datasets show that our proposed algorithms are effective and efficient.

key words: privacy preserving data publishing, differential privacy, personalized privacy, location privacy, trajectory streams

1. Introduction

In recent years, personal data have been increasingly collected, stored, and analyzed. The information is sensitive, especially location data showing where one has visited. These successive points of interest (POIs), along with timestamps, constitute our moving trajectories in daily life. Releasing real-time statistics of trajectory streams produced by crowds of people is expected to be extremely valuable for data-based analysis and decision making. Consider the scenario illustrated in Fig. 1. A trusted server is collecting many people’s moving trajectories continuously. The raw data are spatiotemporal points, as illustrated in Fig. 1 (a). User u’s $t_u$-trajectory is defined as $t_u$ successive (maybe not adjacent) spatiotemporal data points on the timeline, as illustrated in Fig. 1 (b). For example, $l_1 \rightarrow l_2 \rightarrow l_3 \rightarrow l_4$ and $l_3 \rightarrow l_4 \rightarrow l_2$ are $u$’s two overlapped 3-trajectories. Then statistics at each timestamp such as “How many people are in location $l_t$ at the current time?” can be published such as Fig. 1 (c). These statistical answers are expected to be useful for marketing analysis [2], real-time traffic analysis [5], intelligent navigation systems [6], web browsing behavior mining [7] (users’ trajectories in the cyberspace) and so on. To protect privacy, such sensitive data should be anonymized. However, previous research has pointed out that even anonymized information can be used to identify a particular person. Several well-known privacy disclosure cases have arisen, such as an attack on the Netflix database [8] and medical records of the governor of Massachusetts [9]. Such attacks are called “linkage attacks” [10] by which an adversary uses auxiliary information to obtain sensitive data about a target. For example, if an attacker knows that $u_3$ has stayed at some places at timestamps $t_1, t_2, t_3$, then by the published data of Fig. 1 (c), the trajectory of $u_3$ can be inferred as either $l_1 \rightarrow l_2 \rightarrow l_4$ or $l_1 \rightarrow l_3 \rightarrow l_4$. Especially, people’s moving trajectories are extremely vulnerable. Research [11] shows that only four spatiotemporal points from anonymized mobile datasets are sufficient to identify 95% of individuals uniquely. Therefore, a rigorous privacy model is necessary.

e-Differential Privacy (DP), has emerged as a well-known standard for privacy preserving data publishing (PPDP) because of rigorous theoretical guarantees of robustness under arbitrary linkage attack [12, 13]. $\epsilon$ is a positive parameter called privacy budget which is given in advance to control the privacy level. The larger $\epsilon$, the lower the privacy level. Roughly speaking, under $e$-DP, the modification (e.g., removing or changing the values) of any single user’s data is guaranteed to have a limited effect (bounded by $\epsilon$) on the outcome of data publishing. In this way, the data privacy of any single user is protected from arbitrarily analyzing the published data (by attackers or users). However, DP is criticized by practitioners because of sacrificing too much data utility (e.g., adding too much random noises in order to achieve DP). There are only few real-world applications of differential privacy such as US census† private data.

* Manuel received March 22, 2015.
Manuscript revised August 4, 2015.
Manuscript publicized October 6, 2015.
† The authors are with the Graduate School of Informatics, Kyoto University, Kyoto-shi, 606-8501 Japan.
A preliminary version of this paper appeared in [1].
a) E-mail: soyo@db.soc.i.kyoto-u.ac.jp
DOI: 10.1587/transinf.2015EDP7096

†http://ontheemap.ces.census.gov/
First, it is hard to uniformly protect trajectories individually. Therefore, a fine-grained privacy model for trajectory stream data, it focuses on protecting the data within any single sliding window. Besides the parameter $\epsilon$, it introduces an additional parameter $w$ to control the size (the number of contiguous timestamps) of sliding windows, to guarantee that the trajectories within any single sliding window will be protected under $\epsilon$-DP. Regarding 3-event privacy in Fig. 2 (a), it guarantees that the following trajectories will be protected under $\epsilon$-DP: $u_1$'s 1-trajectory $l_1$ and $l_2$, $u_2$'s 1-trajectories $l_2$ and $l_1$, $u_3$'s 3-trajectory $l_1 \rightarrow l_3 \rightarrow l_4$ (including all of the sub-trajectories such as $l_1 \rightarrow l_3$) and $u_3$'s 2-trajectory $l_4 \rightarrow l_2$.

However, w-event privacy is not sufficient to provide uniform and personalized protection for trajectory streams. First, it is hard to uniformly protect every $\ell_a$-trajectory for a given $\ell_a$ under $\epsilon$-DP by its sliding window methodology. For example, regarding 3-event privacy in Fig. 2 (a), the following trajectories are not guaranteed to be protected under $\epsilon$-DP because they exceed any single 3-length sliding window: $u_1$'s 2-trajectory $l_1 \rightarrow l_2$, $u_2$'s 2-trajectory $l_2 \rightarrow l_1$, $u_3$'s 4-trajectory $l_1 \rightarrow l_3 \rightarrow l_4 \rightarrow l_2$. In general, there is no upper bound in the number of contiguous timestamps a $\ell_a$-trajectory spans\(^{11}\). Second, different individuals may have different expectations for the privacy of their trajectories. For example, if $u_1$ wants his/her every 10-trajectory to be protected, while $u_2$ only wants every 3-trajectory to be protected, it is hard to achieve both of them by w-event privacy. Therefore, a fine-grained privacy model for trajectory streams is necessary.

For those reasons, we propose $\ell$-trajectory privacy, where a vector $\ell$ is defined as $\langle \ell_1, \ldots, \ell_d \rangle$. Let $U$ be the set of all users, and $\ell_u$ is user $u$'s desired length of being protected trajectories, namely privacy preference of user $u$. Our model guarantees that every $\ell_u$-trajectory is protected under $\epsilon$-DP. Fig. 2 (b) illustrates the protected range of all users’ 3-trajectory privacy.

In this paper, we formulate $\ell$-trajectory privacy, and prove that it can be achieved. Then, we design algorithms to achieve $\ell$-trajectory privacy and boost data utility. The algorithm will be composed of sub-algorithms. Each one of them takes collected database at each timestamp as input, and outputs private data in real time. A salient challenge to achieve $\ell$-trajectory privacy is that the upcoming spatiotemporal data points are unknown, hence, a naive method that pre-allocates the local budget will lead to inferior data utility. To this end, we propose an algorithmic framework that adopts dynamic budget allocation and approximate publishing. Then, two algorithms, GA+Adj and GA+MMD are implemented, as two approximation strategies.

The rest of this paper is organized as follows. Section 2 briefly surveys the related works on differential privacy of data streams and private trajectory data publishing. Section 3 gives background knowledge of differential privacy and classical mechanisms to achieve it. Section 4 provides the problem formulation. Section 5 presents an overview of our framework, as well as the technical details of two proposed algorithms. Section 6 describes the data set and presents a set of experimental results. Section 7 concludes the paper and states possible directions for future work.

2. Related Work

2.1 Differential Privacy on Streams

For stream data, user-level privacy (e.g., to protect the whole trajectory stream of a user) and event-level (e.g., to protect any single spatiotemporal data point) privacy have been proposed\([12], [13]\) as two different privacy goals. Previous work mainly focuses on event-level privacy on finite or infinite streams\([14], [16], [17]\), and user-level privacy on finite streams\([5], [7], [15]\). Chan et al.\([16]\) proposed scheme of $p$-sum to construct a full binary tree on the sequential data, where each node contains the sum of the sequential data in its subtree, plus noise with scale logarithmic in the length of the sequence. When the noise is accumulated up to a threshold $p$, it makes a release. Bolot et al. proposed a scheme of decayed privacy\([17]\), which consider past data is not as sensitive as current data. Mir et al.\([15]\) considers counters that may also decrease. Fan and Xiong\([5], [7]\) proposed FAST framework for publishing time-series data in a user-level private way. FAST uses sampling and filtering components to reduce the noise; given a specified number of samples, the filtering component predicts the future data and corrects its prior data by noisy samples. The authors report that their adaptively sampling scheme preserves high
utility at the same privacy level. However, this scheme takes as input the total amount of timestamps \( |T| \), which leads to inapplicability in our infinite scenario.

2.2 Privacy of Trajectory Data

The major work in previous research specifically examined techniques of anonymization [18]–[20] and scenarios of directly publishing trajectory data [21], [22]. Little work connects trajectory data and differential privacy. Chen et al. [22] proposed a data-dependent sanitization mechanism by grouping trajectories to produce a noisy prefix tree. Jiang et al. [23] also examined differentially publishing private trajectories by sampling suitable direction and distance of trajectory data. Ho et al. [24], [25] proposed a differentially private approach to mine interesting geographic location patterns from trajectory data. These works address the scenario of directly publishing user-level trajectories or the results of specific data mining tasks, which are different from publishing sequential statistics for sets of users’ trajectories.

3. Preliminary

3.1 Differential Privacy

Differential privacy was proposed by Dwork et al. [26]. Roughly speaking, differential privacy attempts to limit the impact of an individual’s record relative to the answer of a query on the database. In other words, whether the sensitive information is present or not in the database, the answers of a query should have little difference. In this way, an adversary cannot associate any piece of information with a specific individual from mining the answers of queries.

Definition 1 (Neighboring Database). If \( D \) is a database, and \( D' \) is a copy of \( D \) but differing in one record, then we say \( D \) and \( D' \) are two neighboring databases.

Definition 2 (\( \epsilon \)-Differential Privacy, \( \epsilon \)-DP). \( \Lambda \) is a randomized algorithm. \( \Lambda \) represents all possible outputs of \( \Lambda \). Algorithm \( \Lambda \) satisfies \( \epsilon \)-differential privacy, if for any \( r \in R \) and any two neighboring databases \( D, D' \), we have the following.

\[
Pr[\Lambda(D) = r] \leq \exp(\epsilon) \cdot Pr[\Lambda(D') = r]
\]

In Inequality (1), \( \epsilon \) is a privacy budget given in advance. It is used to control the privacy level. We note that lower \( \epsilon \) signifies higher privacy and more randomness, and vice versa. Global sensitivity (or sensitivity for short) \( GS(Q) \) of query \( Q \) is the maximum \( L_1 \) distance between the query results for any two neighboring databases.

To avoid misunderstandings, we note that there are different ways to define differential privacy, based on the definition of neighboring databases that differ by adding/removing an individual’s record (unbounded) [26], or modifying the content of an individual’s record but keep the size of database (i.e., number of records) (bounded) [27].

Unless we explicitly state otherwise, we use bounded differential privacy in this paper. Further discussion of these differences can be found in [28].

3.2 Mechanisms of Differential Privacy

Two widely used methods to achieve differential privacy is Laplace mechanism [27] which adds random noise to actual data to prevent the disclosure of sensitive information, and Exponential mechanism [29] which randomly returns the desired item by some calibrated probability.

Theorem 1 (Laplace Mechanism [27]). \( \epsilon \)-DP can be achieved by adding independent Laplace random noise \( x \) to the answer of query \( Q \). In the following, \( n \) and \( D \) denote the noisy data and a given database, respectively.

\[
n = Q(D) + x
\]

\[
x \sim \Pr(x|\lambda) = \frac{1}{2\lambda} \cdot \exp(-|x|/\lambda)
\]

where \( \lambda \) is a scale parameter of Laplace distribution, which equals to \( GS(Q)/\epsilon \).

Another way to obtain differential privacy is through the Exponential mechanism [29]. Given a quality function \( q \) that scores results of a specified query, where higher scores are better. Then returning results in the probability of exponentially proportional to the corresponding score. In other words, the result with a higher score is exponentially more likely to be chosen. This will ensure differential privacy.

Theorem 2 (Exponential Mechanism [29]). Let \( D' \) denotes all possible databases in the universe, and \( D, D' \in D' \) are two instances of neighboring databases. Let \( q : (D' \times R) \rightarrow \mathbb{R} \) be a quality function that, \( R \) as a certain query’s all possible sensitive outcome, given a database \( D \in D' \), assigns a score to each outcome \( r \in R \). Then we define global sensitivity \( GS(q) \) as max, \( \sum q(D, r) - q(D', r) \). \( D, D' \in D' \). Exponential mechanism \( M \) is obtained by returning the sensitive answer \( r \) with probability proportional to \( \exp\left(\frac{GS(D)}{\lambda}\right)\).

The following sequential composition theory reveals the method of integrating multiple algorithms to achieve differential privacy.

Theorem 3 (Sequential Composition [30]). Presuming that \( \Lambda_i \) satisfies \( \epsilon_i \)-DP over database \( D \), then an integrated algorithm \( \Lambda \) that consists of multiple \( \Lambda_i \) over the same database \( D \) satisfies \( \sum \epsilon_i \)-DP.

4. Problem Definition and Privacy Model

4.1 Data Model

We consider scenario of a trusted server that collects database \( D_i \) of users’ spatiotemporal data points continuously at each timestamp \( t \in [1, T] \). Let \( t \) denote the current timestamp. Each data point \((uID, time, loc)\) is a row in
Table 1  Summary of notations.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \epsilon )</td>
<td>Total privacy budget</td>
</tr>
<tr>
<td>( \epsilon_i )</td>
<td>Privacy budget variable at timestamp ( i )</td>
</tr>
<tr>
<td>( t_u )</td>
<td>The length of user ( u )'s being protected trajectories</td>
</tr>
<tr>
<td>( i, k )</td>
<td>Variables of timestamps</td>
</tr>
<tr>
<td>( D_i, D'_i )</td>
<td>Collected database at timestamp ( i ) and the corresponding neighboring database</td>
</tr>
<tr>
<td>( T )</td>
<td>Set of all timestamps</td>
</tr>
<tr>
<td>( U, u_S )</td>
<td>Set of users in databases of the whole timeline and ( D_i ) respectively</td>
</tr>
<tr>
<td>( \ell_{\alpha k} )</td>
<td>A ( \ell_u )-trajectory of user ( u ) ends with timestamp ( k )</td>
</tr>
<tr>
<td>( \tau_{u k} )</td>
<td>Set of timestamps in ( \ell_{u k} )</td>
</tr>
<tr>
<td>( \text{locs} )</td>
<td>Vector of POIs</td>
</tr>
<tr>
<td>( j )</td>
<td>Index variable on POIs</td>
</tr>
<tr>
<td>( S_t, S'_t )</td>
<td>Trajectory stream prefixes and the corresponding neighboring stream prefixes</td>
</tr>
<tr>
<td>( Q )</td>
<td>Count query</td>
</tr>
<tr>
<td>( R, r_i )</td>
<td>Real statistics of all timestamps and timestamp ( i ) respectively</td>
</tr>
<tr>
<td>( N, n_i )</td>
<td>Noisy statistics of all timestamps and timestamp ( i ) respectively</td>
</tr>
<tr>
<td>( A_i )</td>
<td>Sub-algorithm at timestamp ( i )</td>
</tr>
<tr>
<td>( A )</td>
<td>An integrated algorithm consisting of ( A_i, i \in [1, t] )</td>
</tr>
</tbody>
</table>

\( D_i \) (Fig. 1 (a)). Assuming that \( \text{locs} \) is a set of all locations in which we were interested (POIs), then the server wishes to publish a vector \( r_i \) at each timestamp \( i \) as the counts of \( loc \in \text{locs} \) appeared in \( D_i \) (Fig. 1 (c)), i.e., the answer of count query \( Q : D_i \to \mathbb{R}^{\text{locs}} \). We assume that a user only appears at most one location at each timestamp, in other words, \((uID, \text{time})\) is the primary key in the whole database.

Infinite \( D_i \) constitutes trajectory streams. Prefixes of the infinite trajectory streams are all the database up to the current timestamp \( t \), denoted by \( S_t = \{D_1, \cdots, D_t\} \).

**Definition 3** (\( \ell_u \)-trajectory). A sequence of successive spatiotemporal data points produced by user \( u \) is a \( \ell_u \)-trajectory if the number of data points is equal to a natural number \( \ell_u \). A \( \ell_u \)-trajectory ends at timestamp \( k \), as denoted by \( \ell_{u k} = \{(u, i, \text{loc}), \cdots, (u, k, \text{loc'})\} \) where \( \text{loc'}, \text{loc'} \in \text{locs} \). We say that the set of timestamps appeared in these data points (i.e., \( \{i, \cdots, k\} \)), denoted by \( \tau_{u k} \), is dominated by \( \ell_{u k} \).

For convenience, we summarize the notations and symbols used in this paper in Table 1.

**4.2 Privacy Risk**

In our scenario, publishing the real statistics will make users’ privacy at risk. As described above, even releasing anonymized information is maybe not safe because of the adversary’s unforeseen background knowledge (linkage attack [10]). For instance, in the example of Fig. 1, assuming that an inferred adversary knows that user 3 has stayed at some places at timestamps \( t_1, t_2, t_3 \), then by the published data Fig. 1 (c), the trajectory of user 3 can be inferred as either \( l_1 \to l_2 \to l_4 \) or \( l_1 \to l_3 \to l_4 \).

To avoid such risks, we will propose a privacy model based on differential privacy that does not make any assumption on the background knowledge of the attacker. We will give a formal definition of our privacy goal under \( \epsilon \)-DP in Sect. 4.3.

**4.3 Proposed Privacy Model**

To define our model under \( \epsilon \)-DP, we must first clarify the definition of neighboring databases under our setting.

**Definition 4** (Neighboring databases at timestamp \( i \)). If a database \( D_i \) is collected at timestamp \( i \in [1, l] \), and \( D'_i \) is a copy of \( D_i \) but differing in one \( \text{loc} \) in a single record of an arbitrary user \( u \), then we say that \( D_i, D'_i \) is a pair of neighboring databases with respect to \( u \).

Since we assume that a user only appears at most one \( \text{loc} \) at each timestamp, and modifying single \( \text{loc} \) means either change \( \text{loc} \) to another \( \text{loc'} \in \text{locs} \) or \( \text{loc'} \notin \text{locs} \), according to the definition of differential privacy, the sensitivity of \( Q \) is 2.

We define \( \ell_u \)-trajectory neighboring stream prefixes as follows. Intuitively, they are trajectory streams differing in a \( \ell_u \)-trajectory of user \( u \).

**Definition 5** (\( \ell_u \)-trajectory neighboring stream prefixes). Let \( S'_t \) be a copy of trajectory stream prefixes \( S_t \) but differing in \( \ell_u \)-trajectory neighboring databases \( D'_t \). \( S_t \) and \( S'_t \) are \( \ell_u \)-trajectory stream prefixes neighboring each other if one is obtained from another by modifying single or multiple \( \text{loc} \) in any one \( \ell_u \)-trajectory \( \ell_{u k} \) (recall that a \( \ell_u \)-trajectory is a set of \( \ell_u \) spatiotemporal data points). We say that \( S_t \) and \( S'_t \) are neighboring with respect to \( \ell_{u k} \).

Then we define \( \ell \)-trajectory privacy where \( \ell \) is a vector consisting of all user’s privacy preferences as follows. It is attempted to cover the impact of any \( \ell_u \)-trajectory relative to the answer of a sensitive query.

**Definition 6** (\( \ell \)-trajectory \( \epsilon \)-differential privacy). Let \( \ell = \langle \ell_1, \ldots, \ell_l \rangle \). Let \( \Lambda \) be an algorithm that takes prefixes of trajectory streams \( S_l = \{D_1, \ldots, D_l\} \) as inputs. Let \( N_l = \{n_1, \ldots, n_l\} \) be a possible perturbed output stream of \( \Lambda \). If for any \( \ell_u \)-trajectory neighboring \( S_t \) and \( S'_t \), the following holds,

\[
\Pr[\Lambda(S_t) = N_l] \leq e^{\epsilon} \cdot \Pr[\Lambda(S'_t) = N_l]
\]

then we say that \( \Lambda \) satisfies \( \ell \)-trajectory \( \epsilon \)-differential privacy (simply, \( \ell \)-trajectory privacy).

**4.4 Methodology to Achieve \( \ell \)-Trajectory Privacy**

The next question is how to achieve \( \ell \)-trajectory privacy. We prove Theorem 4. Roughly speaking, to satisfy \( \ell \)-trajectory privacy, the sum of privacy budgets allocated to sub-algorithms at the timestamps of any single \( \ell_u \)-trajectory must be less than or equal to the total privacy budget \( \epsilon \).

**Theorem 4.** Let \( \Lambda \) be an integrated algorithm which takes stream prefixes \( S_l = \{D_1, \ldots, D_l\} \) as input, and \( N_l = \)
{n_1, \ldots, n_t} as output. \Lambda consists of a series of sub-algorithms \{A_1, \ldots, A_J\}, where each A_i takes D_t as input, and outputs noisy data n_i with independent randomness.

Presume A_i ensures \(\epsilon_i\)-DP, and let \(\tau_{u,k}\) be the set of timestamps dominated by an arbitrary trajectory \(\ell_{u,k}\), then \(\Lambda\) satisfies \(\epsilon\)-DP if

\[
\sum_{t \in \tau_{u,k}} e_i \leq \epsilon \text{ holds.} \tag{5}
\]

**Proof.** Presume the integrated algorithm \(\Lambda\) takes inputs as \(S_i\) and \(S'_i\) independently, which are two \(\ell_u\)-neighboring stream prefixes with respect to \(\ell_{u,k}\), and the outputs are \(N_i, N'_i\) respectively. Since each \(A_i\) has independent randomness, the followings hold.

\[
\Pr[A(S_i) = N_i] = \prod_{t=1}^{\tau} \Pr[A_i(D_t) = n_i] \tag{6}
\]
\[
\Pr[A(S'_i) = N_i] = \prod_{t=1}^{\tau} \Pr[A_i(D'_t) = n_i] \tag{7}
\]

Dividing both side of Eq. (6) by (7) respectively, we have the following.

\[
\frac{\Pr[A(S_i) = N_i]}{\Pr[A(S'_i) = N_i]} = \prod_{t=1}^{\tau} \frac{\Pr[A_i(D_t) = n_i]}{\Pr[A_i(D'_t) = n_i]} \tag{8}
\]

Since each \(A_i\) are \(\epsilon_i\)-DP, we derive the following.

\[
\prod_{t=1}^{\tau} \Pr[A_i(D_t) = n_i] \leq e^{\Sigma_{t=1}^{\tau} \epsilon_i} \tag{9}
\]

According to Definition 5 of \(\ell_u\)-trajectory neighboring stream prefixes, the set of timestamps in Eq. (8) is \(\tau_{u,k}\). Therefore, by Eq. (8) and Inequality (9) we obtain the following.

\[
\frac{\Pr[A(S_i) = N_i]}{\Pr[A(S'_i) = N_i]} \leq e^{\Sigma_{t=1}^{\tau} \epsilon_i}, \quad i \in \tau_{u,k} \tag{10}
\]

Finally, by Inequality (5), for any \(\ell_u\)-trajectory neighboring stream prefixes \(S_i, S'_i\), the following holds.

\[
\frac{\Pr[A(S_i) = N_i]}{\Pr[A(S'_i) = N_i]} \leq e^{\Sigma_{t=1}^{\tau} \epsilon_i} \leq e^\epsilon, \quad i \in \tau_{u,k} \tag{11}
\]

By Definition (6), \(\Lambda\) is \(\ell\)-trajectory private. \(\Box\)

4.5 Utility Metrics

We will add random noises to the real data to achieve \(\ell\)-trajectory privacy. Therefore, the data utility will be measured as the magnitude of random noises. We adopt Mean of Absolute Error (MAE) as shown below. Here, \(r_i\) and \(n_i\) as vectors of length \(|\text{loc}\rangle\) are real and noisy data at timestamp \(i\), respectively. \(R\) and \(N\) are real and noisy data of all timestamps, respectively. The error magnitude is measured as Eq. (12) and (13).

\[
\text{MAE}(r_i, n_i) = \frac{1}{|\text{loc}\rangle} \sum_{j=1}^{|\text{loc}\rangle} |r_i[j] - n_i[j]| \tag{12}
\]
\[
\text{MAE}(R, N) = \frac{1}{|T|} \sum_{i=1}^{|T|} \text{MAE}(r_i, n_i) \tag{13}
\]

MAE will be considered as a utility goal in the following algorithm design.

5. Algorithms

5.1 First-Cut Solution

According to the analysis in Sect. 4.4, the essential problem to achieve \(\ell\)-trajectory privacy is how to allocate appropriate privacy budgets at each timestamp. The straightforward idea is to uniformly allocate the privacy budget of \(\epsilon/\ell_{\max}\) at each timestamp, where \(\ell_{\max}\) is the maximum value in \(\ell\), as shown in Algorithm 1. The integrated algorithm Uniform includes Uniform_i of each timestamp i \(\in [1, t]\).

**Theorem 5.** Uniform satisfies \(\ell\)-trajectory privacy.

**Proof.** As the global sensitivity of \(Q^c\) is 2, by Laplace mechanism, each Uniform_i, \(i \in [1, t]\) satisfies \(\epsilon/\ell_{\max}\)-DP. In addition, because of \(\epsilon/\ell_{\max} \leq \epsilon/\ell_u\), the sum of privacy budgets \(\epsilon/\ell_{\max}\) of Uniform at the timestamps of any single \(\ell_u\)-trajectory will be at most \(\epsilon\). By Theorem 4, Uniform satisfies \(\ell\)-trajectory privacy. \(\Box\)

5.2 Proposed Framework

Uniform leaves us no space for optimization. To optimize privacy budget allocation, an idea is to dynamically allocate budgets at each timestamp according to the data points distribution. Besides ensuring \(\ell\)-trajectory privacy, we also wish to spend more budget when it is “worth”, and vice versa. How to measure whether it is a worthwhile investment or not? This question leads us to develop approximation strategy cooperating with dynamic budget allocation component to raise data utility.

We propose the algorithmic framework shown in Fig. 3 and Algorithm 2. It is composed of three components. Dynamic budget allocation component allocates budgets at each timestamp and ensures Inequality (5) (to satisfy \(\ell\)-trajectory privacy). Private approximation component makes a decision of whether it is benefit to approximately publish or not, under a certain utility goal (such as MAE) and an approximation strategy (such as republishing). 

**Algorithm 1: Uniform**

**Input:** Database \(D_t\); Vector of trajectory length \(\ell\).
**Privacy budget \(\epsilon\)**
**Output:** noisy data \(n_t\)

1. Calculate the real statistics \(r_t \leftarrow Q(D_t)\)
2. Add scaled Laplace noises \(n_t \leftarrow r_t + \langle\text{Lap}(2\ell_{\max}/\epsilon)|\text{loc}\rangle\)
3. return \(n_t\)
privacy budgets, because they need to access the real statistics. The latter two components need to be assigned (MMD) noisy data based on utility metric MAE, adjacent, and investigate two different privacy mechanisms.

![Algorithm 2: l-Trajectory Private Data Publishing](image)

**Algorithm 2: l-Trajectory Private Data Publishing**

**Input:** Real statistics $r_i$; Users sets $u_i, i \in \{1, t\}$; Vector of privacy preferences $\ell$; Total privacy budget $\epsilon$

**Output:** Noisy statistics $n_i$, Allocated budget $\epsilon_i$

1. $\epsilon_i \leftarrow$ Dynamic Budget Allocation
2. appx $\leftarrow$ Private Approximation
3. $n_i \leftarrow$ Private Publishing

The dynamic budget allocation component releases private (noisy) data by a certain differential privacy mechanism (such as Laplace mechanism). The latter two components need to be assigned privacy budgets, because they need to access the real statistics $r_i$ to make a better decision. Note that the set of users who appear in $D_t$, denoted by $u_{st}$, is considered as nonsensitive data. The reason is that under the Definition 4 of neighboring databases, $u_{st}$ will not be changed between any two neighboring databases.

**Proposition 1.** Given $l$ and a total privacy budget $\epsilon$ at timestamp $t$, (i) the Dynamic Budget Allocation component (row 1) allocating privacy budget $\epsilon_i$ at timestamp $t$ ($i \in \{1, t\}$), ensures Inequality (5) of Theorem 4; and (ii) the Private Approximation component (row 2) and the Private Publishing component (row 3), achieve overall $\ell_t$-DP. Then Algorithm 2 satisfies $l$-trajectory privacy.

**Proof.** Since the dynamic budget allocation component only takes private data as input, the three components achieve $l$-DP overall. According to Theorem 4, the conclusion holds.

In the following sections, we will design a greedy algorithm (GA) in the dynamic budget allocation component, and investigate two different approximation strategies which are republishing the adjacent (Adj) and globally most similar (MMD) noisy data based on utility metric MAE, then adopt Laplace mechanism in private publishing component. We note that the framework is orthogonal to its implementation. For example, we can define different utility goal (other than MAE), or adopt different approximation scheme and differential privacy mechanisms.

### 5.3 Dynamic Budget Allocation

The first question about privacy budget allocation is that how to decide the budget proportions of the left two components in Fig. 3. We will allocate a fixed budget $\epsilon_{1,t}$ to the private approximation, and dynamic budget $\epsilon_{2,t}$ to the private publishing component. The reason is that, as the private approximation component will decide whether it is necessary to republish or not, it significantly affects the data utility. Therefore, we will allocate a fixed budget (a fixed level of randomness) in order to make it more stable.

Algorithm 3 describes how to dynamically allocate budget to the private publishing component in an exponentially decay fashion at the current timestamp $t$. First, it calculates the maximum available budget of the current timestamp under constraint of Inequality (5) by checking every user $l_{u,t}$ for every $u$. Then, as the algorithm may also need budgets at the future timestamps, it assigns half of the available amount to the current one.

**Algorithm 3: Dynamic Budget Allocation (GA$_t$)**

**Input:** Users set $u_i, i \in \{1, t\}$; Previously allocated privacy budget $\epsilon_i, i \in \{1, t - 1\}$; Vector of privacy preferences $\ell$; Total privacy budget $\epsilon$

**Output:** privacy budget $\epsilon_{1,t}, \epsilon_{2,t}$

1. Allocate a fixed budget $\epsilon_{1,t} \leftarrow \epsilon/(2 \ast \ell_{\text{max}})$
2. Calculate spent budget $\epsilon_i' \leftarrow \max_{u \in u_i} \{\sum_{t \in u_i} \epsilon_i, i \in u_i\}$
3. Calculate remainder budget $\epsilon_i'' \leftarrow \epsilon - \epsilon_i'$
   // exponentially decreasing
4. Allocate dynamic budget $\epsilon_{2,t} \leftarrow \epsilon_i''/2$
5. return $\epsilon_{1,t}, \epsilon_{2,t}$

Lemma 1. Algorithm 3 (GA$_t$) ensures that the sum of privacy budgets on any one $\ell$-trajectory is at most $\epsilon$.

**Proof.** Two parts of budgets are allocated on each $\ell$-trajectory: the fixed budgets $\epsilon_{1,t}$ and the dynamic budgets $\epsilon_{2,t}$. We will prove that both parts are at most $\epsilon/2$. In Line 1, it is uniformly allocated $\epsilon/(2 \ast \ell_{\text{max}})$ at each timestamp. In other words, the sum of the fixed budgets on any $\ell$-trajectory is at most $\epsilon/2$. In Line 2, since all of the previous budgets are known, it calculates the sums of previously spent budgets on each $l_{u,t}$, $u \in U$, and the maximum one is denoted by $\epsilon_i'$. Then the maximum available budget $\epsilon_i''$ is calculated in Line 3. Finally, in Line 4, it saves half of the total available budget for the future data and assigns another half to $\epsilon_{2,t}$. Hence, the worst case in GA is that a user produces data points at every timestamps. Then, the sum of dynamically allocated budgets on this user’s $l$-trajectory will be $\epsilon/2^2 + \epsilon/2^3 + \cdots + \epsilon/2^{t+1}$, which is less than $\epsilon/2$. Therefore, the sum of all allocated budgets on any single $\ell$-trajectory will be less than $\epsilon$. 

□
counts appear around 8 a.m. or 6 p.m. everyday. The observation scheme is to search the most similar noisy data on the timeline, as shown in Algorithm 5 (MMD). The principle of temporal utility.

The first one is to simply employ the adjacent noisy data, as shown in Algorithm 4 (Adj). As the principle of temporal utility.

Approximation strategies have been investigated in earlier research, such as histogram publishing [31], [32], and statistics on data stream publishing [5], [14]. Instead of directly adding noise to real data, they function by transformation of original data or a query structure to achieve better overall utility.

In our case, we will (i) choose an appropriate noisy data which was previously published (Fig. 4), and (ii) republish it if it is “close to” the real statistics which we want to publish. The distance between the real and noisy data will be measured by MAE of Eq. (13).

We adopt two schemes in the above mentioned step (i). The first one is to simply employ the adjacent noisy data, as shown in Algorithm 4 (Adj). As the principle of temporal locality, the last published data may appear again. The second scheme is to search the most similar noisy data on the timeline, as shown in Algorithm 5 (MMD). The observation is that the periodically repeating pattern\(^1\) is an intrinsic characteristic in real-life data. Now we will discuss the step (ii). This idea is the same in Lines 1 to 4 of Algorithm 4 and Lines 4 to 8 of Algorithm 5. Let us take the former as an example. In Line 1, it attempts to make a judgment on whether a republishing is benefit or not. Obviously, in this inequality, the left part of MAE indicates the error of republishing \(n_{t-1}\) (the similarity between \(r_t\) and \(n_{t-1}\)). We now examine the left part. When adding Laplace noise to real data, the error is only introduced by Laplace noises. It is easy to know that MAE between real and Laplace noisy data is numerically equal to the scale parameter of the Laplace noise, that is \(2/|\text{locs}| \ast \varepsilon_{t,2}\). Hence, the right part of this inequality indicates Laplace error. Since the left part of MAE needs to access real data \(r_t\), it is necessary to add randomness to perturb the result of this judgment. Then, in the following parts of Algorithm 4, the index of noisy data with relatively lower error will be returned. Therefore, it trades off lower error between Laplace and approximation error.

\(^1\)E.g., for people flow trajectory data, in general, the peak-hour counts appear around 8 a.m. or 6 p.m. everyday.

\begin{algorithm}
\caption{Approximation Strategy (Adj)}
\begin{algorithmic}
\State \textbf{Input:} Real statistics \(r_t\); Privacy budget \(\varepsilon_{t,1}\)
\State \textbf{Output:} timestamp of approximate data
\If{\(\text{MAE}(r_t, n_{t-1}) + \text{Lap}(2/|\text{locs}| \ast \varepsilon_{t,1}) \leq 2/|\text{locs}| \ast \varepsilon_{t,2}\)}
\State \(\text{appx} \leftarrow t - 1\)
\Else
\State \(\text{appx} \leftarrow t\)
\EndIf
\State \textbf{return} \text{appx}
\end{algorithmic}
\end{algorithm}

5.4 Approximation Strategies

Approximation strategies have been investigated in earlier research, such as histogram publishing [31], [32], and statistics on data stream publishing [5], [14]. Instead of directly adding noise to real data, they function by transformation of original data or a query structure to achieve better overall utility.

In our case, we will (i) choose an appropriate noisy data which was previously published (Fig. 4), and (ii) republish it if it is “close to” the real statistics which we want to publish. The distance between the real and noisy data will be measured by MAE of Eq. (13).

We adopt two schemes in the above mentioned step (i). The first one is to simply employ the adjacent noisy data, as shown in Algorithm 4 (Adj). As the principle of temporal locality, the last published data may appear again. The second scheme is to search the most similar noisy data on the timeline, as shown in Algorithm 5 (MMD). The observation is that the periodically repeating pattern\(^1\) is an intrinsic characteristic in real-life data. Now we will discuss the step (ii). This idea is the same in Lines 1 to 4 of Algorithm 4 and Lines 4 to 8 of Algorithm 5. Let us take the former as an example. In Line 1, it attempts to make a judgment on whether a republishing is benefit or not. Obviously, in this inequality, the left part of MAE indicates the error of republishing \(n_{t-1}\) (the similarity between \(r_t\) and \(n_{t-1}\)). We now examine the left part. When adding Laplace noise to real data, the error is only introduced by Laplace noises. It is easy to know that MAE between real and Laplace noisy data is numerically equal to the scale parameter of the Laplace noise, that is \(2/|\text{locs}| \ast \varepsilon_{t,2}\). Hence, the right part of this inequality indicates Laplace error. Since the left part of MAE needs to access real data \(r_t\), it is necessary to add randomness to perturb the result of this judgment. Then, in the following parts of Algorithm 4, the index of noisy data with relatively lower error will be returned. Therefore, it trades off lower error between Laplace and approximation error.

\begin{algorithm}
\caption{Approximation Strategy (MMD)}
\begin{algorithmic}
\State \textbf{Input:} Real statistics \(r_t\); Privacy budget \(\varepsilon_{t,1}\)
\State \textbf{Output:} timestamp of approximate data
\State \(\varepsilon_t = \varepsilon_{t,1}/2\)
\State \textbf{Calculate} \(\text{score}_i\), \text{of quality function}
\State \(q = \text{MD}(r_t, n_i), i \in [1, t - 1]\)
\State \(n_{MMD} \leftarrow n_i \propto \exp(\varepsilon_t \times \text{score}_i), i \in [1, t - 1]\)
\If{\(\text{MAE}(r_t, n_{MMD}) + \text{Lap}(2/|\text{locs}| \ast \varepsilon_{t,1}) \leq 2/|\text{locs}| \ast \varepsilon_{t,2}\)}
\State \(\text{appx} \leftarrow \text{MMD}\)
\Else
\State \(\text{appx} \leftarrow t\)
\EndIf
\State \textbf{return} \text{appx}
\end{algorithmic}
\end{algorithm}

\begin{lemma}
\label{lemma:Adj}
Algorithm 4 (Adj) satisfies \(\varepsilon_{t,1}\)-DP.
\end{lemma}

\begin{proof}
Since the sensitive data \(r_t\) is accessed in Line 1 by MAE query which defined in Eq. (13), it is necessary to add randomness to ensure differential privacy. By Definition 4 of neighboring database \(D_t, D'_t\), the maximum difference (i.e., the sensitivity) of MAE between \(D, D'_t\) is \(2/|\text{locs}|\). Therefore, according to Laplace mechanism, by adding Laplace noises with scale parameter \(2/|\text{locs}| \ast \varepsilon_{t,1}\), it satisfies \(\varepsilon_{t,1}\)-DP. From Lines 2 to 5, Algorithm 4 does not access any sensitive data, hence the conclusion holds.
\end{proof}

On the other hand, under the utility goal of MAE, we observe that the “most similar” noisy data is equal to the one which holds the minimum Manhattan Distance (Eq. (15)) to the current real data, denoted by \(n_{MMD}\). That is because Eqs. (12) and (14) are only different in a factor of \(1/|\text{locs}|\).

\begin{equation}
\text{MD}(r_t, n_i) = \sum_{j=1}^{|\text{locs}|} |r_t[j] - n_i[j]| \tag{14}
\end{equation}

\begin{equation}
\text{MMD}(r_t, n_i) = \min. \{\text{MD}(r_t, n_i), i \in [1, t]\} \tag{15}
\end{equation}

A challenge arises when we attempt to retrieve \(n_{MMD}\) because MMD query accesses \(t - 1\) times real data, which means if we adopt Laplace mechanism it will lead to an unacceptable scaled noise. By adopting Exponential mechanism (Theorem 2) and designing an appropriate quality function, we can reduce this noise within acceptable limits. Algorithm 5 implements this idea and uses the negative value of MD query as quality function of Exponential mechanism. In Line 1, it spent half of privacy budget of \(\varepsilon_{t,1}\) to approximately retrieving \(n_{MMD}\). In Lines 2 and 3, it calculates the quality score and randomly returns the \(n_i\) according the probability of proportional to \(\exp(\varepsilon_t \times \text{score}_i)/4\).

\begin{lemma}
\label{lemma:MMD}
Algorithm 5 (MMD) satisfies \(\varepsilon_{t,1}\)-DP.
\end{lemma}
Algorithm 6: Private Publishing

Input: Real statistics \( r_t \); Privacy budget \( \varepsilon_{t,2} \); Timestamp of approximate data \( apps \)
Output: Noisy data \( n_t \)
1 if \( apps = t \) then
   // add calibrated Laplace noises
   \( n_t \leftarrow r_t + (Lap(2/\varepsilon_{t,2}))_{appx} \), \( \varepsilon_t \leftarrow \varepsilon_{t,1} + \varepsilon_{t,2} \)
3 else
   // republish the approximate noisy data
4 \( n_t \leftarrow n_{appx}, \varepsilon_t \leftarrow \varepsilon_{t,1} \)
5 Return \( n_t, \varepsilon_t \)

Proof. According to the above analysis, Exponential mechanism ensures that the result of Line 3 is \( \varepsilon^m_{t,1} \)-DP, while the procedure of Lines 4 to 7 is \( \varepsilon^o_{t,1} \)-differential private (the proof is similar to Lemma 2, hence omitted here). By the sequential composition of Theorem 3, Algorithm 5 satisfies \( (\varepsilon^s_{t,1} + \varepsilon^o_{t,1}) \)-DP, that is \( \varepsilon_{t,1} \)-DP.

5.5 Private Publishing

We employ Laplace mechanism in the privacy publishing component as shown in Algorithm 6. It takes the real statistics and timestamp of approximate data as inputs, and outputs private statistics. As input, \( apps \) is the timestamp of the approximate noisy data which will be republished. If \( apps \) is equal to the current timestamp, then the private data will be published by adding Laplace noises. Otherwise, the noisy data at timestamp \( apps \) will be republished.

Lemma 4. Algorithm 6 satisfies \( \varepsilon_{t,2} \)-DP.

Proof. According Laplace mechanism [27], adding Laplace noise with scale parameter \( 2/\varepsilon_{t,2} \) ensures \( \varepsilon_{t,2} \)-DP under the sensitivity of 2. □

Theorem 6. GA+Adj satisfies \( \ell \)-trajectory privacy.

Proof. In Proposition 1, the condition (i) is true according to Lemma 1, and the condition (ii) is true because of Lemmas 1, 3 and 4. Hence, the conclusion follows. □

Theorem 7. GA+MMD satisfies \( \ell \)-trajectory privacy.

Proof. This theorem can be proved in the same method as shown before. The conclusion follows from Lemmas 1, 3 and 4. □

6. Experiments

In this section, we will evaluate the privacy level and data utility of the proposed method on four real-life trajectory datasets. The real-life datasets include PeopleFlow (PFlow), Geolife, T-Drive, WorldCup98 (WC98), summarized in Table 2. All of them are available online. The first three datasets are people moving trajectory data by diverse manner, such as walking, driving, by train, or by plane. The forth one is users’ click streams (i.e., from one webpage to another webpage), and it can be seen as people’s trajectory in the cyberspace (i.e., webpages are equivalent to locations). Since our basic setting is that each user is located at most one location at a single timestamp, we randomly remove data points who violate this assumption and sample data points according to intervals shown in Table 2.

As we introduced in Sect. 2, FAST is a sampling and filtering framework for privacy preserving time series data publishing. The authors design two kinds of algorithms based on FAST, that is FAST with adaptively sampling and with fixed sample rate repetitively. Since the former one needs to know amount of total timestamps \( T \) in advance that leads to inapplicability of infinite trajectory streams. Therefore, we use FAST with fixed sampling rate as our utility competitor, denoted by FASTfixed. We configure it according to the original paper [5].

\( \ell \)-Trajectory privacy is featured as a personalized privacy model, which users can specify the length of the protected trajectories. However, since our competitors such as w-event privacy and FASTfixed inherently cannot provide protection with customized \( (\ell_1, \ldots, \ell_{|U|}) \), we will set all \( \ell_u \) as the same value in the following experiments. In addition, since the outputs of the algorithms are including randomness, each algorithm runs 50 times then outputs the average results.

6.1 Privacy Evaluation

In this section, we will quantitatively evaluate the privacy risk of w-event privacy for achieving our privacy goal. As explained in Sect. 1, w-event privacy [14] cannot provide personalized and uniform protection. The former is true because essentially w-event privacy is an event-level, not a user-level privacy model. The latter is because there is no upper bound in the number of contiguous timestamps a \( \ell_u \)-trajectory spans, which has been explained in Sect. 1. However, there is still a little chance for w-event privacy to achieve our privacy goal (i.e., satisfy Theorem 4). Therefore, we quantitatively measure the risk of w-event privacy for achieving our privacy goal. The metric is the percentage of risky \( \ell_u \)-trajectories whose privacy budgets are greater than \( \varepsilon \) (violation of Theorem 4) among all users’ \( \ell_u \)-

---

Table 2 Real-life datasets.

<table>
<thead>
<tr>
<th>Data Points</th>
<th>Time Interval</th>
<th>Max len. traj.</th>
<th>Median len. traj.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFlow</td>
<td>Geolife</td>
<td>T-Drive</td>
<td>WC98</td>
</tr>
<tr>
<td>102,468</td>
<td>5min</td>
<td>9</td>
<td>165</td>
</tr>
<tr>
<td>14,440</td>
<td>1min</td>
<td>7</td>
<td>128</td>
</tr>
<tr>
<td>2,089</td>
<td>10min</td>
<td>21</td>
<td>208</td>
</tr>
<tr>
<td>12,390</td>
<td>18</td>
<td>56</td>
<td>4,016</td>
</tr>
<tr>
<td>3,089,598</td>
<td>6d days</td>
<td>21</td>
<td>2,698</td>
</tr>
<tr>
<td>2,089</td>
<td>39</td>
<td>39</td>
<td>21</td>
</tr>
<tr>
<td>2085</td>
<td>1h</td>
<td>91</td>
<td>10</td>
</tr>
<tr>
<td>2085</td>
<td>&gt;6 days</td>
<td>&gt;87 days</td>
<td>&gt;87 days</td>
</tr>
</tbody>
</table>

---

\( ^{1} \)http://pflow.csis.u-tokyo.ac.jp/
\( ^{11} \)http://research.microsoft.com/projects/geolife
\( ^{111} \)http://research.microsoft.com/projects/tdrive
\( ^{1111} \)http://ita.ee.lbl.gov/html/contrib/WorldCup.html

---

\( ^{11111} \)http://www.mathcs.emory.edu/aims/FAST/index.html
We run a private data publishing algorithm BD [14] of \( w \)-event privacy, and calculate the ratio of risky \( \ell_u \)-trajectories. We set \( w \) as 10 to 90 with intervals of 20, and \( \ell_u = w \). From the result shown in Table 3 (e.g., the first cell of 9.1% is the ratio of risky 10- trajectories under the protection of 10-event privacy), we can conclude that \( \ell_u \)-trajectories cannot be uniformly protected by \( w \)-event privacy with a given \( \ell_u \) and \( w \), unless \( w \gg \ell_u \) or dense trajectory (data points appear at each cells in Fig. 2). For \( w \gg \ell_u \), \( w \)-event privacy has a higher probability to protect every \( \ell_u \)-trajectory. Two examples are the cases of \( w = 70 \) and 90 in dataset PFlow, and the actual protected length \( \ell_u \) is 39 because the longest trajectory in PFlow is 39 as shown in Table 2. However, we do not know the appropriate \( w \) for different datasets to protect every \( \ell_u \)-trajectory. For dense trajectory, \( w \)-event privacy can protect every \( \ell_u \)-trajectory; however, the users cannot specify different \( \ell_u \).

On the other hand, the ratios of risky trajectories of Uniform, GA+Adj and GA+MMD are all 0%. In other words, they exactly satisfy \( \ell \)-trajectory privacy.

6.2 Utility Evaluation

Figure 5 shows the real and noisy statistics which were published by different algorithms under the same privacy level of \( \ell_u = 20 \) and \( \epsilon = 1 \). We can find that the noisy data published by GA+MMD is more similar to the real data than others.

6.2.1 MAE, MRE and MSE

We quantitatively evaluate the data utility by the following metrics, Mean of Absolute Error (MAE), Mean of Relative Error (MRE), Mean of Square Error (MSE) and KL-divergence [7]. MAE has been given as Eq. (13). In MRE, a user-specified constant \( \delta \) is used to mitigate the effect of excessively small query results. This definition follows from previous work [33], [34]. We set \( \delta = 1 \). MSE is more sensitive to larger errors. The results are shown in Fig. 6, 7, 8 by varying \( \ell_u \) (set \( \epsilon = 1 \)) and varying \( \epsilon \) (set \( \ell_u = 20 \)) respectively.

\[
MRE(R, N) = \frac{1}{|T| + |locs|} \cdot \sum_{i=1}^{|T|} \sum_{j=1}^{|locs|} \frac{|r_i[j] - n_i[j]|}{\max(\delta, r_i[j])} \tag{16}
\]

\[
MSE(R, N) = \frac{1}{|T| + |locs|} \cdot \sum_{i=1}^{|T|} \sum_{j=1}^{|locs|} |r_i[j] - n_i[j]|^2 \tag{17}
\]
Fig. 7  MRE on four datasets by varying $\ell_u$ (10 to 100) and varying $\epsilon$ (0.0001 to 1).

Fig. 8  MSE on four datasets by varying $\ell_u$ (10 to 100) and varying $\epsilon$ (0.0001 to 1).

Uniform's MAE becomes larger almost linearly with the increasing of $\ell_u$, and exponentially with decreasing of $\epsilon$, because $\ell_u$ and $\epsilon$ are the only factors affecting Uniform's performance. For FASTfixed, since the adaptive prediction and correction features are unavailable for achieving $\ell$-trajectory privacy, it is not stable with different $\epsilon$ and $\ell_u$. GA+Adj and GA+MMD are superior to others because of their data-dependent dynamic budget allocation and approximation. However, GA+Adj are susceptible to the similarity between each pair of the adjacent values on the timeline. This is one of the reasons that GA+Adj performs better in dataset WC98 which has the largest number of $|\text{locs}|$ (the similarity is calculated with a factor of $1/|\text{locs}|$; refer to Sect. 5.4). While, GA+MMD is more adaptive because it is capable of searching the most similar value all through the past timeline, not only the adjacent one, to republish it as the current value. Comparing the performances of GA+MMD among the four datasets, the one in PFlow is worst. One potential reason is that the dynamic budget allocation (GA) has a high probability of encountering with the worst case (refer to Sect. 5.3) because the trajectories in PFlow are quite "crowded" (more users but less
6.2.2 KL-Divergence

KL-divergence is widely used to measure the similarity of two probability distributions. It is a non-symmetric measure that computes the information lost when we use published noisy data \( N \) to estimate real data \( R \), as known as relative entropy. To get the probability distribution, we normalize the real and noisy data at each timestamp, which is denoted by \( \tilde{r}_i \) and \( \tilde{n}_i \) respectively.

\[
D_{KL}(R||N) = \frac{1}{|T|} \sum_{t=1}^{|T|} \sum_{j=1}^{|loc|} \ln \left( \frac{\tilde{r}_i[j]}{\tilde{n}_i[j]} \right) \tilde{r}_i[j] \tag{18}
\]

Figure 9 shows the result of KL-divergence between noisy data and real data by varying \( \ell_u \). The lower the better. It shows that the noisy data published by GA+MMD obtains lowest KL-divergence, which means its distribution is more close to original data. This is also a quantitative evidence of the visual comparison of Fig. 5.

6.3 Runtime

It is easy to know the time complexity of \textsc{uniform}, is \( O(|loc|) \). For algorithm \( GA \), since the complexity of Lines 1, 3, 4 and 5 are \( O(1) \), and the complexity of Line 2 is \( O(|us|) \) (to find the maximum value), the overall complexity is \( O(|us|) \). For algorithm \( Adj \), the time complexity is \( O(|loc|) \) because of computing MAE. For algorithm MMD, the most time-consuming operations are Lines 2 and 3. The calculation of the scores in Line 2 needs to access \( |loc| \) amount of data for total \( t \) timestamps, and the calculation in Line 3 involves sorting the scores. Therefore, the overall complexity of MMD are \( O(t + |loc| + t + \log t) \). We notice that if \( t \) is large, the complexity of MMD is high because of comparing \( r_i \) with \( n_i \) where \( i \in [1, t - 1] \). One relaxation of MMD is that, by introducing a heuristic threshold \( t_{period} \), we only compare \( r_i \) with \( n_i \) where \( i \in [t - 1 - t_{period}, t - 1] \). \( t_{period} \) means the possible repetition interval which can be specified by experts. For example, if the expert knows that the data has a repeating pattern every week, then we can set \( t_{period} \) as a week. By this relaxation, \( t \) becomes a constant value \( t_{period} \). Hence, the complexity of relaxed MMD reduces to \( O(|loc|) \).

Therefore, the time complexity of \( GA + Adj \) and \( GA + MMD \) are \( O(|us| + |loc|) \) and \( O(|us| + t + |loc| + t + \log t) \) respectively. The relaxed version of \( GA + MMD \) has the same complexity as \( GA + Adj \). According to [5], real-time algorithm \( FAST_{fixed} \) has complexity \( O(|loc|) \).

Besides \( FAST_{fixed} \), algorithms \textsc{uniform}, \( GA + Adj \) and \( GA + MMD \) are capable of real-time processing and publishing because the algorithms are only need to input data of the current timestamps \( t \) and the previous ones. In the above experiments, we simulated the generating process of real-time data. We make sure that the algorithms are input the data of each timestamp one by one, and output the corresponding private data one by one. Figure 10 shows the overall runtime of each algorithm on the largest dataset WorldCup98.

7. Conclusion

We explored the potential of applying differential privacy to continuously publishing statistics over infinite trajectory streams.

First, we provided a formal definition of \( \ell \)-trajectory privacy as a practical and customizable model. It is a user-level privacy model which allows different users to specify their own \( \ell_u \), even varying \( \ell_u \) at different times. It can be
easily applied to other time-sequenced sensitive data, such as social activities or physiologic data.

Second, to achieve $\ell$-trajectory privacy, we proposed an algorithmic framework to publish statistics in real time. Algorithms GA+Adj and GA+MMD implemented the framework to achieve $\ell$-trajectory privacy in a data-dependent way. The algorithms were optimized towards a specified utility goal of MAE. However, the proposed framework can be implemented towards other utility goals or data mining tasks.

Third, the experiments conducted with real datasets show that privacy and utility performance of the proposed method is better than other competitors. We quantitatively evaluated the privacy level of $\ell_w$-trajectories under the proposed model comparing with a competitor model of $w$-event privacy. The results showed the effectiveness of $\ell$-trajectory privacy. In utility evaluation, algorithm GA+MMD is best under the metrics of MAE, MRE, MSE and KL-divergence.

As future work, we will explore more flexible and practical privacy model. One interesting question is how to specify different privacy level to different sensitive data.

References


Yang Cao received B.S. degree from the School of Software Engineering, Northwestern Polytechnical University, Xi’an, China, in 2008, the M.S. degree from the Graduate School of Informatics, Kyoto University, Kyoto, in 2014. He is currently a Ph.D. student at Graduate School of Informatics, Kyoto University. His research interests are privacy preserving data publishing and mining.

Masatoshi Yoshikawa received the B.E., M.E. and Ph.D. degrees from Department of Information Science, Kyoto University in 1980, 1982 and 1985, respectively. From 1985 to 1993, he was with Kyoto Sangyo University. In 1993, he joined Nara Institute of Science and Technology as an Associate Professor of Graduate School of Information Science. From April 1996 to January 1997, he has stayed at Department of Computer Science, University of Waterloo as a visiting associate professor. From June 2002 to March 2006, he served as a professor at Nagoya University. From April 2006, he has been a professor at Kyoto University. He is a member of ACM and IPSJ.