Protecting Locations with Differential Privacy under Temporal Correlations

Yonghui Xiao
Dept. of Math and Computer Science
Emory University
yonghui.xiao@emory.edu

Li Xiong
Dept. of Math and Computer Science
Emory University
lxiong@emory.edu

ABSTRACT

Concerns on location privacy frequently arise with the rapid development of GPS enabled devices and location-based applications. While spatial transformation techniques such as location perturbation or generalization have been studied extensively, most techniques rely on syntactic privacy models without rigorous privacy guarantee. Many of them only consider static scenarios or perturb the location at single timestamps without considering temporal correlations of a moving user's locations, and hence are vulnerable to various inference attacks. While differential privacy has been accepted as a standard for privacy protection, applying differential privacy in location based applications presents new challenges, as the protection needs to be enforced on the fly for a single user and needs to incorporate temporal correlations between a user's locations.

In this paper, we propose a systematic solution to preserve location privacy with rigorous privacy guarantee. First, we propose a new definition, "δ-location set" based differential privacy, to account for the temporal correlations in location data. Second, we show that the well known ℓ_1-norm sensitivity fails to capture the geometric sensitivity in multidimensional space and propose a new notion, sensitivity hull, based on which the error of differential privacy is bounded. Third, to obtain the optimal utility we present a planar isotropic mechanism (PIM) for location perturbation, which is the first mechanism achieving the lower bound of differential privacy. Experiments on real-world datasets also demonstrate that PIM significantly outperforms baseline approaches in data utility.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

CCS'15, October 12–16, 2015, Denver, Colorado, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3832-5/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2810103.2813640.

Keywords

Location privacy; Location-based services; Differential privacy; Sensitivity hull; Planar isotropic mechanism

1. INTRODUCTION

Technology and usage advances in smartphones with localization capabilities have provided tremendous opportunities for location based applications. Location-based services (LBS) [20, 8] range from searching points of interest to location-based games and location-based commerce. Location-based social networks allow users to share locations with friends, to find friends, and to provide recommendations about points of interest based on their locations.

One major concern of location based applications is location privacy [3]. To use these applications, users have to provide their locations to the respective service providers or other third parties. This location disclosure raises important privacy concerns since digital traces of users' whereabouts can expose them to attacks ranging from unwanted location based spams/scams to blackmail or even physical danger.

Gaps in Existing Works and New Challenges. Many location privacy protection mechanisms have been proposed during the last decade [23, 15] in the setting of LBS or continual location sharing. In such setting, a user sends her location to untrusted service providers or other parties in order to obtain some services (e.g. to find the nearest restaurant). One solution is Private Information Retrieval (PIR) technique, based on cryptography instead of revealing individual locations (e.g. [32]). However, such technique tends to be computationally expensive and not practical in addition to requiring different query plans to be designed for different query types.

Most solutions proposed in the literature are based on location obfuscation which transforms the exact location of a user to an area (location generalization) or a perturbed location (location perturbation) (e.g. [14, 1]). Unfortunately, most spatial transformation techniques proposed so far rely on syntactic privacy models such as k-anonymity, or ad-hoc uncertainty models, and do not provide rigorous privacy. Many of them only consider static scenarios or perturb the location at single timestamps without considering the temporal correlations of a moving user's locations, and hence are vulnerable to various inference attacks. Consider the following examples.

I Suppose a user moved from school to the cafeteria (where "$\star$" is) in Figure 1 (left). Three perturbed locations were released by selecting a point probabilistically in each of

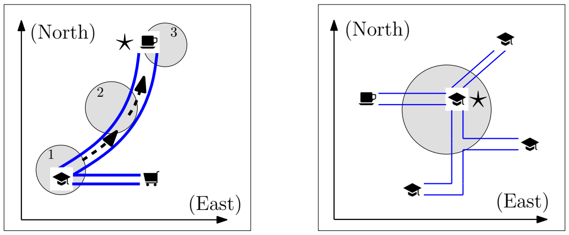


Figure 1: Examples of privacy breach caused by temporal correlations of user locations

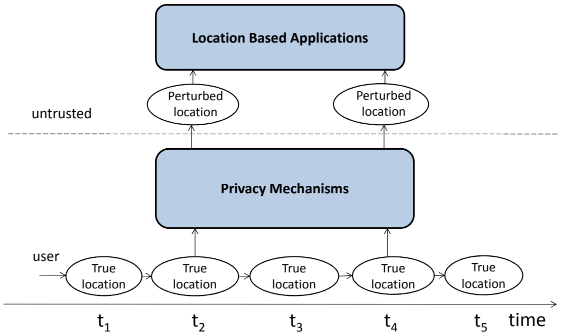


Figure 2: Problem setting

the three circles (by some spatial cloaking methods). Even though the individual locations were seemingly protected at each timestamp, considering them together with road constraints or the user's moving pattern will enable an adversary to accurately figure out the user is in the cafeteria, resulting in privacy breach.

II Suppose a user's location "$\star$" is protected in a circle as shown in Figure 1 (right). If by estimation based on previous locations the user can only be in the five places at current timestamp as shown in the figure, then the obfuscated location actually exposes the true location. Thus technically, the radius of the circle (in location obfuscation) should be subject to temporal correlations.

While such temporal correlations can be commonly modeled by Markov chain [37, 17, 25], and few works have considered such Markov models [37, 17], it remains a challenge to provide rigorous privacy protection under temporal correlations for continual location sharing.

Differential privacy [9] has been accepted as a standard for privacy preservation. It was originally proposed to protect aggregated statistics of a dataset by bounding the knowledge gain of an adversary whether a user opts in or out of a dataset. Applying differential privacy for location protection is still at an early stage. In particular, several works (e.g. [6, 33, 12]) have applied differential privacy on location or trajectory data but in a *data publishing* or *data aggregation* setting. In this setting, a trusted data publisher with access to a set of location snapshots or user trajectories publishes an *aggregate* or synthetic view of the original data while guaranteeing user-level differential privacy, i.e. protecting the presence of a user's location or entire trajectory in the aggregated data.

There are several challenges in applying differential privacy in the new setting of continual location sharing. First, standard differential privacy only protects *user-level* privacy (whether a user opts in or out of a dataset); while in our setting, the protection needs to be enforced on the fly for *a single user*. Second, as shown in Figure 1, temporal correlations based on road networks or the user's moving patterns exist and the privacy guarantee needs to account for such correlations. Finally, there is no effective location release mechanism with differential privacy under such model.

Contributions. In this paper, we propose a systematic solution to preserve location privacy with differential privacy guarantee. As shown in Figure 2, we consider a moving user with sensitive location stream who needs to share her locations to an untrusted location-based application host or other parties. A user's true locations are only known by the user. The "sanitized" locations released by the privacy mechanisms are observable to the service providers, as well as adversaries. To enable private location sharing, we address (and take advantage of) the temporal correlations, which can not be concealed from adversaries and hence are assumed to be public. Our contributions are summarized as follows.

First, we propose "δ-location set" based differential privacy to protect the true location at every timestamp. The "neighboring databases" in standard differential privacy are any two databases under one operation: adding or removing a record (or a user). However, this is not applicable in a variety of settings [21, 5], which leads to new and extended notions such as δ-neighborhood [13] or event-level [11] differential privacy. In our problem, location changes between two consecutive timestamps are determined by temporal correlations modeled through a Markov chain [37, 17]. Accordingly we propose a "δ-location set" to include all probable locations (where the user might appear). Intuitively, to protect the true location, we only need to "hide" it in the δ-location set in which any pairs of locations are not distinguishable.

Second, we show that the well known ℓ_1-norm sensitivity in standard differential privacy fails to capture the geometric sensitivity in multidimensional space. Thus we propose a new notion, sensitivity hull, to capture the geometric meaning of sensitivity. We also prove that the lower bound of error is determined by the sensitivity hull.

Third, we present an efficient location perturbation mechanism, called planar isotropic mechanism (PIM), to achieve δ-location set based differential privacy.

I To our knowledge, PIM is the first optimal mechanism that can achieve the lower bound of differential privacy[1]. The novelty is that in two-dimensional space we efficiently transform the sensitivity hull to its isotropic position such that the optimality is guaranteed.

II We also implement PIM on real-world datasets, showing that it preserves location utility for location based queries and significantly outperforms the baseline Laplace mechanism (LM).

2. PRELIMINARIES

We denote scalar variables by normal letters, vectors by bold lowercase letters, and matrices by bold capital letters. We use $||\cdot||_p$ to denote the ℓ_p norm, $\mathbf{x}[i]$ to denote the ith

[1] The state-of-art differentially private mechanisms [18, 4] for linear queries can be $O(log(d))$ approximately optimal where d is the number of dimensions.

$\mathbf{s}_i$	a cell in a partitioned map, $i = 1, 2, \cdots, m$
$\mathbf{u}$, $\mathbf{x}$	location in state and map coordinates
$\mathbf{u}^*$, $\mathbf{x}^*$	true location of the user
$\mathbf{z}$	the released location in map coordinate
$\mathbf{p}_t^-$	prior probability (vector) at timestamp t
$\mathbf{p}_t^+$	posterior probability (vector) at timestamp t
$\Delta\mathbf{X}$	δ-location set
K	sensitivity hull

Table 1: Denotation

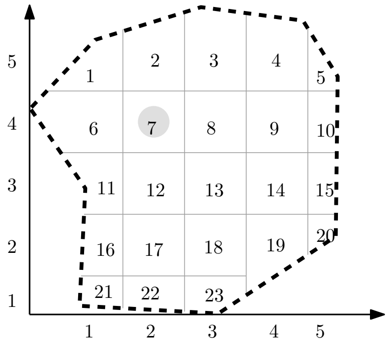


Figure 3: Two coordinate systems

element of $\mathbf{x}$, $\mathbb{E}()$ to denote the expectation, $\mathbf{x}^T$ to denote the transpose of vector $\mathbf{x}$. Table 1 summarizes some important symbols for convenience.

2.1 Two Coordinate Systems

We use two coordinate systems, state coordinate and map coordinate, to represent a location for the Markov model and map model respectively. Denote $\mathcal{S}$ the domain of space. If we partition $\mathcal{S}$ into the finest granularity, denoted by "cell", then $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_m\}$ where each $\mathbf{s}_i$ is a unit vector with the ith element being 1 and other $m-1$ elements being 0. Each cell can represent a state (location) of a user. On the other hand, If we view the space as a map with longitude and latitude, then a 2×1 vector can be used to represent a user's location $\mathbf{x}$ with two components $\mathbf{x}[1]$ and $\mathbf{x}[2]$. Figure 3 shows an example using these two coordinate systems. If a user is in $\mathbf{s}_7$, the state coordinate and map coordinate are shown as follows. Note that the two coordinate systems can be transformed to each other. We skip how to transform them and treat $\mathbf{u}$ and $\mathbf{x}$ interchangeable.

$$\mathbf{u} = \mathbf{s}_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$
$$\mathbf{x} = [2, 4]^T \textit{ with } \mathbf{x}[1] = 2 \textit{ and } \mathbf{x}[2] = 4$$

As time evolves, the trace of a user can be represented by a series of locations, $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_t$ in map coordinate or $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_t$ in state coordinate.

2.2 Mobility and Inference Model

Our approach uses Markov chain [37, 17, 25] to model the temporal correlations between user's locations. Other constraints, such as road network, can also be captured by it. However, we note that Markov model, as well as any mobility models, may have limits in terms of predicability [38]. And we will discuss our solution to address these limits later.

In our problem setting, a user's true locations are unobservable, i.e. only known by the user. The "sanitized" locations released by the perturbation mechanism are observable to the service provider, as well as adversaries. Thus from an adversarial point of view, this process is a Hidden Markov Model (HMM).

At timestamp t, we use a vector $\mathbf{p}_t$ to denote the probability distribution of a user's location (in each cell). Formally,

$$\mathbf{p}_t[i] = Pr(\mathbf{u}_t^* = \mathbf{s}_i) = Pr(\mathbf{x}_t^* = \textit{the coordinate of } \mathbf{s}_i)$$

where $\mathbf{p}_t[i]$ is the ith element in $\mathbf{p}_t$ and $\mathbf{s}_i \in \mathcal{S}$. In the example of Figure 3, if the user is located in cells $\{\mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_7, \mathbf{s}_8\}$ with a uniform distribution, the probability vector can be expressed as follows.

$$\mathbf{p} = \begin{bmatrix} 0 & 0.25 & 0.25 & 0 & 0 & 0 & 0.25 & 0.25 & 0 & \cdots & 0 \end{bmatrix}$$

Transition Probability. We use a matrix $\mathbf{M}$ to denote the probabilities that a user moves from one location to another. Let m_{ij} be the element in $\mathbf{M}$ at ith row and jth column. Then m_{ij} represents the probability that a user moves from cell i to cell j. Given probability vector $\mathbf{p}_{t-1}$, the probability at timestamp t becomes $\mathbf{p}_t = \mathbf{p}_{t-1}\mathbf{M}$. We assume the transition matrix $\mathbf{M}$ is given in our framework.

Emission Probability. If given a true location $\mathbf{u}_t^*$, a mechanism releases a perturbed location $\mathbf{z}_t$, then the probability $Pr(\mathbf{z}_t|\mathbf{u}_t^* = \mathbf{s}_i)$ is called "emission probability" in HMM. This probability is determined by the release mechanism and should be transparent to adversaries.

Inference and Evolution. At timestamp t, we use $\mathbf{p}_t^-$ and $\mathbf{p}_t^+$ to denote the prior and posterior probabilities of a user's location before and after observing the released $\mathbf{z}_t$ respectively. The prior probability can be derived by the posterior probability at previous timestamp $t-1$ and the Markov transition matrix as $\mathbf{p}_t^- = \mathbf{p}_{t-1}^+\mathbf{M}$. Given $\mathbf{z}_t$, the posterior probability can be computed using Bayesian inference as follows. For each cell $\mathbf{s}_i$:

$$\mathbf{p}_t^+[i] = Pr(\mathbf{u}_t^* = \mathbf{s}_i|\mathbf{z}_t) = \frac{Pr(\mathbf{z}_t|\mathbf{u}_t^* = \mathbf{s}_i)\mathbf{p}_t^-[i]}{\sum_j Pr(\mathbf{z}_t|\mathbf{u}_t^* = \mathbf{s}_j)\mathbf{p}_t^-[j]} \quad (1)$$

The inference at each timestamp can be efficiently computed by forward-backward algorithm in HMM, which will be incorporated in our framework.

2.3 Differential Privacy and Laplace Mechanism

Definition 2.1 (Differential Privacy). *A mechanism $\mathcal{A}$ satisfies ϵ-differential privacy if for any output $\boldsymbol{z}$ and any neighboring databases $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ where $\boldsymbol{x}_2$ can be obtained from $\boldsymbol{x}_1$ by either adding or removing one record*[2]*, the following holds*

$$\frac{Pr(\mathcal{A}(\boldsymbol{x}_1) = \boldsymbol{z})}{Pr(\mathcal{A}(\boldsymbol{x}_2) = \boldsymbol{z})} \leq e^{\epsilon}$$

Laplace mechanism [10] is commonly used in the literature to achieve differential privacy. It is built on the ℓ_1-norm sensitivity, defined as follows.

[2] This is the definition of unbounded differential privacy [21]. Bounded neighboring databases can be obtained by changing the value of exactly one record.

Definition 2.2 (ℓ_1-norm Sensitivity). *For any query $f(\boldsymbol{x})$: $\boldsymbol{x} \to \mathbb{R}^d$, ℓ_1-norm sensitivity is the maximum ℓ_1 norm of $f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)$ where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are any two instances in neighboring databases.*

$$S_f = \max_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \text{ neighboring databases}} ||f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)||_1$$

where $||\cdot||_1$ denotes ℓ_1 norm.

A query can be answered by $f(\mathbf{x}) + Lap(S_f/\epsilon)$ to achieve ϵ-differential privacy, where $Lap() \in \mathbb{R}^d$ are i.i.d. random noises drawn from Laplace distribution.

2.4 Utility Metrics

To measure the utility of the perturbed locations, we follow the analysis of metrics in [37] and adopt the expected distance (called "correctness" in [37]) between the true location $\mathbf{x}^*$ and the released location $\mathbf{z}$ as our utility metric.

$$\text{Error} = \sqrt{\mathbb{E}||\mathbf{z} - \mathbf{x}^*||_2^2} \quad (2)$$

In addition, we also study the utility of released locations in the context of location based queries such as finding nearest k Points of Interest (POI). We will use precision and recall as our utility metrics in this context which we will explain later in the experiment section.

2.5 Convex Hull

Our proposed sensitivity hull is based on the well studied notion of convex hull in computational geometry. We briefly provide the definition here.

Definition 2.3 (Convex Hull). *Given a set of points $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$, the convex hull of $\boldsymbol{X}$ is the smallest convex set that contains $\boldsymbol{X}$.*

Note that a convex hull in two-dimensional space is a polygon (also called "convex polygon" or "bounding polygon"). Because it is well-studied and implementations are also available [31], we skip the details and only use $Conv(\mathbf{X})$ to denote the function of finding the convex hull of $\mathbf{X}$.

3. PRIVACY DEFINITION

To apply differential privacy in the new setting of continual location sharing, we conduct a rigorous privacy analysis and propose δ-location set based differential privacy in this section.

3.1 δ-Location Set

The nature of differential privacy is to "hide" a true database in "neighboring databases" when releasing a noisy answer from the database. In standard differential privacy, neighboring databases are obtained by either adding or removing a record (or a user) in a database. However, this is not applicable in our problem. Thus we propose a new notion, δ-location set, to hide the true location at every timestamp.

Motivations. We first discuss the intuitions that motivates our definition.

First, because the Markov model is assumed to be public, adversaries can make inference using previously released locations. Thus we, as data custodians in a privacy mechanism, can also track the temporal inference at every timestamp. At any timestamp, say t, a prior probability of the user's current location can be derived, denoted by $\mathbf{p}_t^-$ as follows.

$$\mathbf{p}_t^-[i] = Pr(\mathbf{u}_t^* = \mathbf{s}_i | \mathbf{z}_{t-1}, \cdots, \mathbf{z}_1)$$

Similar to hiding a database in its neighboring databases, we can hide the user's true location in possible locations (where $\mathbf{p}_t^-[i] > 0$). On the other hand, hiding the true location in any impossible locations (where $\mathbf{p}_t^-[i] = 0$) is a lost cause because the adversary already knows the user cannot be there.

Second, a potential shortcoming of Markov model is that the probability distribution may converge to a stationary distribution after a long time (e.g. an ergodic Markov chain). Intuitively, a user's possible locations can eventually cover the entire map given enough time. Hiding a location in a large area may yield a significantly perturbed location that is not useful at all.

According to [16], moving patterns of human have a "high degree" of temporal and spatial regularity. Hence if people tend to go to a number of highly frequented locations, our privacy notion should also emphasize protecting the more probable locations in Markov model.

δ-Location Set. With above motivations, we define δ-location set at any timestamp t, denoted as $\Delta\mathbf{X}_t$. Essentially, δ-location set reflects a set of probable locations the user might appear (by leaving out the locations of small probabilities).

Definition 3.1 (δ-Location Set). *Let $\boldsymbol{p}_t^-$ be the prior probability of a user's location at timestamp t. δ-location set is a set containing minimum number of locations that have prior probability sum no less than $1 - \delta$.*

$$\Delta\boldsymbol{X}_t = min\{\boldsymbol{s}_i | \sum_{\boldsymbol{s}_i} \boldsymbol{p}_t^-[i] \geq 1 - \delta\}$$

For example, if $\mathbf{p}_t^- = [0.3, 0.4, 0.05, 0.2, 0.03, 0.02]$ corresponding to $[\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4, \mathbf{s}_5, \mathbf{s}_6]$, then $\Delta\mathbf{X} = \{\mathbf{s}_2, \mathbf{s}_1, \mathbf{s}_4\}$ when $\delta = 0.1$; $\Delta\mathbf{X} = \{\mathbf{s}_2, \mathbf{s}_1, \mathbf{s}_4, \mathbf{s}_3\}$ when $\delta = 0.05$.

Note that if $\delta = 0$ the location set contains all possible locations. Thus 0-location set preserves the strongest privacy.

Drift. Because δ-location set represents the most probable locations, a drawback is that the true location may be filtered out with a small probability (technically, $Pr(\mathbf{x}^* \notin \Delta\mathbf{X}) = \delta$). Same situation may also occur if the Markov model is not accurate enough in practice due to its limit in predicability, as we mentioned earlier. Therefore, we denote this phenomenon as "drift" and handle it with the following surrogate approach.

Surrogate. When a drift happens, we use a surrogate location in $\Delta\mathbf{X}$ as if it is the "true" location in the release mechanism.

Definition 3.2 (Surrogate). *A surrogate $\tilde{\boldsymbol{x}}$ is the cell in $\Delta\boldsymbol{X}$ with the shortest distance to the true location $\boldsymbol{x}^*$.*

$$\tilde{\boldsymbol{x}} = \underset{\boldsymbol{s} \in \Delta\boldsymbol{X}}{argmin}\, dist(\boldsymbol{s}, \boldsymbol{x}^*)$$

where function dist() denotes the distance between two cells.

Note that the surrogate approach does not leak any information of the true location, explained as follows. If $\mathbf{x}^* \in \Delta\mathbf{X}$, then $\mathbf{x}^*$ is protected in $\Delta\mathbf{X}$; if not, $\tilde{\mathbf{x}}$ is protected in $\Delta\mathbf{X}$.

Using surrogate does not reveal whether $\mathbf{x}^*$ is in $\Delta\mathbf{X}$ or not. Because in any location release mechanisms $\mathbf{x}^*$ is treated as a black box (oblivious to adversaries), replacing $\mathbf{x}^*$ with $\tilde{\mathbf{x}}$ is also a black box. We formally prove the privacy guarantee in Theorem 5.1.

In some cases, a surrogate may be far from the true location. Then the released location may not be useful. Therefore, we also measure the distance between released location and true location in our experiment to reflect the long-term effect of surrogate.

3.2 Differential Privacy on δ-Location Set

We define differential privacy based on δ-location set, with the intuition that the released location $\mathbf{z}_t$ will not help an adversary to differentiate any instances in the δ-location set.

Definition 3.3 (Differential Privacy). *At any timestamp t, a randomized mechanism $\mathcal{A}$ satisfies ϵ-differential privacy on δ-location set $\Delta\boldsymbol{X}_t$ if, for any output $\boldsymbol{z}_t$ and any two locations $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in $\Delta\boldsymbol{X}_t$, the following holds:*

$$\frac{Pr(\mathcal{A}(\boldsymbol{x}_1) = \boldsymbol{z}_t)}{Pr(\mathcal{A}(\boldsymbol{x}_2) = \boldsymbol{z}_t)} \leq e^{\epsilon} \tag{3}$$

Above definition guarantees the true location is always protected in δ-location set at every timestamp. In another word, the released location $\mathbf{z}_t$ is differentially private at timestamp t for continual location sharing under temporal correlations. For other application settings, like protecting the trace or trajectory of a user, we defer the investigation to future works.

3.3 Adversarial Knowledge

In reality, there may be a variety of adversaries with all kinds of prior knowledge. Accordingly, we prove that for the problem of continual location sharing differential privacy is equivalent to adversarial privacy, first studied in [35].

Definition 3.4 (Adversarial Privacy). *A mechanism is ϵ-adversarially private if for any location $\boldsymbol{s}_i \in \mathcal{S}$, any output $\boldsymbol{z}$ and any adversaries knowing the true location is in $\Delta\boldsymbol{X}$, the following holds:*

$$\frac{Pr(\boldsymbol{u}_t^* = \boldsymbol{s}_i|\boldsymbol{z}_t)}{Pr(\boldsymbol{u}_t^* = \boldsymbol{s}_i)} \leq e^{\epsilon} \tag{4}$$

where $Pr(\boldsymbol{u}_t^* = \boldsymbol{s}_i)$ and $Pr(\boldsymbol{u}_t^* = \boldsymbol{s}_i|\boldsymbol{z}_t)$ are the prior and posterior probabilities of any adversaries.

We can show Definition 3.3 is equivalent to adversarial privacy for continual location sharing, which can be derived from the PTLM property [35].

Theorem 3.1. *For the problem of continual location sharing, Definition 3.3 is equivalent to Definition 3.4.*

Definition 3.4 limits the information gain for adversaries knowing the condition $\mathbf{x}_t^* \in \Delta\mathbf{X}$. If $\mathbf{x}_t^* \notin \Delta\mathbf{X}$, our framework reveals no extra information due to the surrogate approach. Thus adversarial knowledge can be bounded, discussed as follows.

Standard Adversary. For adversaries who have exactly the same Markov model and keep tracking all the released locations, their knowledge is also the same as our model (with location inference in Section 5.3). In this case, differential privacy and adversarial privacy are guaranteed, and we know exactly the adversarial knowledge, which in fact can be controlled by adjusting ϵ.

Weak Adversary. For adversaries who have little knowledge about the user, the released locations may help them obtain more information. With enough time to evolve, they may converge to standard adversaries eventually. But their adversarial knowledge will not exceed standard adversaries.

Strong Adversary. For adversaries who have additional information, the released location from differential privacy may not be very helpful. Specifically, a strong adversary with auxiliary information may have more accurate prior knowledge. However, if the adversary cannot identify the true location so that $Pr(\mathbf{u}^* = \mathbf{s}_i) = 1$ for any $\mathbf{s}_i \in \mathcal{S}$, Definition 3.4 is always satisfied. On the other hand, if an "omnipotent" adversary already knows the true location, then no mechanism can actually protect location privacy.

3.4 Comparison with Other Definitions

Differential Privacy. Since the concept of neighboring databases is not generally applicable (as discussed earlier), induced neighborhood [21], metric based neighborhood [5] and δ-neighborhood [13] were proposed. The general idea is that the neighborhood can be formulated by some constraints of data or distance (metric) functions instead of adding or removing a record. However, applying these neighborhood based differential privacy is not feasible in our model because there is only one sole tuple (location) at each timestamp without any "neighbors". Hence we define δ-location set to extend the notion of "neighborhood".

Geo-indistinguishability. Another closely related definition is the Geo-indistinguishability [1], which protects a user's location within a radius (circle) with a "generalized differential privacy" guarantee. In other words, the neighborhood is defined with Euclidian distance. Nevertheless, such spatial perturbation technique may not be reasonable in reality. For example, as shown in Figure 1, the "generalized differential privacy" can still be breached given the road network constraint or user's moving pattern (which is represented by Markov model). Thus location privacy must be protected under temporal correlations.

Blowfish privacy. Our privacy definition shares the same insight as the unconstrained Blowfish privacy framework [19] in statistical data release context, which uses secret pairs and privacy policy to build a subset of possible database instances as "neighbors". We show that δ-location set based differential privacy can be instantiated as a special case of unconstrained Blowfish privacy at each timestamp.

Theorem 3.2. *Let $\mathcal{S}$ be the domain of all possible locations. Let G be a complete graph where each node denotes a location in $\mathcal{S}$. Let $\Delta\boldsymbol{X}$ be a condition such that $\boldsymbol{x}^* \in \Delta\boldsymbol{X}$. At each timestamp, Definition 3.3 is equivalent to $\{\epsilon, \{\mathcal{S}, G, \Delta\boldsymbol{X}\}\}$-Blowfish privacy.*

3.5 Discussion

Learning Markov Model. Existing methods such as the knowledge construction module in [37] or EM method in HMM can be used to acquire the transition matrix $\mathbf{M}$, which will not be discussed in this paper. However, depending on the power of adversaries, two typical $\mathbf{M}$ can be learned.

I Popular $\mathbf{M}$ can be learned from public transit data.

II Personal $\mathbf{M}$ can be derived with personal transit data[3].

No matter which $\mathbf{M}$ is adopted in our framework, the adversarial knowledge is always bounded, as discussed before. However, the usefulness of released locations may vary for different adversaries. We also compare the two models in our experiments.

Compobility. Since we only need to release one perturbed location at a timestamp, the sequential composition [28] is not applicable. Otherwise, for multiple releases at a timestamp the composition of ϵ holds. On the other hand, given a series of perturbed locations $\{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_t\}$ released from timestamp 1 to t, a new problem is how to protect and measure the overall privacy guarantee of the entire trace. We defer this to future work.

4. SENSITIVITY HULL

The notion of sensitivity indicates the differences between any two query answers from two instances in neighboring databases. However, in multidimensional space, we show that ℓ_1-norm sensitivity (in Definition 2.2) fails to capture the exact sensitivity. Thus we propose a new notion, sensitivity hull. Note that sensitivity hull is an independent notion from the context of location privacy and can be plugged in any data-independent perturbation mechanisms.

4.1 Sensitivity Hull

To derive the meaning of sensitivity, let us consider the following example in traditional setting of differential privacy.

EXAMPLE 4.1. *Assume we have an employee table T with attributes gender and income. Then we answer the following query workload f:*

$$f_1: \text{ Select count}(*) \text{ from } T \text{ where gender} = \text{“female”}$$

$$f_2: \text{ Select count}(*) \text{ from } T \text{ where income} > 50000$$

Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be neighboring databases so that $\mathbf{x}_1$ is equal to $\mathbf{x}_2$ adding or removing *a random user*. Suppose $f(\mathbf{x}_2) = [10, 20]^T$. Then the possible answers for $f(\mathbf{x}_1)$ could be one of the following columns, from which Δf can be derived.

$$f(\mathbf{x}_1) = \begin{bmatrix} 11 & 10 & 10 & 11 & 9 & 9 & 10 \\ 21 & 21 & 20 & 20 & 20 & 19 & 19 \end{bmatrix}$$

$$\Delta f = f(\mathbf{x}_1) - f(\mathbf{x}_2) = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & -1 & -1 \end{bmatrix}$$

$$S_f = max||\Delta f||_1 = 2 \quad (\ell_1\text{-norm sensitivity})$$

In Figure 4, the dashed lines form the set of $||\Delta f||_1 = 2$ because the ℓ_1-norm sensitivity is 2. However, Δf only consists of all the “•” points. It is obvious that the ℓ_1-norm sensitivity exaggerates the “real sensitivity”. To capture the geometric representation of Δf in multidimensional space, we define sensitivity hull (the solid lines in Figure 4) as follows.

DEFINITION 4.1 (SENSITIVITY HULL). *The sensitivity hull of a query f is the convex hull of Δf where Δf is the set of $f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)$ for any pair $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in δ-location set $\Delta \boldsymbol{X}$.*

$$K = Conv\,(\Delta f)$$
$$\Delta f = \bigcup_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \Delta \boldsymbol{X}} (f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2))$$

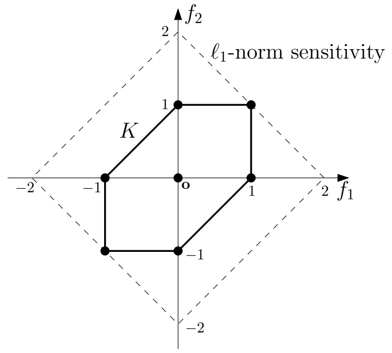


Figure 4: Sensitivity hull of Example 4.1. Solid lines denote the sensitivity hull K; dashed lines are the ℓ_1-norm sensitivity.

THEOREM 4.1. *A sensitivity hull K is centrally symmetric: if $\boldsymbol{v} \in K$ then $-\boldsymbol{v} \in K$.*

THEOREM 4.2. *If data $\boldsymbol{x}$ is in discrete domain, then for any $f : \boldsymbol{x} \to \mathbb{R}^d$, the sensitivity hull of f is a polytope in $\mathbb{R}^d$.*

4.2 Error Bound of Differential Privacy

We extend the error bound of differential privacy in database context [18] to our location setting using sensitivity hull.

LEMMA 4.1. *Suppose $\boldsymbol{F} : \mathbb{R}^N \to \mathbb{R}^d$ is a linear function. When neighboring databases are obtained by adding or removing a record, the sensitivity hull K of $\boldsymbol{F}$ is a polytope $\boldsymbol{F}\boldsymbol{B}_1^N$ where $\boldsymbol{B}_1^N$ is the N-dimensional unit ℓ_1 ball.*

THEOREM 4.3 (LOWER BOUND). *Let K be the sensitivity hull of δ-location set $\Delta \boldsymbol{X}$. To satisfy Definition 3.3, every mechanism must have*

$$\text{ERROR} \geq \Omega\left(\frac{1}{\epsilon}\sqrt{\text{AREA}(K)}\right)$$

where AREA(K) *is the area of K.*

5. LOCATION RELEASE ALGORITHM

5.1 Framework

The framework of our proposed location release algorithm is shown in Algorithm 1. At each timestamp, say t, we compute the prior probability vector $\mathbf{p}_t^-$. If the location needs to be released, we construct a δ-location set $\Delta\mathbf{X}_t$. Then if the true location $\mathbf{x}^*$ is excluded in $\Delta\mathbf{X}_t$ (a drift), we use surrogate to replace $\mathbf{x}^*$. Next a differentially private mechanism (like Algorithm 2 which will be presented next) can be adopted to release a perturbed location $\mathbf{z}_t$. In the meantime, the released $\mathbf{z}_t$ will also be used to update the posterior probability $\mathbf{p}_t^+$ (in the equation below) by Equation (1), which subsequently will be used to compute the prior probability for the next timestamp $t+1$. Then at timestamp $t+1$, the above process is repeated.

$$\mathbf{p}_t^+[i] = Pr(\mathbf{u}_t^* = \mathbf{s}_i | \mathbf{z}_t, \mathbf{z}_{t-1}, \cdots, \mathbf{z}_1)$$

[3] For example, mobile apps, like Google Now, may have a user's location history to derive the user's moving pattern.

Algorithm 1 Framework

```
Require: $\epsilon_t$, $\delta$, $\mathbf{M}$, $\mathbf{p}^+_{t-1}$, $\mathbf{x}^*_t$
 1: $\mathbf{p}^-_t \leftarrow \mathbf{p}^+_{t-1}\mathbf{M}$;                        ▷ Markov transition
 2: if location needs to be released then
 3:     Construct $\Delta\mathbf{X}_t$;                      ▷ $\delta$-location set
 4:     if $\mathbf{x}^*_t \notin \Delta\mathbf{X}_t$ then                      ▷ a drift
 5:         $\mathbf{x}^*_t \leftarrow$ surrogate;
 6:     end if
 7:     $\mathbf{z}_t \leftarrow$ALGORITHM 2($\epsilon_t$, $\Delta\mathbf{X}_t$, $\mathbf{x}^*_t$);         ▷ release $\mathbf{z}_t$
 8:     Derive posterior probability $\mathbf{p}^+_t$ by Equation (1);
 9: end if
10: return ALGORITHM 1($\epsilon_{t+1}$, $\delta$, $\mathbf{M}$, $\mathbf{p}^+_t$, $\mathbf{x}^*_{t+1}$);
      ▷ go to next timestamp
```

THEOREM 5.1. *At any timestamp t, Algorithm 1 is ϵ_t-differentially private on 0-location set.*

PROOF. It is equivalent to prove adversarial privacy on 0-location set, which includes all possible locations. If $\mathbf{x}^*_t \in \Delta\mathbf{X}_t$, then $\mathbf{z}_t$ is generated by $\mathbf{x}^*_t$. By Theorem 5.3, $\mathbf{z}_t$ is ϵ_t-differentially private. So $\frac{Pr(\mathbf{u}^*_t=\mathbf{s}_i|\mathbf{z}_t)}{Pr(\mathbf{u}^*_t=\mathbf{s}_i)} \leq e^\epsilon$. When $\mathbf{x}^*_t \notin \Delta\mathbf{X}_t$, then a surrogate $\tilde{\mathbf{x}}_t$ replaces $\mathbf{x}^*_t$. Then

$$\frac{Pr(\mathbf{u}^*_t=\mathbf{s}_i|\mathbf{z}_t)}{Pr(\mathbf{u}^*_t=\mathbf{s}_i)} = \frac{\sum_k Pr(\mathbf{u}^*_t=\mathbf{s}_i|\tilde{\mathbf{x}}_t=\mathbf{s}_k)Pr(\tilde{\mathbf{x}}_t=\mathbf{s}_k|\mathbf{z}_t)}{\sum_k Pr(\mathbf{u}^*_t=\mathbf{s}_i|\tilde{\mathbf{x}}_t=\mathbf{s}_k)Pr(\tilde{\mathbf{x}}_t=\mathbf{s}_k)} \leq e^\epsilon$$

Therefore, by equivalence (Theorem 3.1) Algorithm 1 is ϵ_t-differentially private on 0-location set. □

Laplace Mechanism. With the ℓ_1-norm sensitivity in Definition 2.2, Laplace mechanism (LM) can be adopted in Line 7 of Algorithm 1. The problem of this approach is that it will over-perturb a location because ℓ_1-norm sensitivity could be much larger than the sensitivity hull, as discussed in Section 4. We use LM with δ-location set as a baseline in our experiment.

5.2 Planar Isotropic Mechanism

Because we showed (in Lemma 4.1) that the sensitivity hull of a query matrix is a polytope (polygon in our two-dimensional location setting), the state-of-art K-norm based mechanism [18, 4, 30] can be used.

DEFINITION 5.1 (K-NORM MECHANISM [18]). *Given a linear function $\boldsymbol{F}: \mathbb{R}^N \rightarrow \mathbb{R}^d$ and its sensitivity hull K, a mechanism is K-norm mechanism if for any output $\boldsymbol{z}$, the following holds:*

$$Pr(\boldsymbol{z}) = \frac{1}{\Gamma(d+1)\mathrm{VOL}(K/\epsilon)} exp\left(-\epsilon||\boldsymbol{z}-\boldsymbol{F}\boldsymbol{x}^*||_K\right) \tag{5}$$

where $\boldsymbol{F}\boldsymbol{x}^*$ is the true answer, $||\cdot||_K$ is the (Minkowski) norm of K, $\Gamma()$ is Gamma function and VOL() *indicates volume.*

However, standard K-norm mechanism was designed for high-dimensional structure of sensitivity hull, whereas in our problem a location is only two-dimensional. Thus we can further optimize K-norm mechanism to achieve the lower bound of differential privacy. We propose a Planar Isotropic Mechanism (PIM) based on K-norm mechanism as follows.

Rationale. The rationale of PIM is that in two-dimensional space we efficiently transform the sensitivity hull to its isotropic position[4] so that the optimality is guaranteed.

THEOREM 5.2. *[18] If the sensitivity hull K is in C-approximately isotropic position, then K-norm mechanism has error $O(C)\mathrm{LB}(K)$ where $\mathrm{LB}(K)$ is the lower bound of differential privacy.*

From Theorem 5.2, we know that K-norm mechanism would be the optimal solution if the sensitivity hull K is in isotropic position, denoted by K_I. Although in high-dimensional space transforming a convex body to its isotropic position is extremely expensive, it is feasible in two-dimensional space. To this end, we need the following corollary (which can be derived from [36, 27]).

COROLLARY 5.1 (ISOTROPIC TRANSFORMATION). *For any convex body K in $\mathbb{R}^2$, any integer $p \geq 1$, there is an absolute constant c such that if $l \geq 4cp^2$, with probability at least $1-2^{-p}$, $K_I = \boldsymbol{T}K$ is in isotropic position.*

$$\boldsymbol{T} = \left(\frac{1}{l}\sum_{i=1}^{l}\boldsymbol{y}_i\boldsymbol{y}_i^T\right)^{-\frac{1}{2}} \tag{6}$$

where $\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_l$ are independent random points uniformly distributed in K.

Therefore, the isotropic transformation of any sensitivity hull K can be fulfilled by sampling, which is a trivial task in two-dimensional space. For instance, a hit-and-run algorithm [26] only takes $O(log^3(1/\delta))$ time where δ is an error parameter. We skip the sampling details and refer readers to the survey paper of Santosh Vempala [39] for a complete study.

Algorithm. As an overview, PIM involves the following steps:

(1) Compute sensitivity hull K from $\Delta\mathbf{X}$;

(2) Transform K to isotropic position K_I;

(3) generating a noise in the space of K_I by K-norm mechanism;

(4) Transform to the original space.

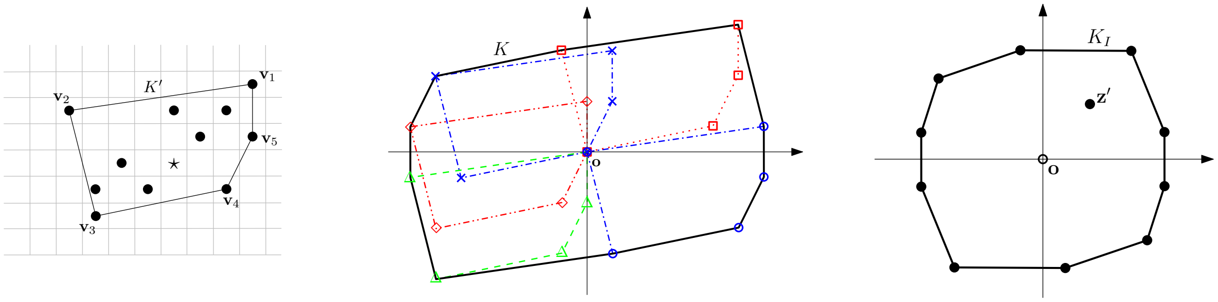
We first describe how to compute sensitivity hull K. Suppose we have a δ-location set $\Delta\mathbf{X}$ at a timestamp. We can first derive the convex hull of $\Delta\mathbf{X}$, denoted by $K' = Conv(\Delta\mathbf{X})$. For example, in Figure 5a, the convex hull K' is shown by the black lines given δ-location set as "•" and "⋆" where "⋆" is the true location. Denote $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_h$ the vertices of K'. Then we use a set $\Delta\mathbf{V}$ to store $\mathbf{v}_i - \mathbf{v}_j$ for any $\mathbf{v}_i$ and $\mathbf{v}_j$ from the vertices of K' as the equation below. In Figure 5b, for instance, the polygon "△ ⋯ △" denotes $\mathbf{v}_i - \mathbf{v}_1$ for all $\mathbf{v}_i$. Then $Conv(\Delta\mathbf{V})$ will be the sensitivity hull K of the δ-location set, as shown by the polygon with solid lines in Figure 5b.

$$K = Conv(\Delta\mathbf{V})$$
$$\Delta\mathbf{V} = \bigcup_{\mathbf{v}_1,\mathbf{v}_2 \in \text{ vertices of } K'} (\mathbf{v}_1 - \mathbf{v}_2)$$

Next we transform K to its isotropic position K_I. We sample $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_l$ uniformly from K. Then a matrix $\mathbf{T}$ can be derived by Equation (6). To verify if $\mathbf{T}$ is stable, we can derive another $\mathbf{T}'$. If the Frobenius norm $||\mathbf{T}' - \mathbf{T}||_F$ is small enough (e.g. $< 10^{-3}$), then we accept $\mathbf{T}$. Otherwise we repeat above process with larger l. In the end, $K_I = \mathbf{T}K$ is the isotropic position of K, as shown in Figure 5c.

Next a point $\mathbf{z}'$ can be uniformly sampled from K_I. We generate a random variable r from Gamma distribution $\Gamma(3, \epsilon^{-1})$.

[4] We refer readers to [2, 29] for a detailed study of isotropic position.



(a) (b) (c)

Figure 5: (a) Convex hull of $\Delta\mathbf{X}$. (b) Finding the sensitivity hull K. (c) Transform K to isotropic position K_I. Sample a point $\mathbf{z}'$.

Let $\mathbf{z}' = r\mathbf{z}'$. Then we transform the point $\mathbf{z}'$ to the original space by $\mathbf{z}' = \mathbf{T}^{-1}\mathbf{z}'$. The released location is $\mathbf{z} = \mathbf{x}^* + \mathbf{z}'$.

Algorithm 2 summarizes the process of PIM. Lines 5∼6 can be iterated until $\mathbf{T}$ is stable, whereas the computational complexity is not affected by the iterations because the number of samples is bounded by a constant (by Corollary 5.1).

Algorithm 2 Planar Isotropic Mechanism

```
Require: $\epsilon$, $\Delta\mathbf{X}$, $\mathbf{x}^*$
 1: $K' \leftarrow Conv(\Delta\mathbf{X})$;                              ▷ convex hull of $\Delta\mathbf{X}$
 2: $\Delta\mathbf{V} \longleftarrow \underset{\mathbf{v}_1,\mathbf{v}_2 \in \text{ vertices of } K'}{\cup} (\mathbf{v}_1 - \mathbf{v}_2)$;
 3: $K \leftarrow Conv(\Delta\mathbf{V})$;                              ▷ sensitivity hull
 4: Repeat lines 5,6 with larger $l$ if $\mathbf{T}$ is not stable:
 5:     Sample $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_l$ uniformly from $K$;
 6:     $\mathbf{T} \leftarrow \left(\frac{1}{l}\sum_{i=1}^{l}\mathbf{y}_i\mathbf{y}_i^T\right)^{-\frac{1}{2}}$;
 7: $K_I = \mathbf{T}K$;                                        ▷ isotropic transformation
 8: Uniformly sample $\mathbf{z}'$ from $K_I$;
 9: Sample $r \sim \Gamma(3, \epsilon^{-1})$;
10: return $\mathbf{z} = \mathbf{x}^* + r\mathbf{T}^{-1}\mathbf{z}'$;                       ▷ release z
```

Privacy and Performance Analysis. We now present the privacy property, complexity, and the error of PIM.

THEOREM 5.3. *Algorithm 2 is ϵ-differentially private on δ-location set $\Delta\mathbf{X}$.*

THEOREM 5.4. *Algorithm 2 takes $O(nlog(h) + h^2 log(h))$ time where n is the size of $\Delta\mathbf{X}$ and h is number of vertices on $Conv(\Delta\mathbf{X})$.*

THEOREM 5.5. *Algorithm 2 has error $O\left(\frac{1}{\epsilon}\sqrt{\text{AREA}(K)}\right)$ at most, which means it achieves the lower bound in Theorem 4.3.*

5.3 Location Inference

The inference of line 8 in Algorithm 1 is a general statement because inference methods depend on specific release algorithms. To implement the inference for PIM, we need to transform the location $\mathbf{s}_i$ and the released location $\mathbf{z}_t$ to the isotropic space of K_I. Then in Equation (1), the probability $Pr(\mathbf{z}_t|\mathbf{u}_t^* = \mathbf{s}_i)$ can be computed as follows. This completes the whole algorithm.

$$Pr(\mathbf{z}_t|\mathbf{u}_t^* = \mathbf{s}_i) = \frac{\epsilon^2}{2\text{AREA}(K_I)} exp(-\epsilon||\mathbf{z}_t' - \mathbf{s}_i'||_{K_I})$$
$$\mathbf{z}_t' = \mathbf{T}\mathbf{z};\ \mathbf{s}_i' = \mathbf{T}\mathbf{s}_i$$

6. EXPERIMENTAL EVALUATION

In this section we present experimental evaluation of our method. All algorithms were implemented in Matlab on a PC with 2.9 GHz Intel i7 CPU and 8 GB Memory.

Datasets. We used two real-world datasets.

I Geolife data. Geolife data [40] was collected from 182 users in a period of over three years. It recorded a wide range of users' outdoor movements, represented by a series of tuples containing latitude, longitude and timestamp. The trajectories were updated in a high frequency, e.g. every 1 ∼ 60 seconds. We extracted all the trajectories within the 3rd ring of Beijing to train the Markov model, with the map partitioned into cells of $0.34 \times 0.34\ km^2$.

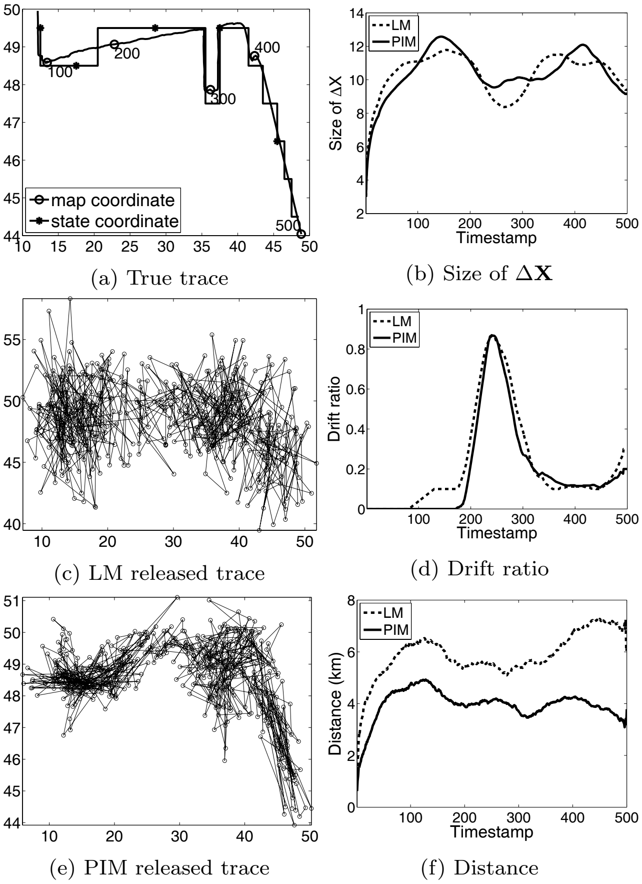
II Gowalla data. Gowalla data [7] contains $6,442,890$ check-in locations of $196,586$ users over the period of Feb. 2009 to Oct. 2010. We extracted all the check-ins in Los Angeles to train the Markov model, with the map partitioned into cells of $0.89 \times 0.89\ km^2$. Because check-ins were logged in a relatively low frequency, e.g. every 1 ∼ 50 minutes, we can examine the difference of the results from Gowalla and Geolife.

Metrics. We used the following metrics in our experiment, including two internal metrics: size of $\Delta\mathbf{X}$, drift ratio, and two sets of utility metrics: distance, precision and recall. We skip the runtime report because most locations were released within 0.3 second by PIM.

I Since our privacy definition is based on δ-location set $\Delta\mathbf{X}$, we evaluated the size of $\Delta\mathbf{X}$ to understand how $\Delta\mathbf{X}$ grows or changes.

II The definition of $\Delta\mathbf{X}$ and the potential limit of Markov model may cause the true location to fall outside $\Delta\mathbf{X}$ (drift). Thus we measured the drift ratio computed as the number of timestamps the true location is excluded in $\Delta\mathbf{X}$ over total number of timestamps.

III We measured the distance between the released location and the true location, which can be considered as a



(a) True trace

(b) Size of $\Delta\mathbf{X}$

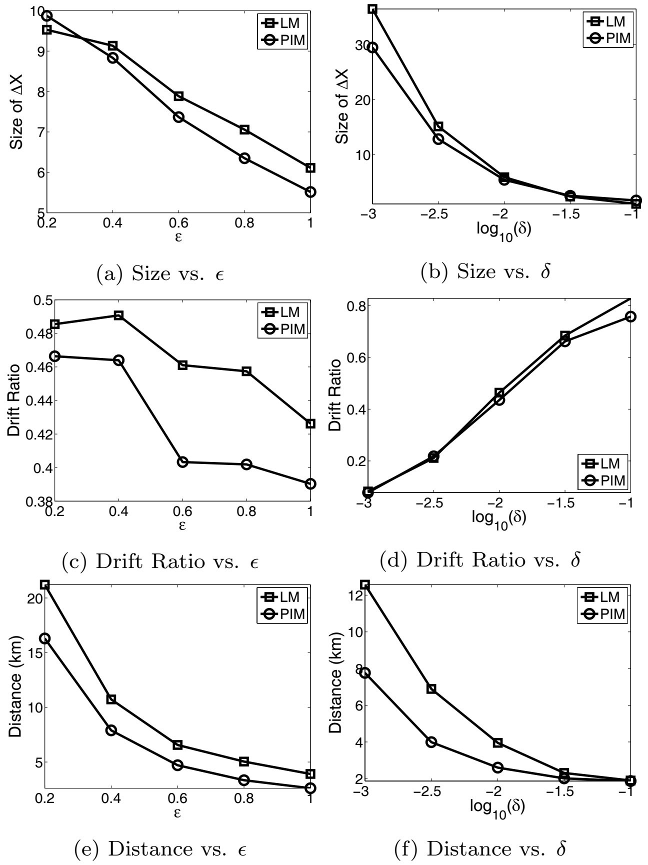
(c) LM released trace

(d) Drift ratio

(e) PIM released trace

(f) Distance

Figure 6: Performance over time: (a) The true (original) trace; (c)(e) Released traces; (b) Size of $\Delta\mathbf{X}$ over time; (d) Drift ratio over time; (f) Distance over time.



(a) Size vs. ϵ

(b) Size vs. δ

(c) Drift Ratio vs. ϵ

(d) Drift Ratio vs. δ

(e) Distance vs. ϵ

(f) Distance vs. δ

Figure 7: Impact of parameters on GeoLife data with popular $\mathbf{M}$: (a)(b) Impact of ϵ and δ on size of $\Delta\mathbf{X}$; (c)(d) Impact of ϵ and δ on drift ratio; (e)(f) Impact of ϵ and δ on distance.

general utility metric independent of specific location based applications.

IV We also run k nearest neighbor (kNN) queries using the released locations and report its precision and recall compared to the true kNN set using the original location. Suppose the true kNN set is R, the returned k'NN set (we set $k' \geq k$) is R', precision is defined as $|R \cap R'|/k'$, and recall is defined as $|R \cap R'|/k$.

6.1 Performance Over Time

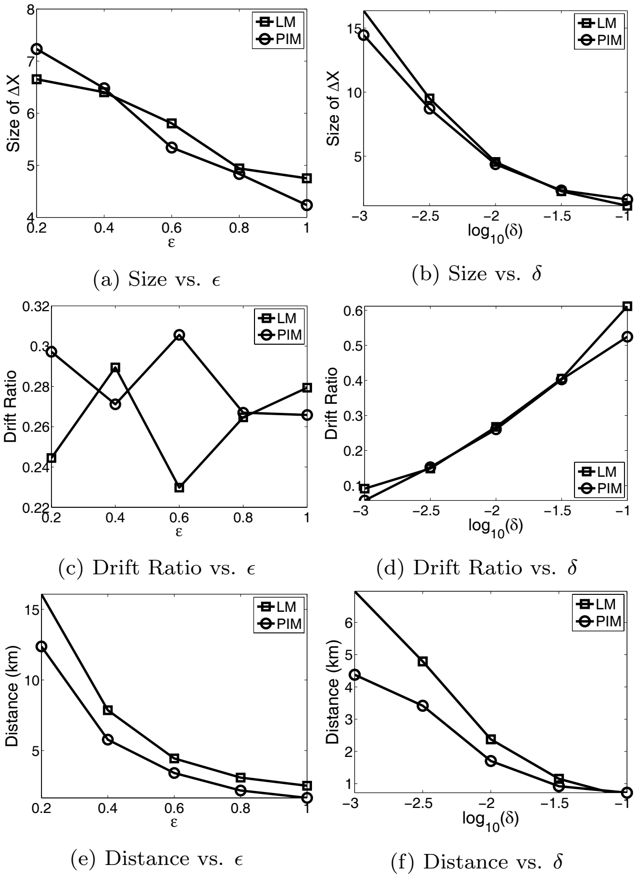
In order to show the performance of a release mechanism as a user moves over time, including how $\Delta\mathbf{X}$ changes, how often drift happens and how accurate is the perturbed location, we first run a set of experiments for a single test trajectory with popular $\mathbf{M}$ learned from all users. We selected a random test trajectory from Geolife dataset consisting of 500 timestamps. We tested both PIM and LM at each timestamp with $\epsilon = 1$ and $\delta = 0.01$. Each method was run 20 times and the average is reported. Figure 6a shows the original trajectory in map and state (grid) coordinates; Figures 6c and 6e show the released (perturbed) locations at each timestamp. We can see that the released locations of PIM is closer to the true location, compared with LM.

Size of $\Delta\mathbf{X}$. From Figure 6b we see that the size of $\Delta\mathbf{X}$ does not increase dramatically, instead it maintains at stable level after a few timestamps. The reason is that by selecting the δ-location set the inference mechanism only boost probabilities of locations in $\Delta\mathbf{X}$. Then the probabilities of other locations decay gradually. Thus a stable δ-location set can be maintained.

Drift Ratio. In Figure 6d, the peak of drift ratio happened in timestamp 200 $\sim$ 300. This can be explained by the fact that the true trajectory has a turning corner as in Figure 6a, and the transition probability of making this right turn is relatively small in the Markov model.

When a drift happens, we use surrogate for release mechanisms. Because the surrogate is the nearest cell to the true location in $\Delta\mathbf{X}$ and the release mechanism is based on the surrogate, the posterior probability of the surrogate will be boosted. Consequently, in the next timestamp the probability that $\Delta\mathbf{X}$ includes the previous true location rises. This "lagged catch-up" can be verified by Figures 6f, 6c and 6e.

Distance. The distance is reported in Figure 6f. We can see that PIM provided more accurate locations than LM for two reasons. First, because PIM is optimal, the posterior probability distribution is more accurate than LM. Second, with such distribution a better (Bayesian) inference can be



(a) Size vs. ϵ

(b) Size vs. δ

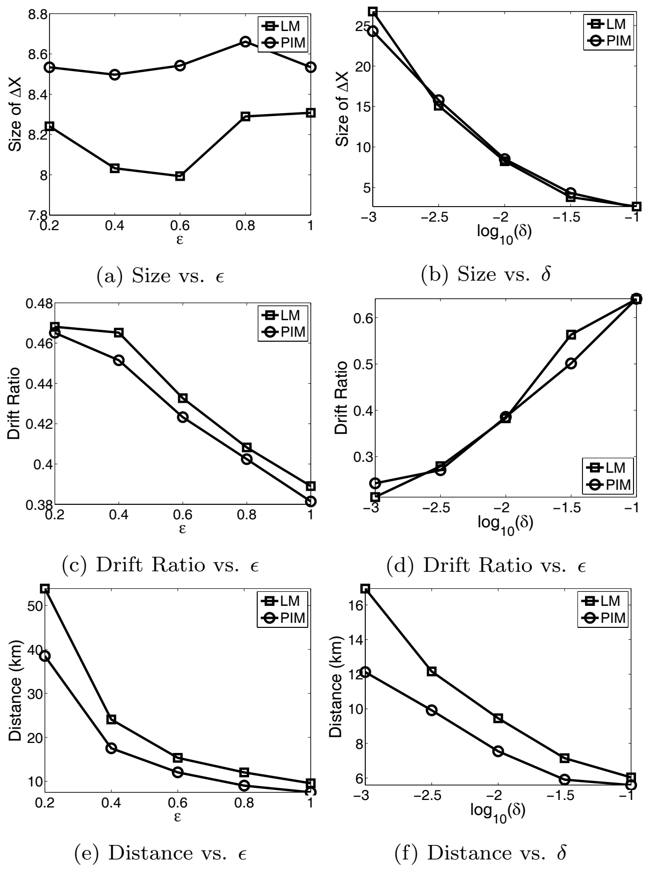
(c) Drift Ratio vs. ϵ

(d) Drift Ratio vs. δ

(e) Distance vs. ϵ

(f) Distance vs. δ

Figure 8: Impact of parameters on GeoLife data with personal $\mathbf{M}$: (a)(b) Impact of ϵ and δ on size of $\Delta\mathbf{X}$; (c)(d) Impact of ϵ and δ on drift ratio; (e)(f) Impact of ϵ and δ on distance.



(a) Size vs. ϵ

(b) Size vs. δ

(c) Drift Ratio vs. ϵ

(d) Drift Ratio vs. ϵ

(e) Distance vs. ϵ

(f) Distance vs. δ

Figure 9: Impact of parameters on Gowalla data with popular $\mathbf{M}$: (a)(b) Impact of ϵ and δ on size of $\Delta\mathbf{X}$; (c)(d) Impact of ϵ and δ on drift ratio; (e)(f) Impact of ϵ and δ on distance.

obtained, making $\Delta\mathbf{X}$ more accurate for the coming timestamp.

6.2 Impact of Parameters

Since the performance may vary for different trajectories, we chose 100 trajectories from 100 users, each of which has 500 timestamps, to evaluate the overall performance and the impact of parameters. The default values are $\epsilon = 1$ and $\delta = 0.01$ if not mentioned. The average performances for both datasets are reported in Figures 7 (on GeoLife data with popular $\mathbf{M}$), Figure 8 (on GeoLife data with personal $\mathbf{M}$) and Figure 9 (on Gowalla data with popular $\mathbf{M}$).

Size of $\Delta\mathbf{X}$ vs. ϵ. In Figures 7a and 8a (Geolife data), size of $\Delta\mathbf{X}$ shrinks with larger ϵ because the inference result is enhanced by big ϵ. On the other hand, impact of ϵ would be negligible in Gowalla data because one-step transition in Markov model has limited predictability (check-ins are not frequent), as in Figure 9a.

Size of $\Delta\mathbf{X}$ vs. δ. Size of $\Delta\mathbf{X}$ is mainly determined by δ as shown in Figures 7b, 8b and 9b. Note that LM and PIM have similar size of $\Delta\mathbf{X}$, meaning the true location is hidden in the similar size of candidates. When δ grows, size of $\Delta\mathbf{X}$ reduces dramatically because more improbable locations are truncated. However, δ cannot be too large because it preserves nearly no privacy if size of $\Delta\mathbf{X}$ is close to 1. Thus we use $\delta = 0.01$ by default, which guarantees the sizes of $\Delta\mathbf{X}$ are larger than 4 in the three settings.

Drift Ratio vs. ϵ. Figures 7c and 9c show that drift ratio declines with larger ϵ, which is easy to understand because larger ϵ provides more accurate release. However, the impact of ϵ is not obvious in Figure 8c. The reason is that the size of $\Delta\mathbf{X}$ is already small as in Figure 8b, hence the increase of ϵ does not help much in improving the accuracy of the inference.

Drift Ratio vs. δ. Figures 7d, 8d and 9d show that drift ratio rises when δ increases due to reduced $\Delta\mathbf{X}$, and PIM is slightly better than LM. However, due to the phenomenon of "lagged catch-up", we will see next that the accuracy of the released locations was still improved with increasing δ.

Distance vs. ϵ. Figures 7e, 8e and 9e show the distance with varying ϵ. We can see that PIM performed better than LM. In Gowalla data, because check-in locations are far away from each other, the distance is larger than Geolife.

Distance vs. δ. Because bigger δ will result in less candidates in δ-location set, the distance declines when δ increases. Figures 7f, 8f and 7f show that PIM achieves better accuracy than LM. However, from $10^{-1.5}$ to 10^{-1}, the improvement on distance is very small while privacy guarantee drops significantly as in Figures 7b, 8b and 9b. Especially in

Figure 8b, size of $\Delta\mathbf{X}$ is 1 when $\delta = 0.1$. Therefore, choosing a high value of δ (like $\delta > 0.03$) does not provide the best trade-off of privacy and utility.

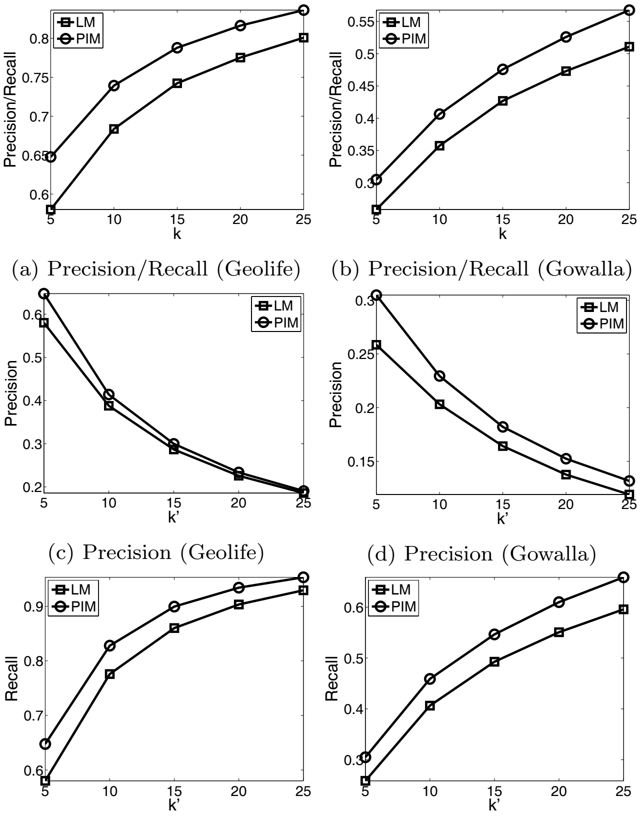
Impact of Markov model. Comparing Figures 7 on popular $\mathbf{M}$ and Figure 8 on personal $\mathbf{M}$, we can see the impact of different Markov model. With more accurate (personal) model, better utility can be achieved, including smaller size of $\Delta\mathbf{X}$, lower drift ratio and less distance. However, the same privacy level (ϵ-differential privacy) is maintained (on different $\Delta\mathbf{X}$) regardless of $\mathbf{M}$.

6.3 Utility for Location Based Queries

To demonstrate the utility of released locations, we also measured the precision and recall of kNN queries at each of the 500 timestamps in the 100 trajectories with popular $\mathbf{M}$. The average results of kNN from original locations and k'NN from released locations are reported in Figure 10 with $\epsilon = 1$ and $\delta = 0.01$.

In Figures 10a and 10b, we show the precision and recall with $k = k'$. Note that in this case precision is equal to recall. We can see that when k grows precision and recall also increase because the nearest neighbors have to be found in larger areas. PIM is consistently better than LM.

Next we fixed $k = 5$ and varied k'. Figures 10c and 10d show the precision drops when k' rises because of a larger returned set. On the other hand, Figures 10e and 10f indicate recall increases with large k'. Overall, PIM has better precision and recall than LM.



(a) Precision/Recall (Geolife) (b) Precision/Recall (Gowalla)

(c) Precision (Geolife) (d) Precision (Gowalla)

(e) Recall (Geolife) (f) Recall (Gowalla)

Figure 10: kNN results: (a)(b) precision and recall under $k = k'$; (c)(d) precision vs. k'; (e)(f) recall vs. k'.

7. RELATED WORKS

7.1 Location Privacy

There is a rich set of literature related to location privacy. A few recent books and surveys [23, 15] provide an up-to-date review of Location Privacy Preserving Mechanisms (LPPMs).

LPPMs generally use obfuscation methods, such as spatial cloaking, cell merging, location precision reduction or dummy cells, to achieve anonymity based privacy or uncertainty based privacy. However, anonymity or ad hoc uncertainty based techniques do not always provide sufficient privacy protection [22, 37]. Most of them do not consider the temporal correlations between locations and are subject to various inference attacks. The recent work [1] proposed a notion of geo-indistinguishability which extends differential privacy. However, a fundamental difference is that neighboring pairs or secrets are defined based on a radius and it does not consider the temporal correlations of multiple locations.

Several works use Markov models for modeling users' mobility and inferring user locations or trajectories [25, 34]. [17] proposed an insightful technique with a provable privacy guarantee to filter a user context stream even if the adversaries are powerful and have knowledge about the temporal correlations but it used suppression instead of perturbation. [37] investigated the question of how to formally quantify the privacy of existing LPPMs and assumed that an adversary can model users' mobility using a Markov chain learned from a population.

7.2 Differential Privacy

Several variants or generalizations of differential privacy have been studied, as discussed in Section 3.4. However, applying differential privacy for location protection has not been investigated in depth. Several recent works have applied differential privacy to publish *aggregate* information from a large volume of location, trajectory or spatiotemporal data (e.g. [6, 33, 12, 24]). Our contribution is to extend differential privacy in a new setting of continual location sharing of only one user whose locations are temporally correlated.

Optimal query answering under differential privacy has been studied recently. Hardt and Talwar [18] studied the theoretical lower bound for any differentially private mechanisms and proposed K-norm mechanism. Bhaskara et al [4] studied another K-norm based method to project the sensitivity hull onto orthogonal subspaces. Nikolov et al [30] also improved the efficiency of K-norm mechanism by finding the minimal enclosing ellipsoid to release multivariate Gaussian noises. So far the best utility of existing mechanisms can be $log(d)$ approximately optimal. We extended the K-norm mechanism to location data by examining the two-dimensional sensitivity hull and designing its isotropic transformation so that optimal utility can be achieved.

8. CONCLUSION AND FUTURE WORK

In this paper we proposed δ-location set based differential privacy to protect a user's true location at every timestamp under temporal correlations. We generalized the notion of "neighboring databases" to δ-location set for the new setting and extended the well known ℓ_1-norm sensitivity to sensitivity hull in order to capture the geometric meaning of sensitivity. Then with sensitivity hull we derived the lower bound of δ-location set based differential privacy. To achieve the lower bound, we designed the optimal planar isotropic mechanism to release differentially private locations with significantly high efficiency and utility.

The framework of δ-location set based differential privacy can work with any mobility models (besides Markov chain). As a future work direction, we are interested in instantiating it with different and more advanced mobility models and studying the impact.

9. ACKNOWLEDGEMENT

This research is supported by NSF under grant No. 1117763 and the AFOSR DDDAS program under grant FA9550-12-1-0240. We thank the anonymous reviewers for their valuable comments that helped improve the final version of this paper.

10. REFERENCES

[1] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. CCS '13, pages 901–914. ACM, 2013.

[2] G. Apostolos. Notes on isotropic convex bodies. 2003.

[3] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *Pervasive Computing, IEEE*, 2(1):46–55, 2003.

[4] A. Bhaskara, D. Dadush, R. Krishnaswamy, and K. Talwar. Unconditional differentially private mechanisms for linear queries. STOC '12, New York, NY, USA, 2012.

[5] K. Chatzikokolakis, M. Andrés, N. Bordenabe, and C. Palamidessi. Broadening the scope of differential privacy using metrics. Lecture Notes in Computer Science, pages 82–102. Springer Berlin Heidelberg, 2013.

[6] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou. Differentially private transit data publication: a case study on the montreal transportation system. KDD '12, pages 213–221, New York, NY, USA, 2012. ACM.

[7] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. KDD '11, pages 1082–1090, New York, NY, USA, 2011.

[8] A. Dey, J. Hightower, E. de Lara, and N. Davies. Location-based services. *Pervasive Computing, IEEE*, 9(1):11–12, 2010.

[9] C. Dwork. Differential privacy. In *Automata, languages and programming*, pages 1–12. Springer, 2006.

[10] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. Proceedings of the 3rd Theory of Cryptography Conference, 2006.

[11] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. STOC '10, pages 715–724, New York, NY, USA, 2010. ACM.

[12] L. Fan, L. Xiong, and V. S. Sunderam. Differentially private multi-dimensional time series release for traffic monitoring. In *DBSec*, pages 33–48, 2013.

[13] C. Fang and E.-C. Chang. Differential privacy with δ-neighbourhood for spatial and dynamic datasets. ASIA CCS '14, pages 159–170, New York, NY, USA, 2014. ACM.

[14] B. Gedik and L. Liu. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *Mobile Computing, IEEE Transactions on*, 7(1):1–18, 2008.

[15] G. Ghinita. *Privacy for Location-Based Services.* Synthesis Lectures on Information Security, Privacy, and Tru. Morgan & Claypool, 2013.

[16] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

[17] M. Götz, S. Nath, and J. Gehrke. Maskit: Privately releasing user context streams for personalized mobile applications. SIGMOD '12, New York, NY, USA, 2012.

[18] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, pages 705–714. ACM, 2010.

[19] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. SIGMOD '14, pages 1447–1458, New York, NY, USA, 2014. ACM.

[20] I. A. Junglas and R. T. Watson. Location-based services. *Communications of the ACM*, 51(3):65–69, 2008.

[21] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204. ACM, 2011.

[22] J. Krumm. Inference attacks on location tracks. PERVASIVE'07, pages 127–143, Berlin, Heidelberg, 2007. Springer-Verlag.

[23] J. Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.

[24] H. Li, L. Xiong, and X. Jiang. Differentially private synthesization of multi-dimensional data using copula functions. EDBT'14, pages 475–486, 2014.

[25] L. Liao, D. J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artif. Intell.*, 171(5-6):311–331, Apr. 2007.

[26] L. Lovász and S. Vempala. Hit-and-run from a corner. STOC '04, pages 310–314, New York, NY, USA, 2004. ACM.

[27] L. Lovász and S. Vempala. Simulated annealing in convex bodies and an o*(n4) volume algorithm. *J. Comput. Syst. Sci.*, 72(2):392–417, Mar. 2006.

[28] McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD '09*, pages 19–30, New York, NY, USA, 2009. ACM.

[29] V. Milman and A. Pajor. Isotropic position and inertia ellipsoids and zonoids of the unit ball of a normed n-dimensional space. Lecture Notes in Mathematics, pages 64–104. Springer Berlin Heidelberg, 1989.

[30] A. Nikolov, K. Talwar, and L. Zhang. The geometry of differential privacy: The sparse and approximate cases. ACM STOC '13, pages 351–360, NY, USA, 2013.

[31] J. O'Rourke. *Computational Geometry in C.* Cambridge University Press, New York, NY, USA, 2nd edition, 1998.

[32] S. Papadopoulos, S. Bakiras, and D. Papadias. Nearest neighbor search with strong location privacy. *Proceedings of the VLDB Endowment*, 3(1-2):619–629, 2010.

[33] W. H. Qardaji, W. Yang, and N. Li. Differentially private grids for geospatial data. In *ICDE*, pages 757–768, 2013.

[34] S. Qiao, C. Tang, H. Jin, T. Long, S. Dai, Y. Ku, and M. Chau. Putmode: prediction of uncertain trajectories in moving objects databases. *Applied Intelligence*, 33(3):370–386, Dec. 2010.

[35] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: output perturbation for queries with joins. PODS '09, pages 107–116, New York, NY, USA, 2009. ACM.

[36] M. Rudelson. Random vectors in the isotropic position. *J. Funct. Anal*, pages 60–72, 1999.

[37] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux. Quantifying location privacy. IEEE SP '11, pages 247–262, Washington, DC, USA, 2011.

[38] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[39] S. Vempala. Geometric random walks: a survey. *Combinatorial and Computational Geometry*, pages 573–612, 2005.

[40] Y. Zheng, X. Xie, and W.-Y. Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.