# `PrivLBS`: Local Differential Privacy for Location-Based Services with Staircase Randomized Response

## ABSTRACT

Location-based services (LBS) have been significantly developed and widely deployed in mobile devices. It is also well-known that LBS applications may result in severe privacy concerns by collecting sensitive locations. A strong privacy model "local differential privacy" (LDP) has been recently deployed in many different applications (e.g., Google RAPPOR, iOS, and Microsoft Telemetry) but not effective for LBS applications due to the low utility of existing LDP mechanisms. To address such deficiency, we propose the first LDP framework for a variety of location-based services (namely "`PrivLBS`"), which privately collects and analyzes user locations with high utility. Specifically, we design a novel randomization mechanism "Staircase Randomized Response" (`SRR`) and extend the empirical estimation to significantly boost the utility for `PrivLBS` in different LBS applications (e.g., traffic density estimation, and k nearest POIs). We have conducted extensive experiments on four real LBS datasets by benchmarking with other LDP schemes in practical applications. The experimental results demonstrate that `PrivLBS` significantly outperforms them.

## 1 INTRODUCTION

Location-based services (LBS) are widely deployed in mobile devices to provide useful and timely location-based information to users. For instance, WeatherBug provides weather information based on users' regions; Google Map not only navigates the routes with real-time traffic conditions but also responds to queries such as nearby restaurants or gas stations; Waze is similar to Google Map but actively collects extra information (e.g., accidents, road construction, and police) from users and shares them to other users.

All of these LBS applications highly rely on the personal locations collected from millions of users. Such locations should be protected, e.g., per the General Data Protection Regulation (GDPR) since visited places can be sensitive (e.g., hospital) or used to re-identify users from the data (e.g., a sequence of them can be unique). To mitigate such risks, location anonymization models [8] were first proposed to achieve $k$-anonymity via location generalization. However, $k$-anonymity can only provide a weak privacy guarantee (e.g., vulnerable to the background knowledge attacks [43]). As a rigorous privacy model against arbitrary prior knowledge known to the adversaries, differential privacy (DP) has been extensively studied to address location privacy risks (e.g., [28]). It ensures that adding or removing any user's location or trajectory still generates indistinguishable results. For instance, AdaTrace [28], a differentially private location trace synthesizer was proposed to ensure provable privacy, deterministic attack resilience, and strong utility. However, in the DP scenario setting, it requires an authorized data center to collect user's location. Unfortunately, in the 2011 Microsoft survey, 87% of participants reported that they care about who accesses their location information; over 78% workers of Amazon interviewed in 2014 still do not trust these LBS applications on collecting their locations and believed apps accessing to their locations can pose significant privacy threats [12]. Thus, it is highly desirable to explore private location collection by an *untrusted* server.

Recently, *local differential privacy* (LDP) techniques [6, 16, 25, 58] have been successfully deployed in industry (e.g., Google [25], Apple [1], and Microsoft [18]) to privately aggregate locally perturbed data. It provides stronger privacy against attackers with arbitrary background knowledge (not only the downstream analysts but also the data aggregator can be *untrusted*). To date, existing LDP schemes such as RAPPOR and generalized randomized response have been extended to privately aggregate different types of data, e.g., set-valued data [18], numerical data [40], and graphs [53]. However, existing LDP schemes are not very effective on private location data collection and analysis due to either limited utility or relaxed privacy protection. To our best knowledge, only [37, 67] applied existing LDP schemes to locations but the utility is still poor. Moreover, `PLDP` [13] relaxed LDP to personalized LDP (not every user can be protected with $\epsilon$-LDP) in the location collection for spatial density estimation.

Furthermore, some other privacy-enhancing techniques [4, 13] privately collect locations for LBS that provides services to individual users (e.g., GPS navigation [68], and nearest point-of-interest (POI) search [39]) *without a trusted server.* For instance, geo-indistinguishability [4] adds Laplace noise to the user's location for ensuring privacy in LBS. However, it cannot strictly satisfy LDP (the locations are indistinguishable only within a radius), and the Laplace mechanism has been shown to be worse than randomized response for local perturbation [63].

To address such limitations, we propose the first strict LDP framework (namely, "`PrivLBS`") to support a variety of LBS applications. First, we design a novel LDP mechanism "*staircase randomized response* (`SRR`)" and revise the empirical estimation to privately aggregate locations with significantly improved utility and strictly satisfied $\epsilon$-LDP. Second, different from all existing works [4, 13, 37, 67], we design additional components (e.g., private matching [39] and private information retrieval [26]) into `PrivLBS` to ensure $\epsilon$-LDP for a variety of LBS applications such as $k$ nearest neighbors search [66], origin-destination analysis [7] and traffic-aware GPS navigation [68], which may collect user trajectories or perform individual services with the aggregated locations/trajectories.

The utility of `PrivLBS` is significantly enhanced by the proposed new `SRR` mechanism and estimation method. Specifically, `SRR` perturbs input locations with *staircase-shape probabilities* for different possible output locations. The probability of perturbing any input $x$ in the domain $\mathcal{D}$ to each possible location $y \in \mathcal{D}$ is optimally pre-computed. Then, users can locally perturb their locations with the optimal probabilities. Different from relaxed privacy notions (e.g., `PLDP` and geo-indistinguishability), every user is still strictly protected by $\epsilon$-LDP. At the server end (data aggregator), we extend an empirical estimation [36] to further improve the utility

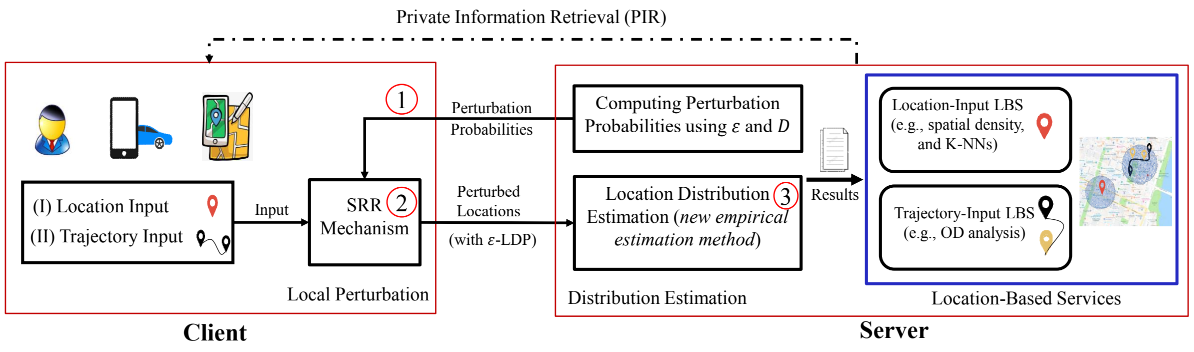

Private Information Retrieval (PIR)
(I) Location Input
(II) Trajectory Input
Input
SRR Mechanism
Perturbation Probabilities
Perturbed Locations (with ε-LDP)
Local Perturbation
Client
Computing Perturbation Probabilities using ε and D
Location Distribution Estimation (new empirical estimation method)
Results
Distribution Estimation
Location-Input LBS (e.g., spatial density, and K-NNs)
Trajectory-Input LBS (e.g., OD analysis)
Location-Based Services
Server

**Figure 1: The PrivLBS framework**

for the SRR mechanism without extra privacy leakage [23]. Thus, the major contributions of this paper are summarized as below:

- To our best knowledge, we design the first LDP mechanism (SRR) to make the strong privacy notion LDP practical (with high utility) for many LBS applications.
- In SRR, we propose a novel hierarchical encoding scheme and relevant algorithms to derive the optimal perturbation probabilities independent of the input data. We also extend the empirical estimation method to further improve utility.
- We design and integrate components in PrivLBS to realize SRR in a series of LBS applications with high accuracy, which may collect locations (e.g., frequency estimation [51]) or trajectories (e.g., origin-destination analysis [7], and traffic-aware GPS navigation [68]).
- Besides theoretical studies on the privacy and utility, we conduct extensive experiments on four real LBS datasets, and benchmark with other LDP schemes, e.g., Generalized Randomized Response (GRR)[58], Local Hash (OLH−H)[58], PLDP (based on Unary Encoding)[13], and Hadamard Response (HR) [36]. PrivLBS greatly outperforms them in almost all the scenarios.

The remainder of this paper is organized as follows. Section 2 introduces some preliminaries. Section 3 illustrates the SRR mechanism, and Section 4 extends SRR to collect trajectories. Section 5 gives related discussions. Section 6 shows the experimental results. Section 7 and 8 discuss the literature and conclude the paper.

## 2 PRELIMINARIES

### 2.1 LBS Applications

We first categorize two different types of LBS applications:

**Location-Input LBS**: LBS App collects a single location from each user, and the untrusted server privately analyzes the aggregated data, e.g., identifying the top crowded areas [55], and spatial density estimation [13]. In some LBS applications, the clients may query the analysis results from the server (e.g., location-based advertising [17], and $k$ nearest point of interests (POIs) for each user [66]).

**Trajectory-Input LBS**: LBS App collects multiple sequential locations (trajectory) from each user, and the untrusted server privately analyzes the aggregated data, e.g., aggregating users' origin-destination (OD) pairs to learn the traffic flow [7, 56]. Similarly, users may query the analysis results computed by the server, e.g., users query the real-time traffic for the GPS navigation [56].

### 2.2 Privacy Model

Users in PrivLBS will locally randomize their location(s) [9] with algorithm $\mathcal{A}$ and send the noisy results to the untrusted server. After local perturbation, all the input locations can be indistinguishable [25]. The privacy notion is formally defined as below:

*Definition 2.1 ($\epsilon$-LDP).* A randomization algorithm $\mathcal{A}$ satisfies $\epsilon$-Local Differential Privacy, if and only if for any pair of input locations $x, x' \in \mathcal{D}$, and for any perturbed output $y \in range(\mathcal{A})$ sent to the untrusted server, we have: $Pr[\mathcal{A}(x) = y] \leq e^{\epsilon} \cdot Pr[\mathcal{A}(x') = y]$.

After each user locally perturbs its data, LDP can be ensured for all the input locations [16, 25, 58], where the privacy bound $\epsilon$ reflects the degree of indistinguishability. The untrusted server will aggregate and analyze the noisy data with estimation methods.

### 2.3 PrivLBS Framework

As shown in Figure 1, we design three major components in PrivLBS: perturbation (by client), analysis (by server), and private retrieval (by both client and server only when the user needs to privately query the analysis results, e.g., traffic-aware GPS navigation):

(1) **Perturbation (client)**: Each user's location data (location or trajectory) is locally perturbed by the client with $\epsilon$-LDP. SRR optimizes the utility after hierarchically encoding the location domain $\mathcal{D}$. Encoding and optimal perturbation probabilities are pre-computed by the server (only based on $\epsilon$ and $\mathcal{D}$) to ensure $\epsilon$-LDP. See details in Section 3.2.

(2) **Analysis (server)**: Before the perturbation, the server shares the pre-computed perturbation probabilities with all the clients. After receiving the perturbed user locations, the server estimates the *location distribution* with a revised empirical estimation method. Then, the server loads such results into specific LBS (along with the required components) to privately derive the analysis result. See details in Section 3.

(3) **Private Retrieval (only for LBS with client queries)**: It is an optional component of PrivLBS. If requested in specific LBS (with client queries), PrivLBS first provides the server-side LBS analysis (e.g., estimating the overall traffic density) with LDP guarantees. At the client end, each user privately

queries his/her result (e.g., nearby traffic) from the analysis results at the server side. This can be achieved with a private information retrieval (PIR) protocol [5]. With the PIR for client queries, server does not know which result is delivered to which user, and each user does not know other users' results either. [1]

**User Requirements**. PrivLBS can be deployed as a privacy preserving API in each LBS App. Users only need to periodically update the privacy bound $\epsilon$ with the server. In each LBS, users only need to locally perturb their location(s) with the pre-computed perturbation probabilities, and send the perturbed result to the server. The integrated PIR [26] also requires very minor computation and communication overheads without affecting the LDP guarantee (see the discussion in Section 5).

**LDP Protection**. Similar to existing LDP models [25, 58], PrivLBS ensures strong privacy against inferences on users' local data based on arbitrary background knowledge, which is orthogonal to mitigating other types of risks (e.g., encryption [38] and defenses against side-channel attacks [15]). Thus, PrivLBS can be integrated with them to further improve security and privacy if necessary.

## 3 PRIVLBS FOR LOCATION-INPUT LBS

In this section, we design the SRR mechanism to privately collect a location from each user for analysis (standard LDP setting [13, 58]).

### 3.1 Staircase Randomized Response

We first review a family of LDP mechanisms. Randomized Response (RR) based schemes, such as generalized randomized response (GRR) [60] and unary encoding (UE) [58], satisfy $\epsilon$-LDP. For instance, in GRR, given the domain size $d = |\mathcal{D}|$, privacy bound $\epsilon$ and input $x \in \mathcal{D}$, the true value has a higher probability to be sampled (output $y$). The following perturbation probabilities $q(y|x)$ ensure $\epsilon$-LDP.

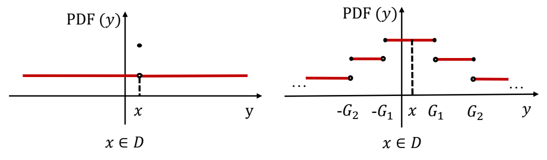
$$\texttt{GRR}: \; q(y|x) = \begin{cases} \frac{e^{\epsilon}}{d+e^{\epsilon}-1}, & \text{if } y = x \\ \frac{1}{d+e^{\epsilon}-1}, & \text{otherwise} \end{cases} \tag{1}$$

Also, Hadamard Response (HR) [36] has a subset domain for each value $x$ and a higher probability for values in the subset to be sampled. Then, the remaining values in the domain are sampled with a smaller probability. However, *only two different perturbation probabilities* are defined in the existing LDP mechanisms (e.g., GRR [60], UE [58], and HR [36]), not sufficiently fine-grained to optimize the utility (since the perturbation probabilities simply treat all the other output locations in the domain equally).

Thus, we propose a novel Staircase Randomized Response (SRR) mechanism for locations and LBS. Intuitively, if the probabilities for locations that are closer to the input location $x$ can be higher, it is more possible for users the query results of the LBS are the same. To this end, SRR will first consider the location distances to the input location $x$. Then, a set of fine-grained probabilities should be pre-computed for all the possible output locations $y \in \mathcal{D}$.

[1]If we directly design a cryptographic protocol for each LBS, it involves location data encryption by the client, and the server should extend each LBS algorithm over encrypted data to a cryptographic protocol, which would result in extremely high computation and communication overheads. Compared to that, PIR establishes a secure channel for privately retrieving the results, which can be independent of the LBS algorithms and extensible to all the analyses on noisy data by the server.

When pre-computing the these probabilities, there are several issues in practice. For instance, for each input location $x$, if we compute the probability $q(y|x)$ for each possible output $y \in \mathcal{D}$, the number of probabilities is the domain size $d$. Then, $\forall x \in \mathcal{D}$, there are $d$ probabilities for each location $x$ and $d \times d$ different probabilities for all the locations in the domain. Thus, there are $d^2$ unknown probabilities to be determined, which makes it time-consuming to derive the optimal probabilities [30] and not extensible if the domain is updated. Second, general objective function (e.g., the variance) to optimize the perturbation probabilities is dependent on the unknown true frequencies. To address this, output locations can be partitioned into different groups in terms of their distances to $x$ (*the probabilities of all the output locations in the same group could be identical*), and we can derive the perturbation probability for each group to boost the utility while globally satisfying $\epsilon$-LDP.



(a) GRR mechanism [60] (b) SRR mechanism (for PrivLBS)

**Figure 2: Probability density function (PDF) for GRR and SRR**

The probability density functions (PDFs) of GRR (w.l.o.g.) and SRR are illustrated in Figure 2. In GRR, the probability that outputs the true value (the point in the 2(a)) is higher than other values. On the contrary, since SRR discretizes the perturbation probabilities for all the grouped possible output locations, the PDF of SRR has a similar shape to the staircase mechanism in differential privacy [29] which also has a staircase-shape PDF for different groups to satisfy $\epsilon$-DP. Motivated by that, we name our new randomization mechanism as the "Staircase Randomized Response" (SRR) in local differential privacy. We formally define the perturbation probabilities from input $x$ to all the output locations as follows.

Given the domain $\mathcal{D}$, for any input $x \in \mathcal{D}$, all the possible output locations can be partitioned into $m$ groups $G_1(x), ..., G_m(x)$ based on their distances to $x$.[2] Notice that, the partitioning $G_j(x)$ is dependent on the input location $x$. For each input location $x$, all its $m$ location groups and the perturbation probabilities (for perturbing $x$ to any output location $y$) will be efficiently computed as:

$$\texttt{SRR}: \forall x \in \mathcal{D}, q(y|x) = \begin{cases} \alpha_1(x), & \text{if } y \in G_1(x) \\ \vdots & \vdots \quad \vdots \\ \alpha_m(x), & \text{if } y \in G_m(x) \end{cases} \tag{2}$$

where $\alpha_1(x), ..., \alpha_m(x)$ are the distance-based perturbation probabilities for locations in $m$ different groups perturbed from $x \in \mathcal{D}$, and the gap between the perturbation probabilities in every adjacent groups is the same ("Staircase-Shape PDF") in $\alpha_1(x), ..., \alpha_m(x)$. Also, the sum of all the perturbation probabilities for each input location $x$ should satisfy: $\sum_{j\in[1,m]} \sum_{y \in G_j(x)} q(y|x) = 1$. The details for computing the probabilities will be given in Section 3.3. SRR generates more accurate locally perturbed locations than the state-of-the-art LDP mechanisms with only two perturbation probabilities (e.g., GRR [60] and HR [36]), as validated in Section 6.

[2]W.l.o.g., the distances from $x$ to locations in $G_j(x)$ are farther if $j$ is larger. The closest group is $G_1(x)$ whereas the farthest group is $G_m(x)$.

## 3.2 Data Encoding and Domain Partitioning

**Hierarchical Location Encoding**. To encode the location data, we use a hierarchical encoding scheme based on the Bing Map Tiles System [2], which recursively partitions geo-coordinates into 4 blocks, and indexes all the locations to reach the desired resolution [28]. Then, the locations are encoded into bit strings by hierarchically concatenating the indices of all the levels for every specific location. Figure 3 illustrates an example for the encoding. Specifically, starting from the root node, at each level $h$, the 4 children of each node (four sub-blocks) can be encoded by 00, 01, 10, 11 (2-bit), and thus form $4^h$ blocks for indexing locations. Then, we can derive the encoded bit string by concatenating the bits from the first level to the leaf node level. For all the locations on the earth, $h$ can be as large as 23 (46 bits for a location) to index each 4.7m×4.7m region. As a result, all the locations can be encoded with the same length of bits if the same precision ($h$) is applied to all the locations.

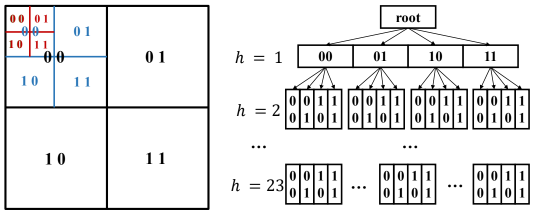


**Figure 3: Hierarchical encoding for locations**

*Example 3.1 (Encoding for "New York").* The coordinates of the center of "New York" are (40.730610, −73.935242). Given $h = 23$, the location is encoded as "e1147b6afff" (hex of the bit string).

**Location Groups**. With hierarchical encoding for the location domain $\mathcal{D}$, the distance between any two locations $x, x' \in \mathcal{D}$ can be directly measured by the longest common prefixes (LCP) of their encoded bit strings. Then, given a location $x$ and any of its output groups $G_j(x), j \in [1, m]$, we define the LCP of the group.

*Definition 3.2 (Group LCP).* Given an input location $x$ and any of its groups $G_j(x), j \in [1, m]$, the group LCP (aka. GLCP) is the shortest LCP between the input location $x$ and $\forall y \in G_j(x)$. The length of GLCP for group $G_j(x)$ is denoted as $\beta_j(x)$.

Thus, the distance between the input location $x$ and each location group $G_j(x), j \in [1, m]$ can be measured by the length of its GLCP $\beta_j(x)$: the larger, the closer. Then, we can partition all the output locations into groups using the GLCP lengths. In each group $G_j(x)$, all the locations share a prefix with at least $\beta_j(x)$ bits with location $x$ (applying such rule for partitioning could reduce the complexity of partitioning to $O(d)$ though not optimal). For the group with a longer GLCP shared with the input location $x$, higher probabilities will be assigned to them (for perturbing $x$).[3]

**Location Partitioning**. We next partition the locations into $m$ groups for each input $x \in \mathcal{D}$, and assign the same perturbation probability to all the locations in the same group. Specifically, for $m$ groups, we define a GLCP length vector $\{\beta_1(x), \dots, \beta_m(x)\}$. All the encoded locations in group $G_j(x), 1 \le j \le m$ share *at least* $\beta_j(x)$-bit prefix with $x$. Then, $\beta_1(x) > \beta_2(x) > \cdots > \beta_m(x)$ since $G_1(x)$ is the closest group to the input location $x$.

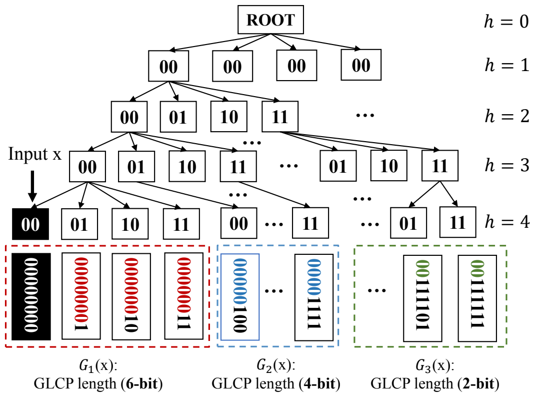


**Figure 4: Example of location domain partitioning**

Figure 4 shows an example for partitioning the location domain. Given the input location $x$, all the locations are partitioned into three groups with the GLCP lengths $\{\beta_1(x) = 6, \beta_2(x) = 4, \beta_3(x) = 2\}$ where $m = 3$. In $G_1(x)$, $G_2(x)$ and $G_3(x)$, all the locations share at least 6-bit, 4-bit and 2-bit prefix with $x$, respectively.

Thus, given any GLCP length vector $\beta_1(x), \dots, \beta_m(x)$, the $m$ groups $G_1(x), \dots, G_m(x)$ for the input location $x$ can be efficiently generated with complexity $O(d)$. Then, denoting the LCP between input $x$ and output $y$ as $LCP(x, y)$, the optimal $\{\beta_1(x), \dots, \beta_m(x)\}$ and the $m$ groups that maximizes $\sum_{\forall y \in \mathcal{D}} LCP(x, y)$ can be derived.

More specifically, if $m$ is not large, we can traverse all the GLCP lengths $\{\beta_1(x), \dots, \beta_m(x)\}$ where $\beta_1(x) > \beta_2(x) > \cdots > \beta_m(x)$ to find the optimal result. Otherwise, the server can apply a meta-heuristic algorithm (e.g., simulated annealing [42]) to derive a near-optimal $\{\beta_1(x), \dots, \beta_m(x)\}$ for partitioning. Next, the location domain $\mathcal{D}$ can be efficiently partitioned by the optimal $\{\beta_1(x), \dots, \beta_m(x)\}$. First, locations sharing a $\beta_1(x)$-bit or longer prefix with $x$ will be assigned to $G_1(x)$; second, the locations sharing a prefix (length between $\beta_2(x)$-bit and $(\beta_1(x) - 1)$-bit) with $x$ will be assigned to $G_2(x)$; repeat the above until $G_m(x)$ is formed.

**Offline Computation**. Since the optimization and partitioning are solely based on the domain $\mathcal{D}$, they can be executed offline and periodically updated with $\mathcal{D}$ by the server in `PrivLBS`. In general, the location domain is stored in the server of company and released as public knowledge for users (e.g., Google Maps) and the company will take about several days to update the domains since the company has to verify locations before making the changes available to the public. Then, for each $x \in \mathcal{D}$, the perturbation probabilities for all the $m$ output location groups $\alpha_1(x), \dots, \alpha_m(x)$ can also be derived offline (see Section 3.3). This is consistent with other LDP schemes [18, 25, 58].

## 3.3 Optimal Staircase Perturbation Probabilities

Recall that the possible output locations can be partitioned into $m$ groups based on their distances to input and the PDF similar to the staircase mechanism [29] in differential privacy. We define the perturbation probabilities from input $x$ to all the output locations as follows. Given any two output locations $y$ and $y'$ in any two

[3]In `SRR`, every input location $x$ will be only perturbed to another location $y$ in the domain $\mathcal{D}$ (rather than an arbitrary location on the map).

neighboring groups $y \in G_j(x)$ and $y' \in G_{j+1}(x)$, we have probability $q(y|x) = q(y'|x) + \Delta(x)$ where the step $\Delta(x) \in [0, 1)$ is the fixed probability difference for any two neighboring groups of input $x$. Compared to the staircase mechanism in differential privacy which aims to the unbounded domain (entire real line or the set of all integers) and these probabilities are geometric sequence to maintain $\epsilon$-DP, for the bounded location domain, we set the fixed difference between groups.Note that the perturbation probability from the given input location $x$ to output location $y$ decreases as $y$ moves to further groups (larger $j$).

Denoting $\alpha_{max}(x)$ and $\alpha_{min}(x)$ as the max and min probabilities in $\alpha_1(x), ..., \alpha_m(x)$, we have $\alpha_{max}(x) = \alpha_1(x)$ and $\alpha_{min}(x) = \alpha_m(x)$. In SRR, for all the input locations $x \in \mathcal{D}$, we specify a constant $c \geq 1$ as the ratio $\frac{\alpha_{max}(x)}{\alpha_{min}(x)}$. Thus, we have:

$$\Delta(x) = \frac{\alpha_{max}(x) - \alpha_{min}(x)}{m-1} = \frac{\alpha_{min}(x) \cdot (c-1)}{m-1} \tag{3}$$

For each $x \in \mathcal{D}$, the sum of the perturbation probabilities of all the output locations is 1. Given the differences of perturbation probabilities for output locations in different groups in Equation 3 and the number of output locations in each group, all the perturbation probabilities can be derived, including $\alpha_{max}(x)$ and $\alpha_{min}(x)$:

$$\begin{aligned}\alpha_{min}(x) =& \frac{m-1}{(m-1)d \cdot c - (c-1)\sum_{j=2}^{m}[(j-1) \cdot |G_j(x)|]} \\ \alpha_{max}(x) =& \alpha_{min}(x) \cdot c\end{aligned} \tag{4}$$

where $d$ is the location domain size and $|G_j(x)|$ the size of group $G_j(x)$. Notice that, different $\alpha_1(x), \ldots, \alpha_m(x)$ will be derived for different input location $x$ since the group sizes $\forall j \in [1, m]$, $|G_j(x)|$ might be different for different $x$. Thus, the privacy upper bound $\epsilon$ can be computed (for any two input locations $x, x' \in \mathcal{D}$).

Theorem 3.3. *Staircase randomized response (SRR) satisfies $\epsilon$-local differential privacy, where*

$$\epsilon = \max_{x,x' \in \mathcal{D}} \log(c \cdot \frac{(m-1)d \cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1) \cdot |G_j(x)|]}{(m-1)d \cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1) \cdot |G_j(x')|]})$$

Proof. Proven in Appendix B.1. □

For each input location $x \in \mathcal{D}$, the groups $G_1(x), \ldots, G_m(x)$ are constants if $m$ and $\mathcal{D}$ are specified (as discussed in Section 3.2). Thus, given the value of $c$, we can derive a constant $\epsilon$ as a strict privacy upper bound for the LDP guarantee.

**Selecting $c$ for $\epsilon$-LDP**. Since $\epsilon$ is positively correlated to $c$, for any desired $\epsilon$-LDP, the required $c$ can be uniquely calculated using $\epsilon$, $\mathcal{D}$ and $m$ (see the relationship between $\epsilon$ and $c$ in Figure 5(a)). Then, all the perturbation probabilities $\alpha_1(x), \ldots, \alpha_m(x)$ for all the input locations $x \in \mathcal{D}$ can be derived and made available to the users.

**Optimal $m$**. In practice, both the server and clients do not know the data distribution before collecting them. Hence, it is critical to learn the optimal $m$ that is also *independent of input data* and ensure good utility for all possible location data distributions in the SRR mechanism. To this end, we will optimize $m$ for location domain partitioning with the mutual information [46, 61] between the input $x$ and output $y$, which can measure the mutual dependence between them. As mutual information varies for different distributions, the maximum mutual information can cover all the cases (since the mutual dependence of any case would not violate such dependency [40]). Thus, the optimal $m$ can be derived by the upper bound of mutual information for all the distributions [21, 40].

Specifically, the mutual information between $x$ and $y$ is expressed by the difference between the differential entropy and conditional differential entropy of $x$ and $y$ [40]:

$$I(x, y) = H(x) - H(x|y) = H(y) - H(y|x) \tag{5}$$

where $H(\cdot)$ is the entropy function. Since no prior knowledge on the input data, the server can model the output as a random sample with uniform distribution [46]. Then, it considers the distribution of $y$ as uniform distribution $U$ to maximize the mutual information (worst utility of the output $y$) [47]. Thus, we have:

$$I(x, y) \leq H(U) - H(y|x) \tag{6}$$

where $H(U) = \log d$. The conditional differential entropy $H(y|x)$ can be computed as below:

$$\begin{aligned}H(y|x) &= -[\sum_{j=1}^{m} |G_j(x)| \cdot \alpha_j(x) \cdot \log \alpha_j(x)] \\ &\geq -d \cdot \alpha_{min}(x) \log \alpha_{max}(x)\end{aligned}$$

Thus, $H(y|x)$ is lower bounded by $-d \cdot \alpha_{min}(x) \log \alpha_{max}(x)$ for $\alpha_1(x), \ldots, \alpha_m(x)$. Finally, the upper bound of mutual information can be derived by the number of groups $m$:

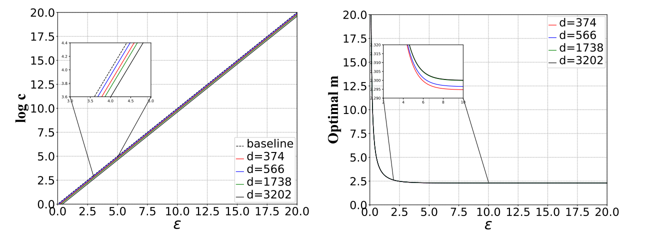
$$I(x, y) \leq \log d - H(y|x) \leq \log d + d \cdot \alpha_{min}(x) \log \alpha_{max}(x)$$

We then explore the optimal $m$ based on the mutual information metric. Since the smaller mutual information between two variables indicates more independence between them, and the mutual information on $m$ for LDP is convex (as proven in Appendix A.1), the optimal $m$ can be derived by making the derivation of the upper bound to 0 which minimizes the mutual information bound.

Lemma 3.4. *The optimal $m$ to minimize the mutual information bound is $m = \frac{2 \cdot (c \cdot d - e^{1+\log c})}{(c-1) \cdot d}$. [See proof in Appendix A.2]*

**Specifying $\epsilon$ for LDP**. In our setting, there are three parameters $\epsilon$, $c$ and $m$. With the given privacy requirement $\epsilon$, the server can calculate the $m$ and the corresponding $c$ with Lemma 3.4 and Theorem 3.3 to make the privacy meet the requirement $\epsilon$. Specifically, we can set a value $c$ and get the corresponding $m$ with Lemma 3.4. Since the location domain can be partitioned into $m$ groups and $\forall x, j \in [2, m-1]$, $|G_j(x)|$ are fixed, we can then calculate the privacy bound by Theorem 3.3 to see if it meets the privacy requirement $\epsilon$. Per the Theorem 3.3, the $\epsilon$ is positively correlated to $c$ with the fixed $m$ and partition groups. Thus, there should be many value $c$ that make the privacy requirement satisfy $\epsilon$. For example, if the $c$ value equals to 5 to meet the privacy requirement $\epsilon = 6$, the value less than 5 should cause the $\epsilon$ smaller that also meets the privacy requirement. However, to fully utilize the privacy that can make the utility maximize, it should only take the maximum of $c$ with the fixed domain. Figure 5 shows the numeric results for $c = \frac{\alpha_{max}(x)}{\alpha_{min}(x)}, \forall x \in \mathcal{D}$ and the optimal $m$ versus a varying $\epsilon \in [0.01, 20]$ (given four different domains in our experimental datasets). The plots confirm that $\epsilon$ is positively correlated to $c$ (given any domain $\mathcal{D}$), and $c$ is extremely close to $e^{\epsilon}$ (slightly smaller). In the experiment, with the given $\epsilon$ value and the domain $\mathcal{D}$, we search the maximum value $c$ to satisfy the $\epsilon$-LDP by binary search method. Figure 5(b), the optimal $m$ is dominantly determined by

$\epsilon$. The optimal $m$ (rounded to its floor or ceiling) is a small integer, e.g., 2-6 for all the four different domains.



(a) $\log c$ vs $\epsilon$ (baseline curve $\epsilon = \log c$) (b) Optimal $m$ vs $\epsilon$

**Figure 5:** $\log c$ **and optimal** $m$ **vs** $\epsilon$ **(and domain size** $d$**); domain size** $d$ **is** 374, 566, 1738 **and** 3202 **in datasets Portcabs [48], Geolife [68], Gowalla [3], and Foursquare [65], respectively**

## 3.4 Perturbation Algorithm

For each location $x \in \mathcal{D}$, the server partitions $m$ groups $G_1(x), \ldots, G_m(x)$ and derives the perturbation probabilities for all the output locations in $m$ groups $\alpha_1(x), \ldots, \alpha_m(x)$. After receiving such information from the server, each client perturbs its location $x$ by sampling the output location $y$. See details in Algorithm 1.

```
Input   : user location $x$, privacy budget $\epsilon$, and domain $\mathcal{D}$
Output  : perturbed location $y$
server pre-computes the optimal $m$ and $\beta_j(x), j \in [1, m]$
foreach location $x \in \mathcal{D}$ do
    foreach group $j \in [1, m]$ do
        foreach location $z \in \mathcal{D}$ do
            if $length(LCP(z, x)) \geq \beta_j(x)$ then
                $G_j(x) \leftarrow z$; $\mathcal{D} \leftarrow \mathcal{D} \setminus z$
            end
        end
    end
    foreach $j \in [1, m]$ do
        compute the perturbation probability $\alpha_j(x)$ for
          locations in $G_j(x)$
    end
end
client samples an output location $y$ from all the locations in
  $G_1(x), \cdots, G_m(x)$ (per Equation 2) and submit it to the server
```

**Algorithm 1:** Staircase Randomized Response

## 3.5 Distribution Estimation

Similar to other LDP mechanisms, the distribution of noisy locations may be biased. Given samples from unknown data distribution $p$, estimating the distribution $\tilde{p}$ of $p$ has been extensively studied [36, 40]. In PrivLBS, we extend the empirical estimation method with two perturbation probabilities [36] to estimate the location distribution from the perturbed locations using staircase perturbation probabilities. In our experiment, we also compare the performance of [36] (named HR) with PrivLBS.

Specifically, in the empirical estimation, for each $x \in \mathcal{D}$, the server creates a candidate location set $C_x$ for input $x$ to estimate the item distribution $\tilde{p}$ from the observed noisy distribution $p$. Each set $C_x$ which contains $\frac{d}{2}$ locations is a subset of the domain. In PrivLBS, the server generates a candidate location set $C_x$ for each $x$ with a Hadamard matrix (a square matrix with either +1 or −1 entries and mutually orthogonal rows). Given $\mathcal{H}_1 = 1$, for any $\mathcal{H}_K$, we have:

$$\mathcal{H}_K = \begin{pmatrix} \mathcal{H}_{K/2} & \mathcal{H}_{K/2} \\ \mathcal{H}_{K/2} & -\mathcal{H}_{K/2} \end{pmatrix} \tag{7}$$

The server then applies a recursion algorithm [36] to generate such Hadamard matrix with size $K \times K$ (denoting it as $\mathcal{H}_K \in \{-1, +1\}^{K \times K}$) where $K = 2^{\lceil \log_2(d+1) \rceil}$ and $d$ is the domain size [36]. Then, each row of $\mathcal{H}_K$ except the 1st row (the 1st row includes only "1" and $\mathcal{H}_K$ includes $d + 1$ rows) can be mapped into a unique location in domain $\mathcal{D}$. Specifically, given location $x \in \mathcal{D}$, its candidate set will be derived using the $(i+1)$th row in $\mathcal{H}_K$ where $i$ is the index of $x$ in $\mathcal{D}$. Then, $\forall x \in \mathcal{D}$, we can generate the candidate set $C_x$ for each user's input $x$ as the locations related to the column indices with a "+1" in the mapping row of matrix $\mathcal{H}_K$ [36]. We denote the candidate set of all the locations in $\mathcal{D}$ as $\mathcal{H}_K \circ \mathcal{D}$.

Let $p(C_x)$ be the probability for sampling $y \in C_x$. Then, we can derive $p(C_x)$ with the output $y$ in the corresponding candidate set in case of inputs $x$ and $x'$ ($x$ differs from $x'$ and $C_x$ also differs from $C_{x'}$). Thus, we have $\forall x \in \mathcal{D}$, $p(C_x) = p(x) \sum_{y \in C_x} q(y|x) + \sum_{x'_i \neq x} p(x') \cdot [\sum_{y \in C_x \setminus C_{x'}} q(y|x') + \sum_{y \in C_x \cap C_{x'}} q(y|x')]$, where $p(x)$ is the distribution of $x$ (to be estimated).

All the perturbation probabilities $q(y|x)$ are known in Equation 2. Thus, for each $x \in \mathcal{D}$, there exists one equation as above. Given $d$ independent linear equations (due to random coefficients), the $d$ variables $\forall x \in \mathcal{D}, p(x)$ can always be solvable. Specifically, $\forall x \in \mathcal{D}, p(C_x)$ are the observed distribution of all the locations from the aggregated noisy data. Each user sends its perturbed location to the server, which derives the total frequency of all the locations in the pre-computed candidate set of location $x$. Then, the above $d$ equations can be constructed for estimating the distribution of all the locations $\forall x \in \mathcal{D}, p(x)$. We apply the lower-upper ($LU$) decomposition algorithm [10, 50] to solve these independent linear equations. Algorithm 2 presents the details.

```
Input   : perturbed locations $y_1, ..., y_n$
Output  : estimated location distribution $\forall x \in \mathcal{D}, \tilde{p}(x)$
server generates the candidate location set $\mathcal{H}_K \circ \mathcal{D}$ for all the
  locations in $\mathcal{D}$
// $\mathbb{I}$ returns 1 if $y \in C_x$; otherwise, 0
foreach $x \in \mathcal{D}$ do
    calculate the $p(C_x)$ with $y_1, ..., y_n$:
      $p(C_x) := \sum_{j=1}^{n} \frac{\mathbb{I}\{y_j \in C_x\}}{n}$
    construct a linear equation for $x$ with $p(C_x)$ and
      perturbation probabilities
end
solve linear equations with the $LU$ decomposition to derive
  $\forall x \in \mathcal{D}, p(x)$
return the estimated location distribution $\forall x \in \mathcal{D}, \tilde{p}(x) = p(x)$
```

**Algorithm 2:** Location Distribution Estimation

## 3.6 Private Retrieval for Client Queries

Recall that the client may need to query the estimated location distribution with its true location, e.g., $k$ nearest users [66] (see Section 6.3), and traffic-aware GPS navigation [56] (see Section 4.2).

In PrivLBS, users can retrieve the results from the server using the Private Information Retrieval (PIR) protocol [5, 26, 35] (when needed), which enables any user to privately retrieve information from a database server without letting the server know which record has been retrieved. In the PIR, the database server has an $n$-bit string $V = \{v_1, ...., v_n\}$, and the client would like to know $v_i$. The client first sends an encrypted request $E(i)$ for the $i$-th value to the server, where $E(\cdot)$ denotes encryption function. The server also responds with an encrypted value $r(v_i, E(i))$ (e.g., by quadratic residuosity). Finally, the client can retrieve the record $v_i$ privately based on the server's encrypted response.

Most of the off-the-shelf PIR algorithms can work as a post-processing component (e.g., [26] takes only a few seconds in our experiments). Moreover, local perturbation and distribution estimation require only $\sim 0.014$ second for the client and a few seconds for the server (see Section 6.5). Thus, the system performance of PrivLBS would be very efficient for real-time LBS deployment.

## 3.7 Privacy and Utility Analysis

**Privacy Analysis.** $\epsilon$-LDP has been proven for the SRR mechanism in Theorem 3.3. The server cannot distinguish users' true locations from the noisy data. Moreover, as post-processing procedures applied on the results of LDP scheme, the empirical estimation and PIR (if needed) do not leak any extra information [23].

**Error Bounds.** Error bounds for the estimation methods in LDP schemes can be derived to understand the expectation of the randomized noise. Then, we derive the error bounds (based on the expectation of the $L_1$ and $L_2$-distance) for the estimated distribution of all the locations $\tilde{p}$ deviated from the true distribution $p$.

Theorem 3.5. *In SRR,* $\mathbb{E}[L_1(\tilde{p}, p)] \leq \frac{2d}{\sqrt{n}\cdot(2\gamma - d\cdot\mu)}$, *where* $\gamma = \min\{\sum_{y \in C_x} q(y|x), x \in \mathcal{D}\}$ *and* $\mu = \min\{\alpha_{min}(x), x \in \mathcal{D}\}$.

Theorem 3.6. *in SRR,* $\mathbb{E}[L_2(\tilde{p}, p)] \leq \frac{2\sqrt{d}}{\sqrt{n}(2\gamma - d\cdot\mu)}$, *where* $\gamma = \min\{\sum_{y \in C_x} q(y|x), x \in \mathcal{D}\}$ *and* $\mu = \min\{\alpha_{min}(x), x \in \mathcal{D}\}$.

Both theorems are proven in Appendix B. Both error bounds decline if increasing the privacy bound $\epsilon$ or the number of users $n$ (thus the error bound would be minor in real-world LBS due to a large number of users). Notice that, the expected $L_1$-loss for the GRR mechanism is upper bounded by $\frac{d}{\epsilon}\sqrt{\frac{2(d-1)}{n\pi}}$ [47], which can be $\sqrt{d}$ times of the SRR error bound in the worst case.

## 4 PRIVLBS FOR TRAJECTORY-INPUT LBS

In this section, we extend SRR to support trajectory-input LBS using two example applications: (1) collecting the origin and destination (OD) of users for OD analysis [7], and (2) collecting a sequence of user locations for traffic-aware GPS navigation [68].

## 4.1 Origin-Destination Analysis

OD analysis aggregates a pair of origin-destination from each user to estimate the traffic flow [7]. In this case, the LDP notion (Definition 2.1) should be extended to protect each user's OD pair.

*Definition 4.1 ($\epsilon$-Local Differential Privacy).* A randomization algorithm $\mathcal{A}$ satisfies $\epsilon$-LDP, if for any two different location pairs $(x_o, x_d), (x'_o, x'_d) \in \mathcal{D}\times\mathcal{D}$, and for any output location pair $(y_o, y_d) \in range(\mathcal{A})$ sent to the untrusted server, we have $Pr[\mathcal{A}(x_o, x_d) = (y_o, y_d)] \leq e^{\epsilon} \cdot Pr[\mathcal{A}(x'_o, x'_d) = (y_o, y_d)]$.

The LDP scheme for OD analysis should preserve the sequential correlation from the origin to the destination (OD pair). Thus, the domain has been greatly expanded to $d^2$ OD pairs in $\mathcal{D} \times \mathcal{D}$. To avoid the bad utility resulted from a large domain, we extend the Lasso regression [54] to a novel *private matching method* to preserve the OD sequence.[4] Then, we integrate the private matching into PrivLBS to ensure accurate OD distribution with $\epsilon$-LDP.

Specifically, users perturb their two locations with privacy budget $\frac{\epsilon}{2}$ for each. The server receives a large number of noisy samples of all users from specific distributions for origins and destinations, respectively. The server may estimate the distribution from the noisy sample space using the linear regression $\vec{y} = \text{M} * \vec{w}$, where matrix M includes the predictor variables, vector $\vec{y}$ includes the response variables, and vector $\vec{w}$ includes the regression coefficients. The predictor variables in M consist of all the combinations of trajectories from each origin to each destination ($d^2$ pairs), which could be known to the server and client beforehand. Moreover, the response variables $\vec{y}$ can be estimated from the SRR perturbed values. Notice that, the frequencies of most combinations $(x_o, x_d) \in \mathcal{D}\times\mathcal{D}$ are very small or even equal to zero in LBS. Thus, Lasso regression [54] can effectively solve such sparse linear regression by encoding the predictor variables M for all the OD pairs.

As shown in Figure 6, we have two steps in PrivLBS: (1) perturbing the origin and destination separately by each client, and (2) estimating the joint distribution of OD pairs using Lasso regression by the server. Each client first applies SRR to perturb the origin and destination with privacy budget $\frac{\epsilon}{2}$ each. Then, the server estimates the distribution of origin and destination to generate the vector $\vec{y}$. Meanwhile, the server encodes the overall candidate set of OD pairs M based on the location domain $\mathcal{D}$. Finally, the server fits a Lasso regression model to the vector $\vec{y}$ and the candidate matrix M to learn $\vec{w}$. Therefore, the non-zero coefficients in $w$ will be considered as the frequencies for the candidate OD pairs.

**Privacy Bound**. Although the origin and destination are correlated, each user sends these two perturbed locations sequentially. The sequential composition of releasing two locations would only result in the total leakage ($\epsilon$-LDP) even if they are highly correlated [23]. The Lasso regression is performed on the two sets of perturbed data (one set of origins and another set of destinations) as post-processing to retain the correlation, which would not consume privacy budget [23]. Thus, the OD analysis still satisfies $\epsilon$-LDP.

## 4.2 Traffic-Aware GPS Navigation

In this App, users may seek the route with shortest time by avoiding congested roads. At that moment, users may update and send multiple locations to the server in sequence. Meanwhile, each user will privately retrieve the real-time nearby traffic from the server to help update the route in case of traffic congestion.

Specifically, the route recommendation algorithm can be deployed in the client to compute the best route with the shortest

[4]Lasso regression was used to generate the synthetic high-dimensional dataset with LDP and preserve the correlation across dimensions [54].

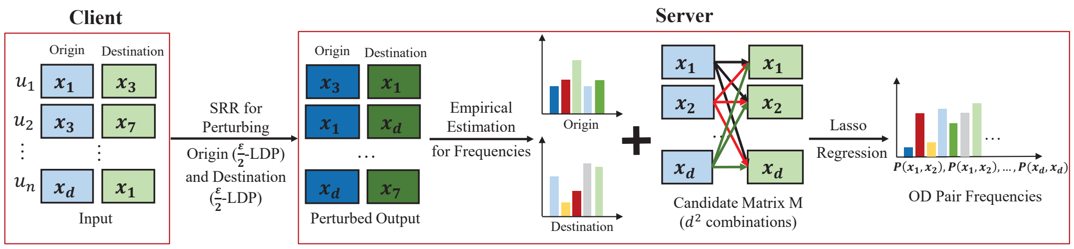

Client
Server
Origin
Destination
$u_1$
$u_2$
$u_n$
$x_1$
$x_3$
$x_7$
$x_d$
Input
SRR for Perturbing
Origin ($\frac{\epsilon}{2}$-LDP) and Destination ($\frac{\epsilon}{2}$-LDP)
Perturbed Output
Empirical Estimation for Frequencies
Origin
Destination
Candidate Matrix M ($d^2$ combinations)
Lasso Regression
$P(x_1,x_2), P(x_1,x_2), \dots, P(x_d,x_d)$
OD Pair Frequencies

**Figure 6: Extending SRR to collect and aggregate origin-destination pairs with $\epsilon$-LDP**

traveling time on an offline map (integrated with the real-time traffic information from the server) [68]. For any route, the total traveling time $t$ can be predicted with the historical dataset.[5] Also, each user can update the current location $x_i$ to the server and learn the current traffic density. Then, the client may recompute the best route and update the estimated traveling time. Intuitively, if the suggested route does not have any traffic, it is unnecessary to update the user's location to learn the real-time traffic density (this would avoid consuming more privacy budget). Thus, we follow this idea to extend our SRR. In PrivLBS, the client will identify these "location updates" (similar to [45]). Let $\mathbb{T}$ denote a trajectory and $Agg(x_o, x_i), x_i \in \mathbb{T}$ represent the actual traveling time from the origin $x_o$ to current location $x_i$. In the meanwhile, the GPS can predict the piece-wise traveling times between the origin $x_o$ and any location $x_i \in \mathbb{T}$ before the arrival. It is worth noting that the time is treated as the condition for the update (as above). It can be extended to update the location with other criterion in specific applications (e.g., distance, and checkpoints).

Denoting such predicted time as $Agg^p(x_o, x_i), x_i \in \mathbb{T}$, the client will examine the difference between their actual traveling time $Agg^t(x_o, x_i)$ and the predicted time $Agg^p(x_o, x_i)$ at different locations $x_i \in \mathbb{T}$. If the client finds that the actual traveling time $Agg^t(x_o, x_i)$ is significantly more than predicted one $Agg^p(x_o, x_i)$, e.g., delayed time exceeds a threshold: $Agg^t(x_o, x_i) - Agg^p(x_o, x_i) > \theta$, there is likely a traffic congestion. Then, the client requests a "location update" to privately upload the perturbed location to the server, and privately retrieve the current traffic density. Moreover, the server will periodically estimate the traffic density using all the perturbed locations collected from the clients in the past time window (e.g., 5 minutes for each time window). Once a location update is requested by any client, the server privately delivers the traffic density to the client via the PIR protocol.

**Privacy Bound**. Since every perturbed location is individually aggregated (based on individual locations) rather than as a combination, such data collection can be done for all the locations separately and simply follows sequential composition [30]. Thus, SRR for such trajectory-input LBS satisfies $\lambda\epsilon$-LDP where $\lambda$ is the number of requested location updates from the origin to the destination. We have empirically evaluated that $\lambda$ is small in practice (e.g., 2 or 3). Finally, PIR may result in side-channel leakage (e.g., who requested the location update may be in the congested areas). If necessary, this can be simply mitigated by an anonymizer (e.g., shuffler [24]), which also further amplifies the LDP protection [24].

[5] These historical datasets could be obtained from public traces and check-in datasets, or datasets generated from LBS applications.

## 5 DISCUSSION

**Relaxed LDP**. Some recent works [4, 30, 31] relaxed the LDP by considering the input variants. For instance, ID−LDP [30] relaxes the LDP with different $\epsilon$ for different inputs; geo-indistinguishability (GI) relaxes the protection to only the locations within a radius; CLDP [31] provides distance discriminative privacy, and relaxes the protection for different pairs of inputs. Different from PrivLBS, all of them cannot strictly satisfy $\epsilon$-LDP. To validate their limitations on rigorous LDP guarantee, we present some numeric analysis with the same setting (by converting them to $\epsilon$-LDP). *Note that PLDP [13] is experimentally compared in Section 6 since it focuses on LBS.*

First, we generate a synthetic dataset including items with uniformly distributed frequencies (the distance between inputs can also be directly measured). For ID−LDP, we randomly assign the privacy bound from $\{0.5\epsilon, 0.8\epsilon, \epsilon\}$ to each distinct item. Since $\{0.5\epsilon, 0.8\epsilon, \epsilon\}$-ID-LDP satisfies $\min\{\{\epsilon\}, 2 * \{0.5\epsilon\}\}$-LDP, it can guarantee $\epsilon$-LDP for all the items (*if any $\epsilon$ where $> 1$ is assigned to any item, $\epsilon$-LDP will be violated*). For GI, we sample the output $y$ with the Laplace-based PDF centered at input $x$. For CLDP, we adopt the conversion between $\epsilon$ and $\alpha$ [31]. Table 1 shows the $L_1$-distance of the outputs on different $\epsilon$. The utility of PrivLBS significantly outperforms all the relaxed LDP with the same LDP guarantees.

**Table 1: Average $L_1$-distance**

| Privacy Bound $\epsilon$ | 0.5 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| ID−LDP | 2.14 | 1.97 | 1.64 | 1.45 | 1.18 |
| GI | 2.21 | 1.84 | 1.75 | 1.43 | 1.21 |
| CLDP | 0.93 | 0.90 | 0.84 | 0.72 | 0.70 |
| PrivLBS | 0.65 | 0.62 | 0.51 | 0.44 | 0.36 |

**Security/Robustness**. Similar to most LDP schemes [1, 9, 25], PrivLBS focuses on making the strong privacy notion LDP practical for different LBS apps, which is orthogonal to security/robustness studies. The recent data poisoning attack on LDP [11] showed that 5% fake users may affect the aggregation result in the LDP protocols. It may not be easy to launch such an attack on PrivLBS by manipulating mobile devices. Also, if malicious users submit impossible locations, the server may simply detect it with the domain.

**Generalization.** PrivLBS can be potentially extended to other data types if the distances between values/items can be measured (e.g., numerical data). In such contexts, the data items can also be partitioned and staircase perturbation probabilities can be derived and allocated to values/items in different groups. We will evaluate its performance in other domains and benchmark with the corresponding LDP schemes (e.g., Piecewise [62]) in the future.

**Encoding and Precision**. The precision of the encoded locations can be tuned by the level of the bit string hierarchy. Although larger

$h$ more accurately encodes locations, the domain size will grow and thus the perturbation probability (for the true location) may decline for the same privacy. Thus, larger $h$ does not necessarily make the staircase perturbation scheme more accurate (thus we use the standard $h = 23$ as Bing Map).

**System Deployment**. `PrivLBS` can be deployed as an application or integrated with the existing LBS applications in the server and clients (e.g., mobile devices). Given the privacy bound $\epsilon$ and a location domain $\mathcal{D}$, the server will pre-compute the required $c$, the optimal $m$, the GLCP for group partitioning $\forall x \in \mathcal{D}, \beta_1(x), \dots, \beta_m(x)$, and the perturbation probabilities $\forall x \in \mathcal{D}, \alpha_1(x), \dots, \alpha_m(x)$ for `SRR`, and then shared them to all the clients. In `PrivLBS`, the location domain is updated periodically by the server rather than per users' requests. It would not cause any privacy leakage, and it is very efficient to update the domain. If a user is at a location not in the domain before the update, the client will approximate it to the nearest location in the domain. Each client only needs to perturb their locations based on the stored $md$ perturbation probabilities $q(y|x)$, and then directly send the output to the server. Even if the client may privately retrieve the analysis result related to his/her location from the server, the PIR protocol can be efficiently executed without many overheads. Thus, the clients do not need to be equipped with strong computing capabilities (mobile devices suffice). Each client should download an offline map if required in certain LBS applications, e.g., traffic-aware GPS navigation.

**Provable Privacy for PIR and LDP.** The PIR protocol is applied as the post-processing to the query results that guarantees $\epsilon$-LDP. The PIR protocol does not cause any additional information leakage since the query results are retrieved based on the encrypted location by employing the provably secure cryptographic technique. From the viewpoint of the server, the PIR request might be originated from any user. Therefore, the probability to identify every user as the querying user is exactly 1/n (for all the users). Thus, it does not cause additional leakage from such random guess either (after the private data collection with $\epsilon$-LDP).

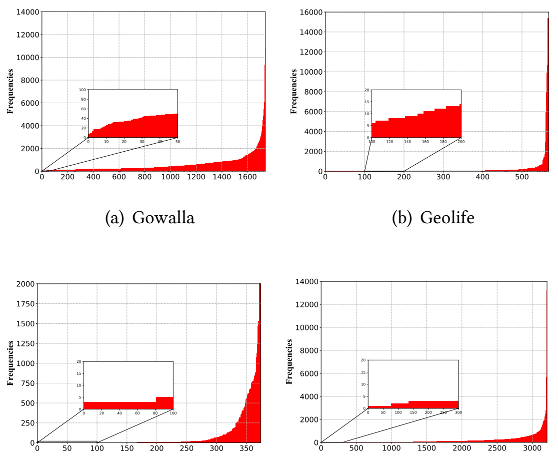
**Complex Applications**. The staircase randomized response can generate more accurate location distribution than existing LDP mechanisms. As a key building block of LBS applications, such high accurate location frequency/distribution estimation by the proposed `SRR` mechanism could universally support different LBS applications, including complex LBS such as traffic-aware GPS navigation. In our experiments, we simulate the route recommendation by the GPS, which shows better performance of `SRR` (see the details in Appendix C). In practice, as the LBS application becomes more complicated (e.g., more data collection), `SRR` would outperform the state-of-the-art LDP schemes more.

## 6 EXPERIMENTS

### 6.1 Experimental Setting

**Experimental Datasets**. We conduct our experiments on four real-world location datasets.

- *Gowalla Dataset* [3] collects 6,442,890 check-ins records of 196,591 users in Austin, USA via the social network app Gowalla between 02/2009 and 10/2010.



(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

**Figure 7: Location frequencies in experimental datasets**

- *Geolife Dataset* [68] collects 17,621 GPS trajectories of 182 users in Beijing between 04/2007 and 08/2012.
- *Portocabs Dataset* [48] collects the GPS trajectories of 441 taxis in Porto between 07/2013 and 06/2014.
- *Foursquare Dataset* [65] collects 90,048,627 check-in locations of 2,733,324 users in New York City, USA.

Since each of the four datasets is collected from locations within a city, we focus on a large geographical region covering a 40×30km$^2$ area for each dataset. Since the encoded bit strings for all the locations in each dataset share a 20-bit common prefix, the last 26 bits (out of 46 bits for $h = 23$) could sufficiently index all the locations with high accuracy for all the 4.7m×4.7m regions (removing the common prefixes does not affect the accuracy due to fixed domain size and groups). All the experiments were performed on the NSF Chameleon Cluster with Intel(R) Xeon(R)Gold 6126 2.60GHz CPUs and 192G RAM [49]. Docker is used to start containers to emulate the server/clients with system and network setup.

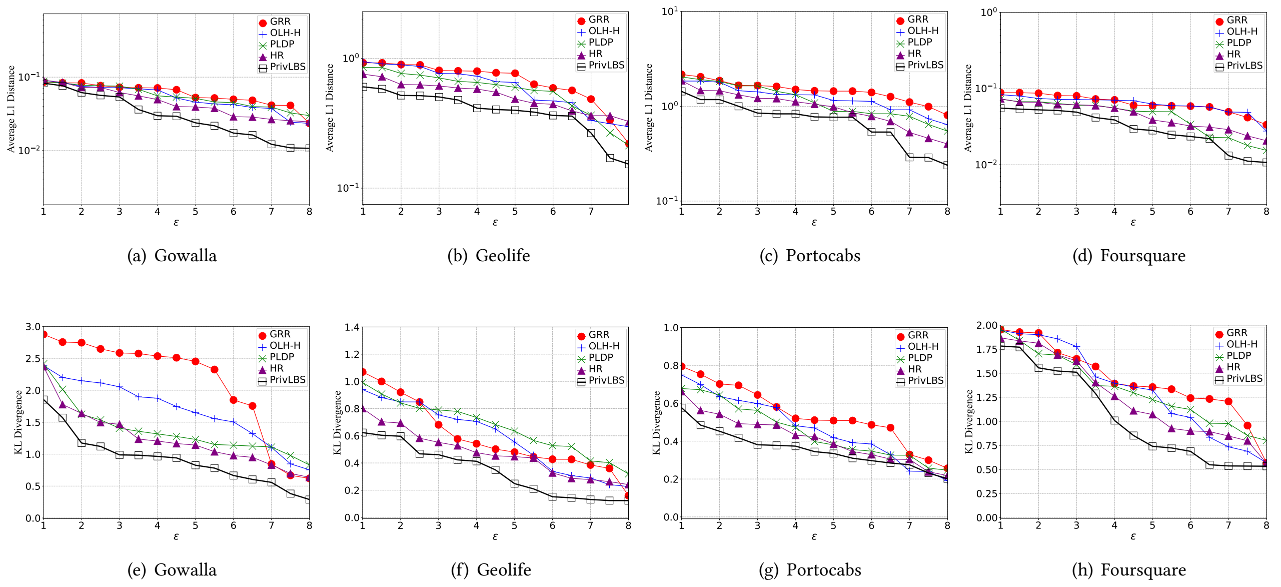
**Dataset Characteristics.** Table 2 presents the number of locations and users in four datasets. The total user number can vary from 30,000 to 1M. As we know, infrequent locations in the LDP can cause more utility loss than frequent locations. So we use four dataset that have different densities of users. Figure 7 presents the original frequencies of all the locations in four datasets.

**Table 2: Characteristics of datasets (after pre-processing)**

| Dataset | Location # | User # |
|---|---|---|
| Gowalla | 1,738 | 1,120,147 |
| Geolife | 566 | 104,488 |
| Portcabs | 374 | 34,438 |
| Foursquare | 3,202 | 701,528 |

### 6.2 Distribution Estimation (Location-Input)

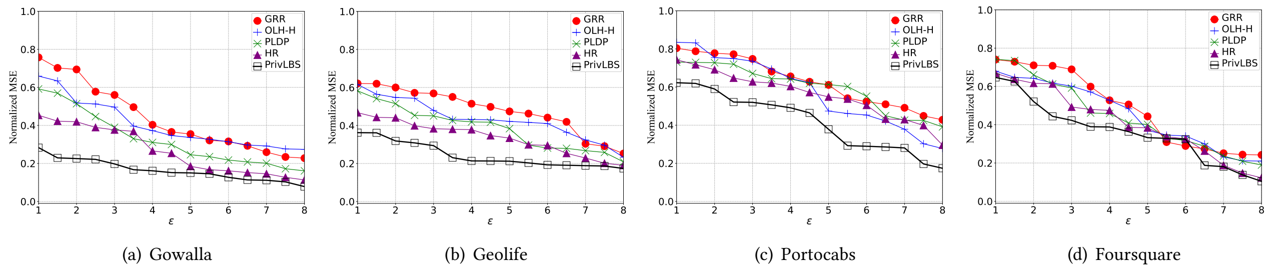
We first evaluate the utility of `PrivLBS` for the distribution estimation while benchmarking with the state-of-the-art LDP schemes, including Generalized Randomized Response [60] (`GRR`), Optimal Local Hash with hierarchy structure [59] (`OLH-H`), the Location



(a) Gowalla (b) Geolife (c) Portocabs (d) Foursquare

(e) Gowalla (f) Geolife (g) Portocabs (h) Foursquare

**Figure 8: Average $L_1$-distance and KL-divergence for the distribution estimation on four datasets using different LDP schemes**



(a) Gowalla (b) Geolife (c) Portocabs (d) Foursquare

**Figure 9: MSE of all the locations' $k$-NN lists on four datasets using different LDP schemes ($k$ = 25)**

Data Aggregation [13] (PLDP) and the Hadamard Response (HR) [36]. We follow the original perturbation and estimation method in each benchmark. Here, we choose the OLH mechanism since it has better utility than unary encoding (UE) and choose the existing location LDP framework PLDP instead of existing location framework in [37, 67] since PLDP is an optimized framework that boosts the utility. For fair comparisons, in OLH−H, we randomly sample a hierarchical level for each location. Then, we adapt the constrained inference [32] to adjust the frequencies of parent and leaf nodes for consistency. In PLDP, we assign the same protection region level for all the users as other LDP schemes to satisfy the strict $\epsilon$-LDP.

The server derives the spatial density for many LBS applications, e.g., urban traffic density [51], and crowd density for events [55]. In most existing LDP settings, the $\epsilon$ is in the range between 0.5 to 10 for privacy protection. Too large $\epsilon$ value can't protect user's location well. Similar to that, we set $\epsilon$ between 1 and 8 with a step of 0.5 (covering both strong and weak privacy regime).

Figure 8 shows the average $L_1$-distance and KL-divergence between the true and estimated distributions of all the locations. Both $L_1$-distance and KL-divergence decrease as $\epsilon$ increases. Especially for the GRR, the error dramatically decreases since the probability grows exponentially. However, PrivLBS still greatly outperforms other LDP schemes on all the four datasets.

## 6.3 Case Study I: $k$-NN Query (Location-Input)

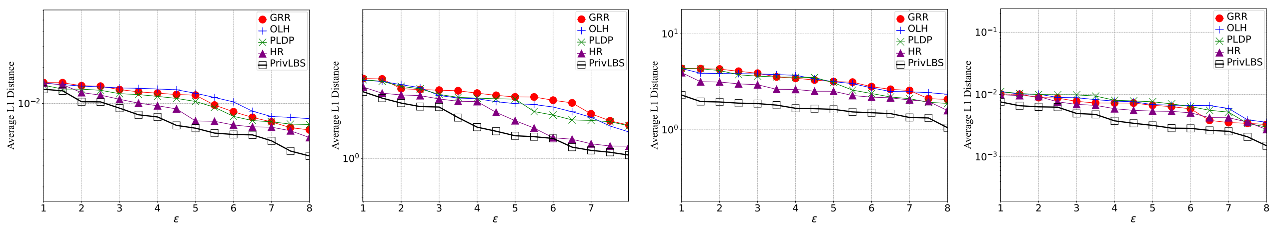
We first evaluate the performance of SRR in specific applications on recommendations based on the location distribution. $k$-nearest neighbors ($k$-NNs) is a typical application in which queries can be made for the nearest point-of-interests or users. We next show the results for querying the $k$-NN users [66], which can be extended from the distribution estimation. The $k$-NN lists for any user (with a location) are the other $k$ closest users, measured by the MSE of their coordinates. Then, given the estimated location distribution, the server can directly derive each location's list of $k$-NNs.

**$k$-NN Lists Computed by Server.** Figure 9 shows the normalized MSE between the true and estimated coordinates of all the users' $k$-NN lists. The normalized MSE also decreases while $\epsilon$ increases. In Figure 9(a), 9(b), 9(c), and 9(d), PrivLBS outperforms GRR, OLH−H, PLDP, and HR, which is consistent with the previous results.

We also present the *precision* and *recall* of all the users' estimated $k$-NN lists in Table 3. Again, PrivLBS can produce more accurate $k$-NN lists than all the other LDP schemes. Note that $\epsilon$ might be

**Table 3: Precision and recall for the derived $k$-NNs of all the users ($k$=25)**

| Dataset | $\epsilon$ | GRR | | OLH-H | | PLDP | | HR | | PrivLBS | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Precision | Recall | Precision | Recall | Precision | Recall | Precision | Recall | Precision | Recall |
| Gowalla | 1 | 31.9% | 47.6% | 27.1% | 35.6% | 38.1% | 46.4% | 53.2% | 63.1% | **60.7%** | **69.4%** |
| | 3 | 55.5% | 54.1% | 30.6% | 38.9% | 50.1% | 56.9% | 66.8% | 74.7% | **68.7%** | **77.4%** |
| | 5 | 63.5% | 66.2% | 51.0% | 57.2% | 67.6% | 74.3% | 68.3% | 75.8% | **73.6%** | **78.1%** |
| | 7 | 78.4% | 81.2% | 68.8% | 73.3% | 73.9% | 79.5% | 75.4% | 80.9% | **80.1%** | **81.9%** |
| | 9 | 86.1% | 87.2% | 69.2% | 74.4% | 80.3% | 85.3% | 82.1% | 84.1% | **87.7%** | **89.3%** |
| Geolife | 1 | 17.8% | 26.4% | 30.1% | 34.2% | 33.4% | 34.2% | 30.8% | 35.3% | **35.2%** | **39.9%** |
| | 3 | 35.0% | 43.6% | 42.4% | 49.1% | 48.7% | 54.5% | 50.4% | 53.1% | **51.6%** | **58.7%** |
| | 5 | 53.4% | 60.3% | 60.5% | 65.9% | 69.9% | 74.9% | 67.1% | 68.8% | **78.3%** | **83.3%** |
| | 7 | 78.6% | 82.9% | 73.1% | 76.5% | 77.0% | 80.7% | 76.4% | 78.1% | **85.9%** | **88.9%** |
| | 9 | 91.4% | 93.0% | 89.8% | 90.8% | 90.2% | 93.8% | 90.8% | 92.2% | **92.7%** | **94.2%** |
| Portocabs | 1 | 41.9% | 50.7% | 30.4% | 40.4% | 48.8% | 58.4% | 51.2% | 58.4% | **56.2%** | **64.1%** |
| | 3 | 63.8% | 72.6% | 43.6% | 50.6% | 55.6% | 63.3% | 57.7% | 63.1% | **68.3%** | **75.8%** |
| | 5 | 70.5% | 78.2% | 61.9% | 66.9% | 70.6% | 76.1% | 59.7% | 65.0% | **77.4%** | **83.8%** |
| | 7 | 87.8% | 93.3% | 66.0% | 69.4% | 76.2% | 81.3% | 84.9% | 88.2% | **92.7%** | **98.1%** |
| | 9 | 93.4% | 98.7% | 86.5% | 89.5% | 86.7% | 89.2% | 91.6% | 93.3% | **95.9%** | **98.9%** |
| Foursquare | 1 | 32.2% | 40.9% | 42.2% | 52.1% | 46.6% | 56.1% | 52.2% | 60.7% | **55.7%** | **65.3%** |
| | 3 | 58.8% | 65.2% | 50.1% | 57.4% | 50.1% | 58.0% | 59.1% | 67.3% | **67.1%** | **75.3%** |
| | 5 | 80.7% | 84.6% | 64.6% | 68.6% | 68.1% | 75.7% | 80.6% | 86.9% | **83.9%** | **87.2%** |
| | 7 | 87.1% | 89.7% | 65.4% | 69.1% | 68.7% | 76.3% | 85.4% | 88.9% | **87.2%** | **91.3%** |
| | 9 | 88.1% | 92.3% | 76.1% | 77.4% | 82.6% | 86.9% | 86.1% | 89.1% | **91.3%** | **94.6%** |



(a) Gowalla (b) Geolife (c) Portocabs (d) Foursquare

**Figure 10: Average $L_1$-distance for the OD pair frequency on four datasets using different LDP schemes**

relatively large for very high accuracy (e.g., $\epsilon = 5$ similar to the privacy setting by Apple [1]). If involving more users in the practical LBS App, $\epsilon$ can be much smaller for such very high accuracy.

## 6.4 Case Study II: Trajectory-Input LBS

We next evaluate the performance of PrivLBS on collecting trajectories for two example LBS applications: (1) origin and destination (OD) analysis which estimates the OD pairs frequencies with the Lasso regression, and (2) traffic-aware GPS navigation.

**OD Analysis**. The true number of distinct OD pairs in four datasets are 2,315, 876, 1,034, and 5,634, respectively. We apply the same Lasso regression algorithm to all the LDP schemes. Figure 10 presents the average $L_1$-distance between the true and estimated OD pair distribution. As $\epsilon$ increases, $L_1$-distance decreases. PrivLBS again shows the smallest $L_1$-distance of PrivLBS in all the experiments. Moreover, we also observe that the $L_1$-distance is smaller than LBS with single-location input (see Figure 8).

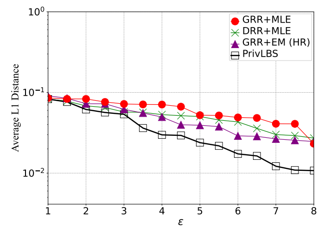
**Traffic-Aware GPS Navigation**. To test the performance of the traffic-aware GPS navigation, we make the simulation the experiment of recommendation for the fastest route. We can also draw the conclusion that PrivLBS outperforms other LDP schemes (see the detailed results and discussions in Appendix C).

## 6.5 Ablation Study and Runtime

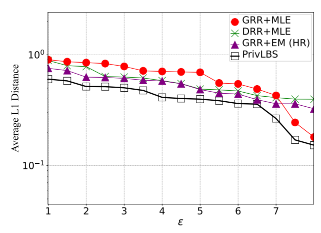
**Ablation Study**. We compare the results with different combinations of perturbation mechanisms (GRR, HR and SRR) and estimation methods. Since the standard estimation method cannot be applied to SRR (more than two perturbation probabilities), we apply the maximum likelihood estimation (MLE) instead. Moreover, the GRR with empirical estimation (EM) is a special case of Hadamard response (HR): $|C_x| = 1$. Figure 11 shows that SRR and the revised EM (PrivLBS) perform the best. Even with the MLE, SRR is better than GRR in most cases. Also, the revised EM can further boost the utility of SRR (compared to SRR and MLE).

**Runtime**. Since users only need to perturb their locations, the user-side runtime is negligible. It takes only 0.014 second for each user on average in the experiments, and thus we only report the server-side runtime in Figure 12. We test 10% to 100% of each dataset with a step of 10%. Similar to GRR, OLH-H, PLDP and HR, the runtime of PrivLBS only slightly increases as the number of users reaches ~1M (e.g., 9 seconds for Gowalla dataset), which is acceptable.

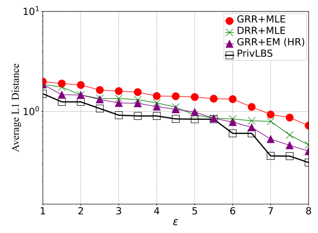
Notice that, there is an offline partitioning time for PrivLBS. Thus, we also present such offline partitioning time w.r.t. the number of users and the number of locations, as shown in Figure 13. We



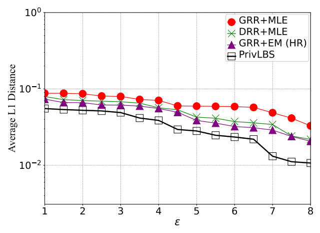
(a) Gowalla



(b) Geolife

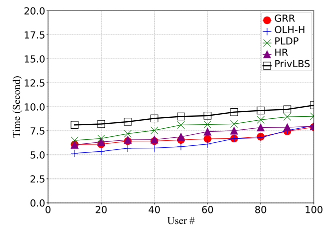


(c) Portocabs

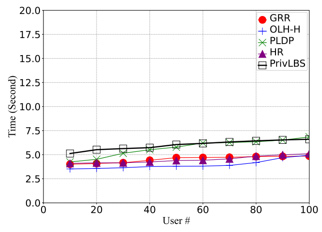


(d) Foursquare

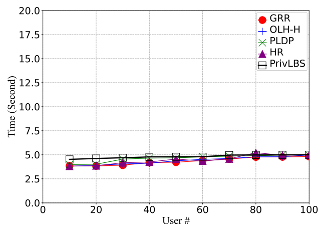
**Figure 11: Average $L_1$-distance for frequency estimation using different combinations of perturbation and estimation methods**



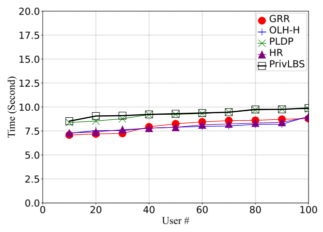
(a) Gowalla



(b) Geolife



(c) Portocabs

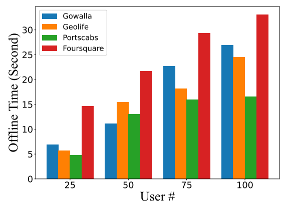


(d) Foursquare

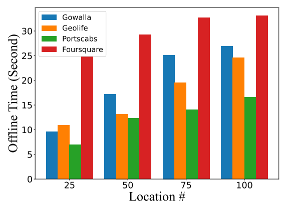
**Figure 12: Runtime for the server (vs. the number of users)**

uniformly extract 25%, 50%, 75%, and 100% of users and locations from each dataset at random as the test datasets. As shown in Figure 13(a), the offline time increases as the number of users increases due to the growth of distinct locations.

In Figure 13(b), we also see that the offline runtime grows linearly on the number of locations, and the offline time is around 30 seconds at most. However, the offline execution is needed when the location domain is updated. Recall that the domain is only updated periodically (e.g., every day). Thus, such offline costs are efficient for real-world deployments.



(a) Runtime vs.# of users



(b) Runtime vs. # of location

**Figure 13: Offline runtime**

## 7 RELATED WORK

Many privacy preserving location-based services techniques have been proposed (e.g., [8, 57]). $K$-anonymity was first defined to protect privacy for LBS. Dummy locations [57] and cloaking region [8] have been utilized for anonymity. However, these methods are highly vulnerable to background knowledge attacks. Another type of techniques design cryptographic protocols [52] to securely perform LBS computations. However, both computational costs and communication overheads might be very high. More recently, rigorous privacy notion differential privacy (DP) has also been applied to LBS applications [33, 34, 41, 44]. For instance, a synthetic data generation technique [44] was proposed to publish statistical information about commuting patterns (including locations) with DP guarantee. Moreover, a quadtree spatial decomposition technique [34] has been used to ensure DP in a database with location pattern mining capabilities. However, the DP model may not be suitable to real LBS applications in case that the users do not trust the server.

The emerging LDP models enable private data collection by untrusted server, which provides stronger protection than the centralized DP models. It has been extended to privately collect different types of data (e.g., histogram [9, 25], social graphs [53], itemsets [58]). Meanwhile, LDP has been successfully deployed in industry (e.g., Google [25], Apple [1], and Microsoft [18]). Recall that two works directly apply randomized response and unary encoding to collect workload-aware indoor positioning data [37] and generate synthetic locations [67] but result in poor utility. Moreover, several relaxed LDP notions have been proposed to protect location privacy [4, 13]. Andrés et al. [4] relaxes the protection for locations within a radius via geo-indistinguishability. Chen et al. [13] relaxes LDP to `PLDP` which allows users to specify personalized privacy budgets for private location collection. However, they cannot ensure rigorous LDP and are also less accurate than our `SRR`.

## 8 CONCLUSION

Severe privacy risks arise in LBS applications due to sensitive location collection. To address the deficiency on privately collecting locations with LDP guarantees and high utility, we propose a novel LDP mechanism “Staircase Randomized Response” (`SRR`) and extend the empirical estimation for `SRR` to significantly improve the accuracy of the LDP model for LBS applications. In addition, we have also extended `SRR` to privately collect trajectories with $\epsilon$-LDP. We have conducted extensive experiments on real datasets to show that `PrivLBS` drastically outperforms other LDP schemes.

## REFERENCES

[1] https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf

[2] https://docs.microsoft.com/en-us/bingmaps/articles/bing-maps-tile-system

[3] http://snap.stanford.edu/data/loc-gowalla.html

[4] M. Andrés, N. Bordenabe, K. Chatzikokolakis and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *CCS*, 2013.

[5] W. Al Amiri, et al.. Privacy-preserving smart parking system using blockchain and private information retrieval. In *SmartNets*, pages 1–6, 2019.

[6] H. H. Arcolezi, J. F. Couchot, B.A. Bouna, and X. Xiao (2021). Improving the Utility of Locally Differentially Private Protocols for Longitudinal and Multidimensional Frequency Estimates. arXiv preprint arXiv:2111.04636.

[7] M.G. Bell. The estimation of an origin-destination matrix from traffic counts. In *Transportation Science*, 1983.

[8] B. Bamba, L. Liu, P. Pesti, and T. Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *WWW*, 19(4), pages 237–246, 2008.

[9] R. Bassily and A. Smith. Local, private, efficient protocols for succinct histograms. In *STOC*, 2015.

[10] C. Camarero. Simple, Fast and Practicable Algorithms for Cholesky, LU and QR Decomposition Using Fast Rectangular Matrix Multiplication (2018). arXiv preprint arXiv:1812.02056.

[11] X. Cao, J. Jia, and N. Gong. Data poisoning attacks to local differential privacy protocols. In *USENIX*, 2021.

[12] K. Chatzikokolakis, E. ElSalamouny, C. Palamidessi and A. Pazii. Methods for location privacy: A comparative overview. In *Foundations an Trends in Privacy and Security*, 19(4), pages 12:1–12:31, 2017.

[13] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin. Private spatial data aggregation in the local setting. In *ICDE*, pages 289–300, 2016.

[14] 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. In *SIGKDD*, pages 213–221, 2012.

[15] S. Chen, X. Zhang, M. K. Reiter, and Y. Zhang. Detecting privileged side-channel attacks in shielded execution with Déjá Vu. In *ASIACCS*, pages 7–18, 2017.

[16] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang. Privacy at scale: Local differential privacy in practice. In *SIGMOD*, pages 1655–1658, 2018.

[17] T. Dao, S. Jeong, and H. Ahn. A novel recommendation model of location-based advertising: Context-Aware Collaborative Filtering using GA approach. In *Expert Systems with Applications*, 2012.

[18] B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. arXiv:1712.01524.

[19] K. Dong, T. Guo, H. Ye, X. Li, and Z. Ling. On the limitations of existing notions of location privacy. In *Future Generation Computer Systems*, pages 1513–1522, 2018.

[20] R. Dewri and T. Ramakrisha. Exploiting service similarity for privacy in location-based search queries. In *TPDS*, pages 374–383, 2013.

[21] J.C Duchi, M. I Jordan, and M. J Wainwright. Local privacy and statistical minimax rates. In *FOCS*, 2013.

[22] U. Demiryurek, F. Banaei-Kashani, C. Shahabi, and A. Ranganathan. Online computation of fastest path in time-dependent spatial networks. In *SSTD*, 2011.

[23] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. In *Foundations and Trends in Theoretical Computer Science 9(3-4)*, 2014.

[24] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *SODA*, 2019.

[25] Ú. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, pages 1054–1067, 2014.

[26] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K. Tan. Private queries in location based services: Anonymizers are not necessary. In *SIGMOD*, 2008.

[27] R. Gideon and A. M. Rothan. Location and scale estimation with correlation coefficients.In *Commun. Stat.*, pages 1561–1572, 2011.

[28] M.E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei. Utility-aware synthesis of differentially private and attack-resilient location traces. In *CCS*, 2018.

[29] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath (2015). The staircase mechanism in differential privacy. In *JSTSP*, 9(7), 1176–1184, 2015.

[30] X. Gu, M. Li, L. Xiong, and Y. Cao. Providing input-discriminative protection for local differential privacy. In *the International Conference on Data Engineering (ICDE)*, pages 505–516, 2020.

[31] M.E. Gursoy, A. Tamersoy, S. Truex, W. Wei, and L. Liu. Secure and Utility-Aware Data Collection with Condensed Local Differential Privacy. In *TDSC*, 2019.

[32] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private histograms through consistency. In *VLDB Endowment*, 2010.

[33] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava. DPT: differentially private trajectory synthesis using hierarchical reference systems. In*Proceedings of the VLDB Endowment*, 8(11), 1154-1165, 2015.

[34] S.S. Ho and S. Ruan. Differential privacy for location pattern mining. In *SPRINGL*, pages 17–24, 2011.

[35] P. Indyk and D. P. Woodruff. Polylogarithmic Private Approximations and Efficient Matching. In *TCC*, 2006.

[36] P. Kairouz, K. Bonawitz, and D. Ramage. Discrete distribution estimation under local privacy. arXiv preprint arXiv:1602.07387.

[37] J. Kim and B. Jang. Workload-aware indoor positioning data collection via local differential privacy. In *Communications Letters*, pages 1352–1356, 2019.

[38] L. Li, R. Lu, and C. Huang. EPLQ: Efficient privacy-preserving location-based query over outsourced encrypted data. In *IEEE IoT Journal*, 2016.

[39] C. Li, B. Palanisamy, and J. Joshi. Differentially private trajectory analysis for points-of-interest recommendation. In *BigData Congress*, pages 49–56, 2017.

[40] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric. Estimating numerical distributions under local differential privacy. In *SIGMOD*, 2020.

[41] L. Yu, L. Liu, and C. Pu. Dynamic Differential Location Privacy with Personalized Error Bounds. In *NDSS*, 2017.

[42] Van Laarhoven, P. JM, and E. H. Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15, 1987.

[43] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam (2007). l-diversity: Privacy beyond k-anonymity. *TKDD*, 1(1), 3-es.

[44] A. Machanavajjhala, D. Kifer, J. M. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *ICDE*, pages 277–286, 2008.

[45] M. Joseph, et al. Local differential privacy for evolving data. arXiv preprint arXiv:1802.07128 (2018).

[46] A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, and S. Vadhan. The limits of two-party differential privacy. In *the Foundations of Computer Science*, pages 81–90, 2010.

[47] T. Murakami, and Y. Kawamoto. Utility-optimized local differential privacy mechanisms for distribution estimation. In *USENIX Security*, pages 1877–1894, 2019.

[48] L. Moreira-Matias,J. Gama, M. Ferreira,J. Mendes-Moreira, , and L. Damas. Predicting taxi–passenger demand using streaming data. In *T-ITS*, 2013.

[49] K. Keahey et al. Lessons Learned from the Chameleon Testbed. In *USENIX ATC*, 2020.

[50] C. Ozcan and B. Sen. Investigation of the performance of LU decomposition method using CUDA. In *Procedia Technology*, pages 50–54, 2012.

[51] C. Ozkurt and F. Camci. Automatic traffic density estimation and vehicle classification for traffic surveillance systems using neural networks. In *Mathematical and Computational Applications*, pages 187–196, 2009.

[52] Asuquo, Philip, et al. Security and privacy in location-based services for vehicular and mobile communications: An overview, challenges, and countermeasures. In *IEEE Internet of Things Journal* pages: 4778-4802, 2018.

[53] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren. Generating synthetic decentralized social graphs with local differential privacy. In *CCS*, pages 425–438, 2017.

[54] Ren, Xuebin, et al. LoPub: High-Dimensional Crowdsourced Data Publication with Local Differential Privacy. In *TIFS*, 13.9 (2018): 2151-2166.

[55] P. Shankar, V. Ganapathy, and L. Iftode. Privately querying location-based services with sybilquery. In *UbiComp*, pages 31–40, 2009.

[56] S. Shang, H. Lu, T. B. Pedersen, and X. Xie. Finding traffic-aware fastest paths in spatial networks. In *International Symposium on Spatial and Temporal Databases*, pages 128–145, 2013.

[57] R. Shokri, G. Theodorakopoulos, J.Y. Le Boudec, and J.- P. Hubaux. Quantifying location privacy. In *IEEE Symposium on Security and Privacy*, 2011.

[58] T. Wang, J. Blocki, N. Li, and S. Jha. Locally differentially private protocols for frequency estimation. In *USENIX Security*. pages 729–745, 2017.

[59] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha. Answering multi-dimensional analytical queries under local differential privacy. In *SIGMOD*, pages 159–176, 2019.

[60] S. Wang et al. Private weighted histogram aggregation in crowdsourcing. arXiv:1607.08025, 2016.

[61] S. Wang et al. Mutual information optimally local private discrete distribution estimation. In *WASA*, 2016.

[62] N. Wang, X. Xiao, Y. Yang, T.D. Hoang, H. Shin, J. Shin, and G. Yu. PrivTrie: Effective frequent term discovery under local differential privacy. In *ICDE*, 2018.

[63] Y. Wang, X. Wu, D. Hu. PrivTrie: Using Randomized Response for Differential Privacy Preserving Data Collection. In *EDBT/ICDT Workshops*, 2018.

[64] D. Woodard, et al. Predicting travel time reliability using mobile phone GPS data. In *Transportation Research Part C: Emerging Technologies* 75, 2017.

[65] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux. Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach. In *WWW*, 2019.

[66] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan. Practical approximate k nearest neighbor queries with location and query privacy. In *TKDE*, 28(6), pages 1546-1559, 2016.

[67] X. Zhao, Y. Li, Y. Yuan, X. Bi, and G. Wang, Ldpart: Effective location-record data publication via local differential privacy In *IEEE Access*, 2019.

[68] Y. Zheng, L. Zhang, X. Xie, and W.Y. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the conference on World Wide Web*, pages 791–800, 2009.

## APPENDIX

## A PROOFS

### A.1 Proof of Convex Property w.r.t. $m$

Proof. With the mutual information bound function $H$, we can take its second order derivative in $m$ as follows:

$$\frac{\partial^2 H}{\partial m^2} =[\frac{1}{c \cdot d - \frac{m\cdot(c-1)\cdot d}{2}} \cdot \log \frac{c}{c - \frac{m\cdot(c-1)\cdot d}{2}}]''$$
$$=\frac{(c-1)^2 d^2}{4(c \cdot d - \frac{(c-1)d}{2} \cdot m)^2} \cdot (2\log \frac{c}{c\cdot d - \frac{(c-1)d}{2}\cdot m} + 3)$$

When the first order derivative is equal to zero, we have $m = \frac{2\cdot(c\cdot d - e^{1+\log c})}{(c-1)\cdot d}$. It is very straightforward to prove that the second order derivative is greater than zero since $(cd - \frac{(c-1)d}{2}m)^2 > 0$ and $2\log\frac{c}{cd - \frac{(c-1)d}{2}m} + 3 > 0$. Therefore, it is a convex function, and we can derive its minimum value by the derivative. □

### A.2 Proof of Optimal $m$

Proof. The mutual information bound is $\log d + d \cdot \frac{m-1}{(m-1)\cdot c\cdot d - R} \cdot \log\frac{c(m-1)}{(m-1)\cdot c\cdot d - R}$ where $R = (\sum_{j=2}^{m}\{(j-1)\cdot|G_j|\})\cdot(c-1)$ is a part of $\alpha_{min}(x)$ (see Equation 4). We can see that $R$ is also determined by $m$. If $|G_1| \neq |G_2| \neq \cdots \neq |G_m|$, $R$ non-differentiable (discrete). To solve this, we consider the worst case: assuming group size $d$ and $R$ is replaced with $R_{max} = (\sum_{j=2}^{m}\{(j-1)\cdot d\})\cdot(c-1)$ (relaxed). The mutual information bound can be derived as below:

$$[\frac{m-1}{(m-1)\cdot c\cdot d - R_{max}} \cdot \log\frac{c(m-1)}{(m-1)\cdot c\cdot d - R_{max}}]'$$
$$= (\log\frac{m-1}{(m-1)\cdot c\cdot d - R_{max}} + \log c + 1) \cdot \frac{(m-1)\cdot R'_{max} - R_{max}}{[(m-1)\cdot c\cdot d - R_{max}]^2}$$

Due to $R_{max} = (\sum_{j=2}^{m}(j-1)\cdot d)\cdot(c-1)$, we have:

$$R_{max} = (c-1)\cdot d\cdot\frac{m^2-m}{2},\ R'_{max} = (c-1)\cdot d\cdot(m-\frac{1}{2}) \quad (8)$$

Then, we replace the derivative of mutual information with $R_{max}$ and $R'_{max}$. Since $(m-1)\cdot(R'_{max}) < R_{max}$, the second part of the derivative cannot be 0. Thus, $m$ is optimal when $\log\frac{m-1}{(m-1)\cdot c\cdot d - R_{max}} + \log c + 1 = 0$, and we have $m = \frac{2\cdot(c\cdot d - e^{\log c + 1})}{(c-1)\cdot d}$. □

## B PRIVACY AND UTILITY ANALYSIS

### B.1 Proof of Theorem 3.3 (Privacy Analysis)

Proof. For any pair of input locations $x, x' \in \mathcal{D}$ and output $y$, the maximum perturbation probability $q(y|x)$ for sampling location $y$ based on input $x$ is $\alpha_{max}(x)$ when $y$ in the same group with $x$ (the first group $G_1(x)$); the minimum perturbation probability $q(y|x')$ for sampling location $y$ based on input $x'$ is $\alpha_{min}(x')$ when $y$ in the furthest group for $x'$ (the last group $G_m(x')$). Thus, the SRR mechanism in PrivLBS satisfies $\epsilon = \max_{x,x'\in\mathcal{D}} \log(c \cdot \frac{(m-1)d\cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1)\cdot|G_j(x)|]}{(m-1)d\cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1)\cdot|G_j(x')|]})$-LDP in all the cases, where $\epsilon$ is a strict constant privacy bound derived by $c$ and domain $\mathcal{D}$. □

### B.2 Proof of Theorem 3.6 ($L_2$ Error Bound)

Proof. With the estimation formula, we have:

$$p(C_x) = p(x)\cdot\sum_{y\in C_x} q(y|x) + \sum_{x'\neq x} p(x')\cdot[\sum_{y\in C_x\backslash C_{x'}} q(y|x')$$
$$+ \sum_{y\in C_x\cap C_{x'}} q(y|x')]$$

With the property of Hadamard matrix [36], the size of the set difference between any two location candidate sets is $\frac{d}{4}$, and the size of intersection between any two candidate sets of locations is also $\frac{d}{4}$. We can integrate these into the equation. Then, we have:

$$p(C_x) \geq p(x)\cdot[\sum_{y\in C_x} q(y|x)] + \sum_{x'\neq x} p(x')\cdot\frac{d\cdot\min\{q(y|x')\}}{2}$$
$$= p(x)\cdot[\sum_{y\in C_x} q(y|x)] + [1-p(x)]\cdot\frac{d\cdot\alpha_{min}(x)}{2}$$
$$\implies p(x) \leq \frac{q(C_x) - \frac{d\cdot\alpha_{min}(x)}{2}}{\sum_{y\in C_x}[q(y|x) - \frac{d\cdot\alpha_{min}(x)}{2}]}$$

Then, we can have the $L_2^2$-distance as below:

$$\mathbb{E}[L_2^2(\tilde{p},p)] \leq (\frac{1}{\sum_{y\in C_x} q(y|x) - \frac{d\cdot\mu}{2}})^2 \cdot \mathbb{E}[L_2^2(\tilde{p}(C),p(C)]$$

where $\mu = \min\{\alpha_{min}(x)\}$. Since $\mathbb{E}[\tilde{p}(C_x)] = \mathbb{E}[\frac{\mathbb{I}\{y_j\in C_x}{n}\}]$ $= p(C_x)$, we have:

$$\mathbb{E}[L_2^2(\tilde{p}(C),p(C))] = \mathbb{E}[\sum_{x\in\mathcal{D}}(\tilde{p}(C_x) - p(C_x))^2] = \sum_{x\in\mathcal{D}} Var(\tilde{p}(C_x))$$

Moreover, each $y$ is independently sampled and $\tilde{p}(C_x) = p(C_x)$ is the mean of $n$ independent multinomial distributions.

$$\sum_{x\in\mathcal{D}} Var(\tilde{p}(C_x)) \leq \sum_{x\in\mathcal{D}}\frac{1}{n}\cdot\max\{p(C_x)\} \leq \frac{d}{n}$$

Thus, we have $\mathbb{E}[L_2(\tilde{p},p)] \leq (\frac{1}{\sum_{y\in C_x} q(y|x) - \frac{d\cdot\mu}{2}})\cdot\sqrt{\frac{d}{n}}$. □

### B.3 Proof of Theorem 3.5 ($L_1$ Error Bound)

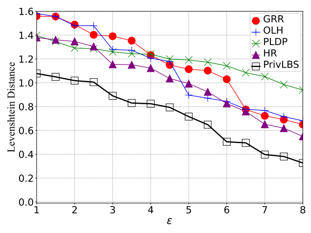
Proof. Since $\forall i, a_i > 0$, $n\cdot\sum_{i=1}^{n}(a_n)^2 \geq [\sum_{i=1}^{n}(a_n)]^2$ holds, we have $d\cdot L_2^2(\tilde{p},p) \geq [L_1(\tilde{p},p)]^2$. Then. we can derive:

$$(\mathbb{E}[L_1(\tilde{p},p)])^2 \leq \frac{d^2}{n\cdot(\gamma - \frac{d\cdot\mu}{2})^2}$$

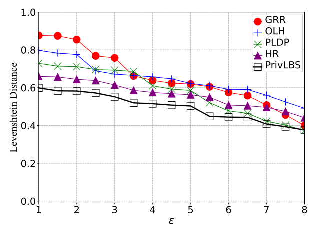
Thus, $\mathbb{E}[L_1(\tilde{p},p)] \leq \frac{2d}{\sqrt{n}\cdot(2\gamma - d\cdot\mu)}$ completes the proof. □

## C ADDITIONAL EXPERIMENTS

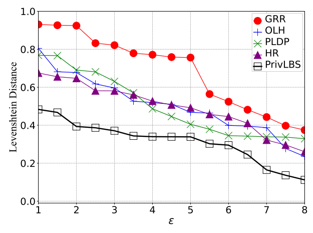
**Traffic-Aware GPS Navigation**. We first simulate many trajectories and predict the time $Agg^p(x_1,x_2)$ between any two locations on the trajectory using the Markov Chain [64]. Specifically, we generate multiple routes for each OD pair (at client). For each route, we compute the predicted time $t$ based on historical datasets for any two locations. In our experiment, we use the data collected earlier as the historical data (e.g., Geolife dataset collected in 2009 as the historical data, and collected in 2010 as the test data). Furthermore, for locations on each route, such LBS calculates the frequencies of users near the location within a range (e.g., 4.7m). If the frequency exceeds a threshold (e.g., 50 users), a 3-second delay time will be added [22]. Finally, given the traffic density, it may update the route to avoid heavy traffic. With PrivLBS, the route recalculation occurs if $Agg^t(x_o,x_i) - Agg^p(x_o,x_i) > \theta$ holds, where $i\in\mathbb{T}$ and $\theta$ is the delay threshold (e.g., 30 seconds). If yes, the client will submit



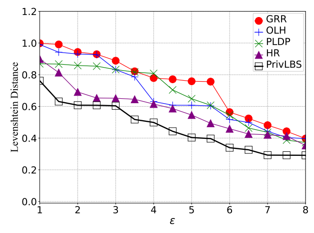
(a) Gowalla



(b) Geolife

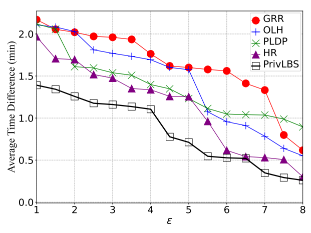


(c) Portocabs

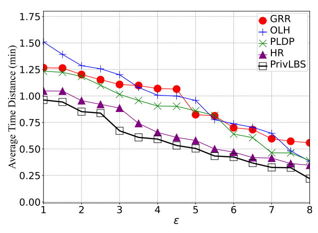


(d) Foursquare

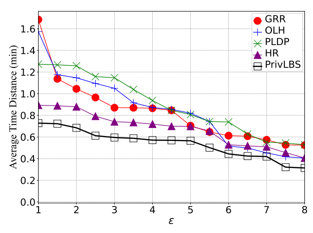
**Figure 14: Relative levenshtein distance of trajectories in the traffic-aware GPS navigation ($\theta$ = 40 seconds)**



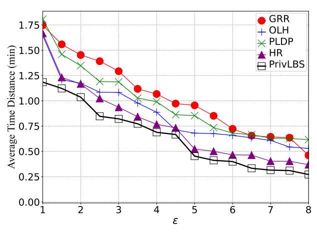
(a) Gowalla



(b) Geolife

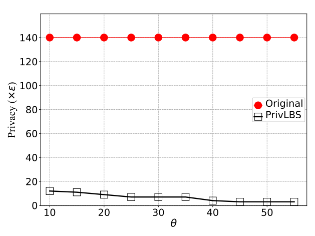


(c) Portocabs

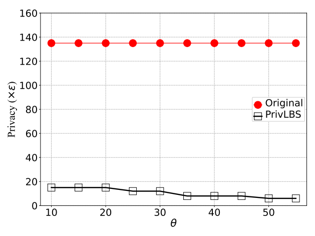


(d) Foursquare

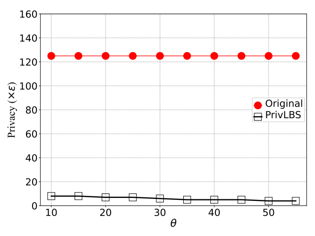
**Figure 15: Average trip time deviation in the traffic-aware GPS navigation ($\theta$ = 40 seconds)**



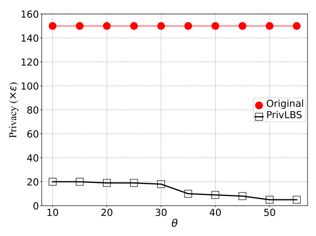
(a) Gowalla



(b) Geolife



(c) Portocabs



(d) Foursquare

**Figure 16: The total privacy bound of PrivLBS for traffic-aware GPS navigation by collecting trajectories**

**Table 4: Average $L_1$-distance for the location distribution on four datasets using Laplace mechanism for centralized DP**

| Dataset | $\epsilon = 1$ | $\epsilon = 3$ | $\epsilon = 5$ | $\epsilon = 7$ |
|---|---|---|---|---|
| Gowalla | 0.18 | 0.16 | 0.15 | 0.13 |
| Geolife | 0.29 | 0.11 | 0.08 | 0.04 |
| Portocabs | 0.43 | 0.26 | 0.15 | 0.07 |
| Foursquare | 13.87 | 4.69 | 2.78 | 1.91 |

its perturbed location, and privately retrieve the real-time traffic density at the current position to recalculate the fastest route [22].

First, we evaluate how the delay time threshold $\theta$ affects the total privacy guarantee. The maximum numbers of locations on the trajectories for four datasets are 140, 135, 127, and 150, respectively. In Figure 16, we set $\theta$ between 10 seconds and 55 seconds with a step of 5 seconds. As $\theta$ increases, the total privacy bound $\epsilon$ decreases with a decreasing number of location updates. As $\theta$ =60 (updating the location once delay exceeds 1 minute), the privacy bound is around $3\epsilon$, which is very small for trajectories.

Second, to measure the route deviation, we apply Levenshtein distance to measure the accuracy between true route and the route recommended by PrivLBS. It measures the difference by calculating the minimum number of location edits (insertions, deletions, or substitutions) required to change one route to the other. Figure 14 shows the relative Levenshtein distance over the total size of the true routes (vs the total privacy bound $\epsilon$). PrivLBS again outperforms other LDP schemes. In addition, we also measure the deviation of the total trip time. Figure 15 shows that the trip time deviation decreases as the privacy bound $\epsilon$ increases for all the LDP schemes, and PrivLBS results in the least trip time deviation.

**SRR and Differential Privacy**. Finally, we discuss the utility of centralized differential privacy. A generic solution is to add the Laplace noise to the frequency of each location (after aggregation). Thus, $\epsilon$ should be equally allocated for $d$ locations. Table 4 presents the $L_1$-distance for the distribution on four datasets using Laplace mechanism. The results show that the $L_1$ distance gets smaller as $\epsilon$ becomes larger. Compared to the LDP mechanism, for Gowalla and Foursquare, the distance with SRR has smaller distance. For Geolife and Portocabs, the distance with SRR has similar distance in case of a smaller domain. Note that the privacy guarantees of LDP and DP are indeed incomparable even for the same $\epsilon$ (since the trust model and indistinguishability are defined in different ways).