Location Privacy and its Degradation Under Continual Observation

David Hooton

August 2019

Abstract

Geospatial data and location-based services are growing ever more ubiquitous, as are calls for increased control over this data. Many attempts have been made to develop mechanisms that can obscure individuals' locations. Several of these follow the principles of differential privacy, a mathematically rigorous privacy guarantee that ensures the privacy of the data is preserved no matter what side information an adversary could use to reconstruct it.

In this work we seek to demonstrate how existing privacy measures for location data worsen under repeated or continual observation, attempting to create an implementation of a location tracking attack that uses these flaws to pinpoint users' locations. We focus on the mechanism of geo-indistinguishability, and provide empirical results for the effectiveness of our algorithm. We then suggest an alternative privacy-preserving location sharing mechanism that is not susceptible to such an attack.

1 Introduction to Differential Privacy

Differential privacy is first and foremost a measure to assess and curtail the privacy infringement that data could cause to an individual, with the goal that individuals can safely give their information to a database or service without their true data being revealed to any untrusted parties. This practice is becoming more widely used; it has been deployed by Apple and Google, and will be used in the upcoming 2020 US Census.

It is formalised with the model of a user who wants to add their data to a database without an adversary being able to query this database in such a way that they can obtain the users information. Adding random noise to the data, or returning randomised results to queries about the data, allows differential privacy to go beyond established measures of data anonymisation and de- identification. With these, an adversary with side information can use this to reconstruct the user's private data or link it to their identity. The randomness has to be such that, with a high probability, the queries the adversary makes to the database would have similar results whether or not the user's data were included, thereby ensuring the adversary has found no new information about the user.

Consider an adversary who wants the answer to some query q of a database we control. We consider a database $D \in \mathbb{R}^{n,k}$ as a matrix, where each row contains some individual's data or records, and each column is a particular attribute of this data. We do not directly return the answer, but rather the output of an approximation to q, a randomised *Database Access Mechanism* $\mathcal{M} : \mathbb{R}^{n,k} \to \mathbb{R}$ (the values need not be real but we consider them as such for simplicity). No matter the potential output of $\mathcal{M}$, the inclusion of any particular record in the dataset should not have a significant effect, which we control with a parameter ε, often called the *Privacy Budget.*

ε allows us to control the trade-off between privacy and accuracy. For our purposes, a smaller value of ε means more noise is added, increasing the privacy protection on the data and reducing its utility. Setting the value for this privacy budget is a subjective issue, with best common practices not yet agreed upon.

Given D, let D' be a *neighbour* of this database, meaning it has been obtained by changing the data for at most one individual. Specifically the Hamming distance between the two databases is at most 1 (at most one row differs between the databases): $\|D - D'\|_1 \leq 1$. By guaranteeing similar outputs for neighbouring databases, we guarantee that no individual has a significant impact on the output of $\mathcal{M}$.

Definition 1.1. Differential Privacy [4]
A database access mechanism $\mathcal{M}$ satisfies ε-Differential Privacy, or ε-*DP*, if $\forall\, x \in \text{range}(\mathcal{M})$, $D, D' \in \mathbb{R}^{n,k} : \|D - D'\|_1 \leq 1$,

$$\mathbb{P}\big(\mathcal{M}(D) = x\big) \leq e^{\varepsilon}\mathbb{P}\big(\mathcal{M}(D') = x\big) \tag{1}$$

2 Private location sharing

We now focus on data privacy in the context of location sharing, and of a user who wishes to share their location for some service. If we consider the problem of privately revealing a single point in the Cartesian plane, the standard definition of ε-*DP* does not apply since there are no databases to consider. Several different definitions have appeared to try and extend the notion of ε-*DP*. Here $\mathcal{M} : \mathbb{R}^2 \to \mathbb{R}^2$ is a location hiding mechanism that takes an individual's true location and returns some random location.

One of these is geo-indistinguishability [1], introduced by Andres et al., which proposes that for nearby points the values returned by the mechanism $\mathcal{M}$ should be similar. This reflects the ideas of differential privacy, which ensured that neighbouring databases should give similar results.

Let $\tilde{\mathcal{M}}(x)$ be the distribution of outcomes of $\mathcal{M}$ for some input x. We briefly introduce the *multiplicative distance* between two distributions σ, μ on $\mathbb{R}^2$, defined to be

$$d_{\times}(\sigma, \mu) = \sup_{A \subseteq \mathbb{R}^2} \left| ln\left(\frac{\sigma(A)}{\mu(A)}\right)\right|$$

Definition 2.1. Geo-indistinguishability
A mechanism $\mathcal{M}$ satisfies ε-geo-indistinguishability if $\forall x, y \in \mathbb{R}^2$,

$$d_{\times}\left(\tilde{M}(x), \tilde{M}(y)\right) \leq \varepsilon d(x, y) \tag{2}$$

Where $d(x, y)$ refers to the standard Euclidean distance between $x, y \in \mathbb{R}^2$.
Equivalently, the definition is satisfied if $\forall A \subseteq \mathbb{R}^2$, $\tilde{M}(x)(A) \leq e^{\varepsilon d(x,y)}\tilde{M}(y)(A)$.

The paper proposes a mechanism to satisfy this definition by adding noise to the true location x from what we will call the *Polar Laplacian* distribution on $[0, \infty) \times [0, 2\pi)$ with density function $f(r, \theta) = \frac{\varepsilon^2}{2\pi}e^{-\varepsilon r}$, where ε is the preset privacy parameter. The parameter ε is directly proportional to the mean radius from this distribution. We can sample from this distribution choosing the angle θ uniformly at random and the radius r from the appropriate Gamma distribution, and can compute these as follows:

Algorithm 1: Polar Laplacian mechanism $\mathcal{M}^{PL}$ for geo-indistinguishability [1]

```
input  : $(x_1, x_2) \in \mathbb{R}^2$
output : A randomised $(\tilde{x}_1, \tilde{x}_2) \in \mathbb{R}^2$

Sample $\theta$ uniformly from $[0, 2\pi)$
Sample $q$ uniformly from $[0, 1)$
Set $r = -\frac{1}{\varepsilon}\big(W_{-1}\big(\frac{q-1}{\varepsilon}\big) + 1\big)$
/* $W_{-1}$ is the $-1$ branch of the Lambert $W$ function                */
return $(x_1 + r\cos(\theta), x_2 + r\sin(\theta))$
```

3 Continual location sharing

All differentially private mechanisms see some form of privacy degradation under repeated observation. If an adversary can query the same real value with an ε-DP mechanism n times, the privacy guarantee will reduce to $n\varepsilon$-DP, as by calculating the mean value the adversary can effectively reduce the noise added to the data. The same holds for n repeated queries of the same position under geo-indistinguishability - the privacy guarantee is weakened from ε to $n\varepsilon$.

This does not hold if the adversary is querying independent values: the output from one query does not give away any new knowledge about the other true values. Many differentially private mechanisms only consider a one-shot computation, including geo-indistinguishability.

In the case of location data, it is frequently useful to track a user's location path over time, either for mapping information or simply repeated use of a service. These change over time but are not independent. Cao et al. [3] have studied the degree to which privacy degrades over time if there are known temporal correlations in the data.

Xiao et al. [7] propose a pseudo-differentially private mechanism for releasing location data under temporal correlations, however it only ensures that at each point the location remains private. This involves continually updating a set of points where the user is likely to be (a δ-location set), adding an increasing amount of noise over time and thereby decreasing utility.

In addition to this, many location privacy mechanisms such as the original geo-indistinguishability mechanism add real-valued random noise, which may ignore additional side information an adversary could have about the geography of the user's area.

Location tracking attacks have been presented in Shokri et al. [6], however these do not focus on geo-indistinguishability or other pseudo-differentially private mechanisms.

4 Methodology

We now set out our approach to building a location tracking attack on geo-indistinguishable data, and then discuss possible solutions to the problem.

We consider a user's movement as a time-homogenous Markov process $X = \{X_n\}_{n=1,\ldots,N}$, with the discrete states L corresponding to locations in the Cartesian plane. This allows us to consider any location $l \in L$ as either a single state or as coordinates in $\mathbb{R}^2$, and we will frequently interchange both representations. Let $S \in [0,1]^{|L|\times|L|}$ be the transition matrix of this Markov chain. The entries are defined as $S_{x,y} = \mathbb{P}(X_1 = y \mid X_0 = x)$. We will often refer to the states in L as having edges or degree, meaning the edges implied by positive entries of S.

Define the *path space* to be $\mathcal{P} = \{(l_1, \dots, l_N) \in L^N : \forall 1 \leq i < N,\ S_{l_i,l_{i+1}} > 0\}$, the set of all valid paths of length N in the Markov process. Each $\omega \in \mathcal{P}$ has a naturally associated path probability $\mathbb{P}(\omega) = \mathbb{P}(l_0) \prod_{1 \leq i < N} S_{l_i,l_{i+1}}$. $\mathbb{P}(l_0)$ is the probability that the chain starts at l_0, which may be given by an initial distribution. Throughout this paper this distribution is a uniform distribution on L, however additional information about these probabilities does not affect the calculations, and useful information would only improve the performance of the deobscuring mechanisms.

We consider each user's true path to be some $\psi^* = (l_1^*, \dots, l_N^*) \in \mathcal{P}$. The locations on this path are revealed by some privacy preserving mechanism $\mathcal{M}$ to create the user's hidden path $\psi^H = \mathcal{M}(\psi^*) = (l_1^H, \dots, l_N^H)$, which is the information available to an adversary. Note that this path does not necessarily have to be valid according to the Markov model, each location l_i^H could be any point in $\mathbb{R}^2$.

The aim is to take the role of the adversary and construct a location tracking attack $\mathcal{D} : (\mathbb{R}^2)^N \to \mathcal{P}$ that exploits weaknesses in $\mathcal{M}$ in order to reconstruct a similar path to ψ^*, which we call the approximate path $\psi^A = \mathcal{D}(\psi^H) = (l_1^A, \dots, l_N^A)$. We want this approximation to be the $\omega \in \mathcal{P}$ that maximises $\mathbb{P}\big(\psi^* = \omega \,|\, \mathcal{M}(\psi^*) = \psi^H\big)$, the objective function in our path search. We call this $\mathcal{D}$ a *deobscuring mechanism.*

We can frame the problem as a Bayesian updating of our beliefs given the new information of the hidden path:

$$\begin{aligned}\mathbb{P}\big(\psi^* = \omega \,|\, \mathcal{M}(\psi^*) = \psi^H\big) &= \frac{\mathbb{P}\big(\mathcal{M}(\psi^*) = \psi^H \,|\, \psi^* = \omega\big)\mathbb{P}\big(\psi^* = \omega\big)}{\mathbb{P}\big(\mathcal{M}(\psi^*) = \psi^H\big)} \\ &\propto \mathbb{P}\big(\mathcal{M}(\omega) = \psi^H\big)\mathbb{P}\big(\psi^* = \omega\big)\end{aligned} \tag{3}$$

$\mathbb{P}(\psi^* = \omega)$ represents our prior beliefs on the distribution of the true path. Since the user in our model is assumed to be moving according to a Markov process, we should initially weight each ω according to its path probability, setting $\mathbb{P}(\psi^* = \omega) = \mathbb{P}(\omega)$.

4.1 Deobscuring geo-indistinguishability

We now apply this theory to a path whose points have been perturbed such that they all obey geo-indistinguishability.

Firstly, we focus on the geo-indistinguishable Polar Laplacian mechanism $\mathcal{M}^{PL}$ seen in Algorithm 1, applied independently to each location in the path. We write $\mathcal{M}^{PL}(\psi) = (\mathcal{M}_1^{PL}(l_1), \dots, \mathcal{M}_N^{PL}(l_N))$, where each $\mathcal{M}_i^{PL}$ is an independent instance of the Polar Laplacian mechanism.

Specifically, if we know the parameter ε then we know the distribution of the polar Laplacian distribution, so we can calculate the probability of revealing the hidden path from any ω, as in equation (3):

$$\mathbb{P}\big(\mathcal{M}^{PL}(\omega) = \psi^H\big) = \prod_{i=1}^{N} \mathbb{P}\big(\mathcal{M}^{PL}(l_i) = l_i^H\big) = \prod_{i=1}^{N} \frac{\varepsilon^2}{2\pi} e^{-\varepsilon d(l_i, l_i^H)}$$

Intuitively, this gives some measure of how close ω is to ψ^H. Together with the path probabilities, this allows us to calculate $\mathbb{P}\big(\psi^* = \omega \,|\, \mathcal{M}(\psi^*) = \psi^H\big)$ for every ω. This probability calculation has time complexity $O(N)$.

At this point, we could take $\psi^A = \operatorname{argmax}_{\omega\in\mathcal{P}} \mathbb{P}\big(\psi^* = \omega \,|\, \mathcal{M}(\psi^*) = \psi^H\big)$. The problem with this approach is that the size of the path space $\mathcal{P}$ grows exponentially with the length of the path; if D be the maximum degree of any state in L we have that $|\mathcal{P}| \in O(|L|D^N)$. Checking all of these paths is computationally infeasible, so a way to narrow down the search space is required.

Since the Polar Laplacian distribution has mean 0, it is likely that the hidden path is in some sense close to the true path, with the mean position of the points in the hidden path in a similar place to the mean position of the true path. Therefore, by searching only those paths which are sufficiently close to the hidden path, we should hope to find an adequate approximation to the true path.

To find this neigbourhood of close paths we use the Levenshtein distance, commonly used for approximate string matching in computer science. The Levenshtein distance d_{Lev} between two strings is k if the minimal number of character insertions, deletions, and substitutions needed to transform one string into the other is k. We can associate a unique unicode character to each $l \in L$ in order to use existing work on approximate string matching. This also requires us to snap each location in ψ^H to the nearest point in L to get a starting string $\overline{\psi}^H$.

Specifically, we use an existing implementation of a *Levenshtein automata* [5] which can efficiently generate $\mathcal{P}_K$, the set all paths within a given Levenshtein distance of the starting path. This constitutes the search space of our algorithm.

We can now fully specify the deobscuring mechanism $\mathcal{D}_0$:

Algorithm 2: Deobscuring mechanism $\mathcal{D}_0$ for paths revealed by $\mathcal{M}^{PL}$

```
input  : $\psi^H = \mathcal{M}^{PL}(\psi^*) \in (\mathbb{R}^2)^N$, $L$, parameters used in $\mathcal{M}^{PL}$
output : An approximation $\psi^A$ to $\psi^*$

Obtain $\overline{\psi}^H$ by snapping $\psi^H$ to the coordinate grid given by $L$
Let $\mathcal{P}_K = \{\omega \in \mathcal{P} : d_{Lev}(\omega, \overline{\psi}^H) \le K\}$
return $\operatorname{argmax}_{\omega\in\mathcal{P}_K} \mathbb{P}\big(\psi^* = \omega \,|\, \mathcal{M}(\psi^*) = \psi^H\big)$
```

An example run of $\mathcal{M}^{PL}$ and $\mathcal{D}_0$ can be seen in Figure 1. Some features of the deobscuring mechanism can be seen, for instance the Levenshtein automata approach allows $\mathcal{D}_0$ to ignore the outlying points with large amounts of noise added. In this experiment the average distance from the user's true locations to the revealed locations was reduced from 2.09 to 0.42 by the deobscuring mechanism.

An alternative geo-indistinguishability preserving mechanism $\mathcal{M}^{PL*}$ that would be less susceptible to this attack would be to generate the random noise once and apply this equally to each point in ψ^*. This would ensure that the mean revealed location is substantially different to the mean true location, and that the true path is a large Levenshtein distance away from the revealed path.

However, this preserves the *shape* of the original path, and so an alternative approach can be used to deobscure ψ^H, which we call $\mathcal{D}_1$. This will be potentially less effective than $\mathcal{D}_0$, but it can still target a vulnerability in adding equal or highly correlated noise to the points in the

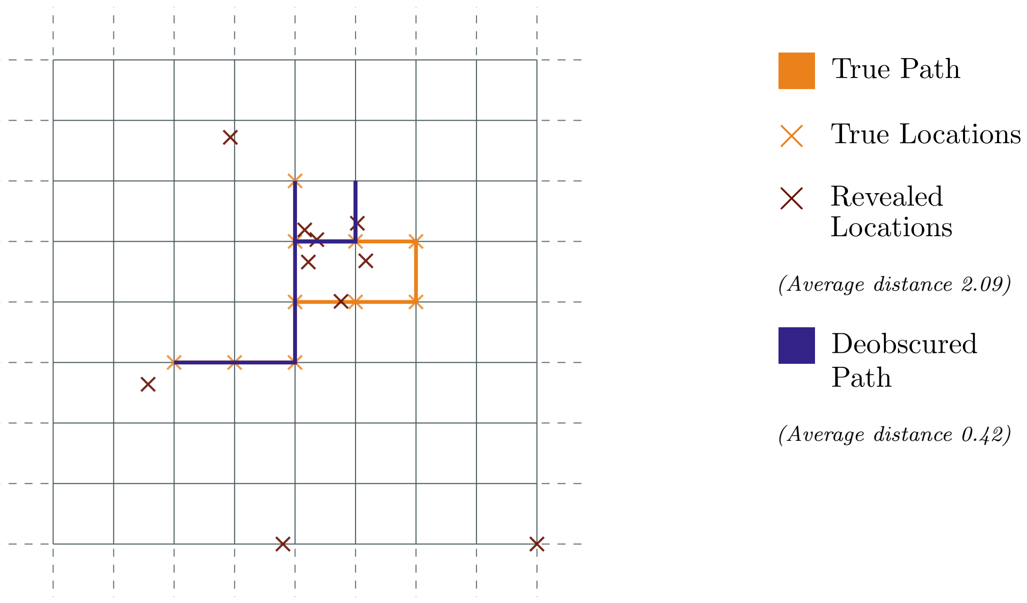


Figure 1: Example run of $\mathcal{M}^{PL}$ and $\mathcal{D}_0$

path. For instance, if a user's path revealed them to be moving due East for an extended period of time, an adversary could search their area for roads where this would be possible, quickly restricting the set of locations where the user could be.

The only candidate paths we have to consider can be generated as $\psi^H + \gamma$, where $\gamma \in \mathbb{R}^2$ is some offset to be added to each point in the path. There can only be as many offsets that give valid paths as there are points in L, so only $|L|$ paths need to be considered as candidates.

When performing the probability calculations for each ω we should only calculate $\mathbb{P}\big(\mathcal{M}(\omega) = \psi^H\big)$ as $\mathbb{P}(\mathcal{M}^{PL}(l_1) = l_1^H)$. The path probabilities still need to be calculated, giving a total complexity of $O(N|L|)$ for this deobscuring mechanism.

To mitigate this issue, $\mathcal{M}^{PL*}$ could be improved further by adding another round of smaller noise to each individual location on the path. The location sharing mechanism we propose, $\mathcal{M}^{EPL}$, works on the same principles and can be seen in Algorithm 3. We proceed as in $\mathcal{M}^{PL*}$ by adding the same noise to every point, then search nearby paths (using the Levenshtein distance) to find the one with highest path probability. This protects against $\mathcal{D}_0$ by adding correlated noise, and protects against $\mathcal{D}_1$ by changing the shape of the path. By using the Levenshtein approach we ensure valid path is revealed.

While this algorithm should be less vulnerable to the attacks we described, a more rigorous statistical framework is needed to confirm this.

An implementation of the techniques and algorithms from this section is now given, which allows us to determine their effectiveness on tangible test cases.

5 Experimental setup

In order to empirically assess the effectiveness of the obscuring and deobscuring mechanisms, simulation experiments were performed in Python.

Algorithm 3: Enhanced Polar Laplacian mechanism $\mathcal{M}^{EPL}$ for path sharing

```
input  : $\psi^* \in \mathcal{P}$, $k \in \mathbb{N}$
output : $\psi^H \in \mathcal{P}$

Sample $\theta$ uniformly from $[0, 2\pi)$
Sample $q$ uniformly from $[0, 1)$
Set $r = -\frac{1}{\varepsilon}\big(W_{-1}\big(\frac{q-1}{\varepsilon}\big) + 1\big)$
foreach $l_i^* \in \psi^*$ do $\xi_i = l_i^* + (r\cos\theta, r\sin\theta)$;
Let $\mathcal{P}_k = \{\omega \in \mathcal{P} : d_{Lev}(\omega, \xi) \leq k\}$
return $\text{argmax}_{\omega \in \mathcal{P}_k} \mathbb{P}(\omega)$
```

Three different location systems are used for the Markov models. Two of these are 10×10 square lattices with positive transition probabilities only between directly neighbouring edges. The last is a 64×64 square lattice created from a dataset of mobility traces for taxis from Rome [2], where the edge probabilities represent the movement of real users throughout the city.

For reasons that will be discussed in the limitations section of this report, results took a long time to collect, and so they are not as complete as planned, especially for the Rome data.

These location systems are created as rates matrices for continuous time Markov processes, from which jump matrices and graph representations (as edge lists) are constructed. This was done such that the system is general enough to facilitate possible future extensions to a model where locations are sampled at arbitrary times from a continuous time Markov model. For now a discrete Markov model is used with the derived jump matrices as the transition matrices.

Because of this, these location systems will be referred to by their rates matrices Q_0, Q_1, and Q_{Rome}[†].

Q_0 consists of 100 states arranged in a 10×10 grid where there is an edge with rate 1 between each neighbouring state. Equivalently, this models a random walk through the underlying graph. In Q_1, the same edges are present, however any horizontal edges have rate 2 whereas the vertical edges have rate 1. This is a simple dummy case to see how the mechanisms respond to unequal edge weights, modelling a situation where some information has been gathered on the movement patterns of users within this system. Both of these have symmetrical rates.

Due to the geometry of these systems, paths are more likely to travel along the edges of the grid. This does not affect the performance of any of the algorithms but it makes it harder to compare their behaviour, and so the start states of the paths are drawn randomly from those more than 3 edges from the exterior of the grid.

In the case of Q_{Rome}, the original data contained timestamped geospatial data from over 300 taxis. The locations were all snapped to the nearest point on a 64×64 coordinate grid, and locations outside of this were removed. Each square in this grid has length $0.0025'$ in both latitude and longitude, which is roughly 350m at this latitude. The time attribute was downsampled from every 7 seconds to approximately every minute, to get a larger range of movement between each timestep. The transitions between locations in consecutive minutes for each taxi were aggregated to compute the Markov rates matrix of this system.

[†]In the code these are named `Q6`, `Q7`, and `QRome`.

$$\begin{pmatrix}
-3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \\
2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \\
0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots \\
0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -5 & 2 & 0 & 0 & 0 & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -3 & 0 & 0 & 0 & \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 2 & 0 & \cdots \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -6 & 2 & \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -6 & \\
 & & \vdots & & & & \vdots & & & & \vdots & & & \ddots
\end{pmatrix}$$

Figure 2: Partial view of Q_1

Q_1 can be seen in Figure 2, and the graphical representations of Q_1 and Q_{Rome} can be see seen in Figures 3 and 4. The edge labels on the graphs are the transition rates at which the Markov model moves along each edge; for Q_1 these are symmetric.

Q_{Rome} is too large to display, so Figure 4 displays a small subgraph. Many nodes actually have large degree in the full graph, where it is possible to move from one node to a non-neighbouring one in the short distance between measurements.

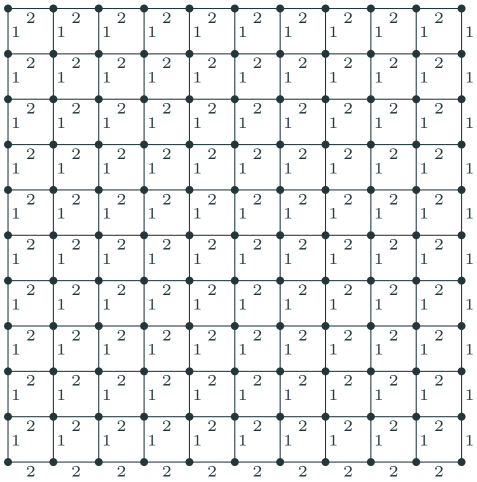


Figure 3: Graphical representation of Q_1

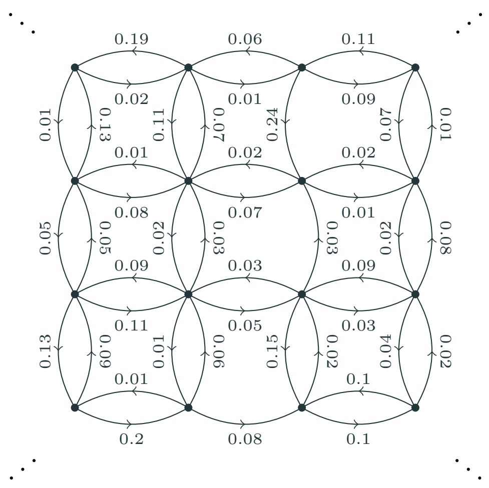


Figure 4: Partial graphical representation of Q_{Rome}

The experiment was performed by first generating 3 paths using the Markov model for each path length. We then apply each obscuring mechanism independently 3 times for each value of ε, and then apply $\mathcal{D}_0$ to these to get deobscured paths. For each combination of path length and ε, there are therefore 3 repeats of the test.

We will assess the effectiveness of the deobscuring mechanism using the *distance ratio*. This is calculated by dividing the average distance of points in the hidden path from the true path by the average distance of points in the deobscured path from the true path:

$$\textit{Distance ratio} = \frac{1/N \sum_{i=1}^{N} d\big(\psi_i^H, \psi_i^*\big)}{1/N \sum_{i=1}^{N} d\big(\mathcal{D}_0(\psi^H)_i, \psi_i^*\big)}$$

Since the privacy parameter ε in geo-indistinguishability is directly proportional to the mean radius of the generated noise this provides a good measure of the privacy degradation caused by deobscuring the path.

6 Results

In Figures 5 and 6 we assess the performance of $\mathcal{D}_0$ under changes to ε, the path length and the weights in the graph. The error bars signify the maximum and minimum results for each experiment. As one would expect, increasing the path length provides more information to the deobscuring mechanism and so makes it more effective; however the increases seen in 5b and 6b are perhaps less than expected. $\mathcal{D}_0$ is also more effective on Q_1 than Q_0, also to be expected as the unequal weights provide more information to the mechanism. Over all experiments run on these location systems, the average distance ratio is 1.26 for Q_0 and 1.28 for Q_1. Larger values of ε mean noise of smaller magnitude is added to the path, and $\mathcal{D}_0$ is more likely to find something very close to the true path, leading to the high distance ratios seen.

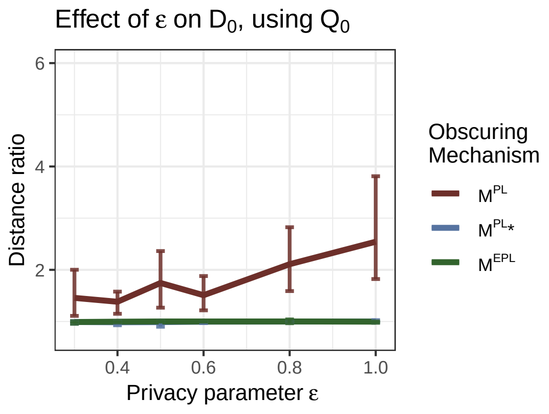
We also see that, as intended, the privacy guarantees given by $\mathcal{M}^{PL*}$ and $\mathcal{M}^{EPL}$ are unaffected by $\mathcal{D}_0$.

Figure 7 displays the performance of $\mathcal{D}_0$ on Q_{Rome}. We see similar overall trends in performance as the parameters vary, although with greater variance. Comparing results for path length 6, the only value in common between all the datasets, we find a mean distance ratio of 1.16 on Q_{Rome}, compared to 1.17 for Q_1 and 1.23 for Q_0.

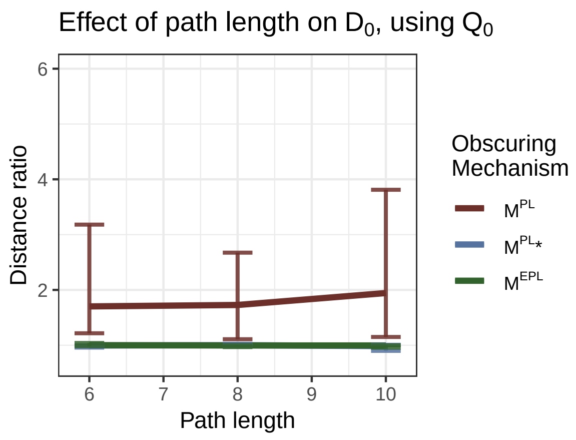
While the results are not conclusive, they do show that adding correlated noise to the data as in M^{EPL} can effectively prevent some location tracking attacks. Further work is also needed to ensure a theoretical privacy guarantee for this mechanism, but the techniques developed should have ensured that an adversary cannot generate an approximate path closer the the true path than what we have revealed.

7 Limitations

There are several issues with this work in its present state that prevent it from being practicable. Crucially, the implementation of the Levenshtein automata currently in place is not as efficient as it was hoped. It has to traverse an implicitly created trie of all possible words in L^N, searching for words that are valid. Checking whether the word is a valid path and is within the required distance is $O(N)$ (where k is the maximum allowed edit distance). Many nodes of the trie are in

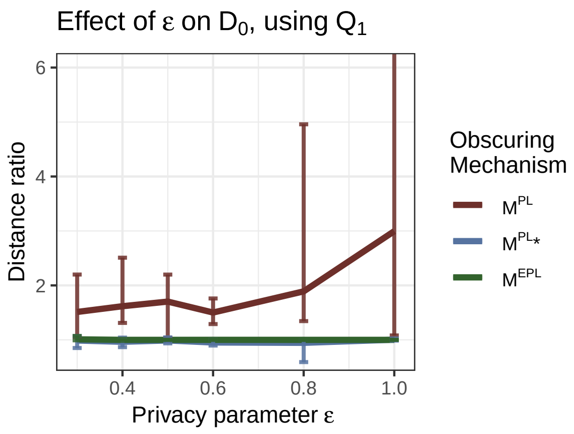


(a)

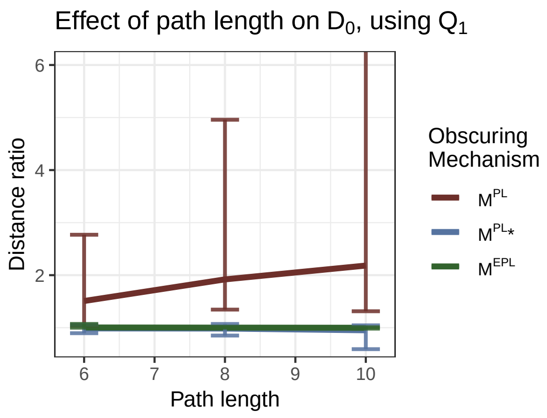


(b)

Figure 5: Effect of parameters on performance of $\mathcal{D}_0$, using Q_0

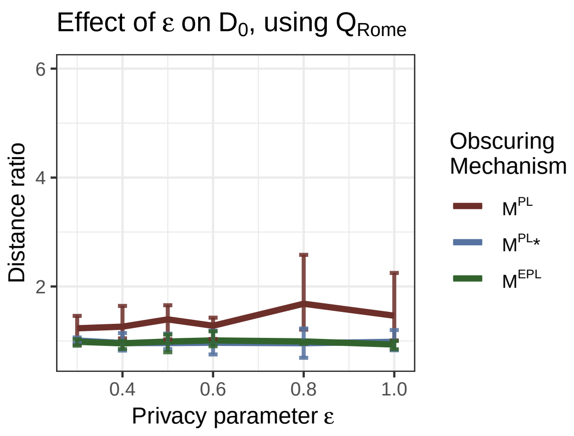


(a)

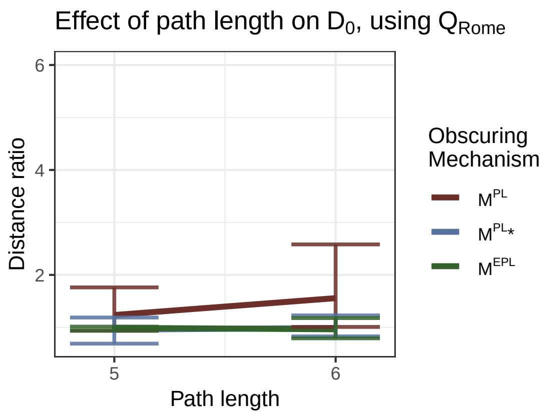


(b)

Figure 6: Effect of parameters on performance of $\mathcal{D}_0$, using Q_1



(a)



(b)

Figure 7: Effect of parameters on performance of $\mathcal{D}_0$, using Q_{Rome}

practice 'pruned' and do not need to be checked, but the runtime of the algorithm $\mathcal{D}_0$ is still $O(N|L|^N)$ in the worst case. This complexity should only be approached when k approaches the length of the input word, but the exponential complexity already makes the algorithm impractically slow. This was especially true for experiments on Q_{Rome}, where there were not only more states but the maximum degree of a node was 13 rather than 4.

This also applies to $\mathcal{M}^{EPL}$, however since the value of k can be significantly smaller this algorithm is far more usable.

In order to fix this issue, we propose a potential alternative method to enumerate nearby paths to an input ω. Far simpler, we can simply fix k and consider paths with k points changed, of which there are $O\big(\binom{N}{k}D^k\big)$ candidate paths. For a quick fix in the experiments, this approach was taken by constructing a Levenshtein automata that only recognised character substitutions as allowed operations, but the runtime was still impractically slow, with code running on a remote server with reasonable computational power. Too little data could be collected to account for the natural variance in results, making them unreliable.

Much of the code could also be written in a less confusing way by utilising classes for paths and location systems, which would have sped up development and made extensions to the model easier to build.

7.1 Future Work

To improve the efficiency of the algorithms, it may be that the Levenshtein automata can be rewritten to be fast enough, or that a heuristic-based approach for path selection needs to be found that does not need to search an exponentially large set of objects.

There are several ideas that were not seen to completion, for instance implementing and testing the improved deobscuring mechanism $\mathcal{D}_1$ discussed earlier, as well as extending the Markov model to continuous time.

The location tracking attacks developed by Shokri et al. [6] should also be tested to ensure that $\mathcal{M}^{EPL}$ is not vulnerable to other methods than the ones discussed in this paper.

8 Conclusions

This report gave a statistical background as to the degradation of location privacy under repeated observation, and used this to build an algorithm to perform a location tracking attack. Sadly this algorithm was hampered with fundamental efficiency issues, and so not only is it impractical for common use but the experiments also suffered. There was insufficient time to redo the experiments and still collect enough data, so many measurements could not be taken. The deobscuring mechanisms found are therefore not useful in practice. However, as can be seen from previous work, location tracking attacks still pose a threat to location sharing systems [6, 3], and the problem of sharing a location path while preserving the utility of each point on the path has not been fully solved.

It is still believed that the location sharing mechanism of $\mathcal{M}^{EPL}$ provides a good basis for a private mechanism to share a users location path over time without decreasing utility, but further work is needed to provide a theoretical justification or better empirical evidence for this claim.

Acknowledgements

This report was completed at WMG, University of Warwick under the supervision of Professors Carsten Maple and Graham Cormode, with sponsorship from the National Cybersecurity Centre.

References

[1] Miguel Andrés et al. "Geo-Indistinguishability: Differential Privacy for Location-Based Systems". In: *20th ACM Conference on Computer and Communications Security*. ACM. 2013, pp. 901–914.

[2] Lorenzo Bracciale et al. *CRAWDAD dataset roma/taxi (v. 2014-07-17)*. Downloaded from `https://crawdad.org/roma/taxi/20140717`. July 2014. DOI: `10.15783/C7QC7M`.

[3] Yang Cao et al. "Quantifying differential privacy under temporal correlations". In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE. 2017, pp. 821–832.

[4] Cynthia Dwork, Aaron Roth, et al. "The algorithmic foundations of differential privacy". In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407.

[5] Klaus U Schulz and Stoyan Mihov. "Fast string correction with Levenshtein automata". In: *International Journal on Document Analysis and Recognition* 5.1 (2002), pp. 67–85.

[6] Reza Shokri et al. "Quantifying location privacy". In: *2011 IEEE symposium on security and privacy*. IEEE. 2011, pp. 247–262.

[7] Yonghui Xiao and Li Xiong. "Protecting locations with differential privacy under temporal correlations". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2015, pp. 1298–1309.