Breaching Euclidean Distance-Preserving Data Perturbation Using Few Known Inputs

Chris R. Giannella*

The MITRE Corporation, 300 Sentinel Dr. Suite 600, Annapolis Junction MD 20701, (301) 617-3000

Kun Liu

LinkedIn, 2029 Stierlin Court, Mountain View, CA 94043

Hillol Kargupta**

Dept. of CSEE, University of Maryland Baltimore County, Baltimore MD 21250

Abstract

We examine Euclidean distance-preserving data perturbation as a tool for privacy-preserving data mining. Such perturbations allow many important data mining algorithms (*e.g.* hierarchical and k-means clustering), with only minor modification, to be applied to the perturbed data and produce exactly the same results as if applied to the original data. However, the issue of how well the privacy of the original data is preserved needs careful study. We engage in this study by assuming the role of an attacker armed with a small set of known original data tuples (inputs). Little work has been done examining this kind of attack when the number of known original tuples is less than the number of data dimensions. We focus on this important case, develop and rigorously analyze an attack that utilizes *any number* of known original tuples. The approach allows the attacker to estimate the original data tuple associated with each perturbed tuple and calculate the probability that the estimation results in a privacy breach. On a real 16-dimensional dataset, we show that the attacker, with 4 known original tuples, can estimate an original unknown tuple with less than 7% error with probability exceeding 0.8.

Keywords: Euclidean distance, privacy, data mining, data perturbation

1. Introduction

Owners of sensitive information face a dilemma in many situations. On the one hand, making this data available for statistical analysis can violate the privacy of the individuals represented in

*Giannella is the corresponding author. This document was approved for public release, unlimited distribution, by The MITRE Corporation under case 10-2893. Giannella's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE concurrence with, or support of, the positions, opinions or viewpoints expressed.

**Kargupta is also affiliated with AGNIK LLC, Columbia MD

Email addresses: cgiannella@mitre.org (Chris R. Giannella), kun@linkedin.com (Kun Liu), hillol@cs.umbc.edu (Hillol Kargupta)

the data or reveal sensitive information about the data owner. On the other hand, making the data available can lead to discoveries that provide societal benefits. For example, mining health-care data for security/fraud issues may require analyzing clinical records and pharmacy transaction data of many individuals over a certain area. While the release of such data may violate privacy laws, mining it can improve the overall quality of the health-care system. Privacy-Preserving Data Mining (PPDM) strives to provide a solution to this dilemma. It aims to allow useful data patterns to be extracted without compromising privacy.

Data perturbation represents one common approach in PPDM. Here, the original private dataset X is perturbed and the resulting dataset Y is released for analysis. Perturbation approaches typically face a "privacy/accuracy" trade-off. On the one hand, perturbation must not allow the original data records to be adequately recovered. On the other hand, it must allow "patterns" that hold in the original data to be recovered. In many cases, increased privacy comes at the cost of reduced accuracy and vice versa. For example, Agrawal and Srikant [1] proposed adding randomly generated *i.i.d.* noise to the dataset. They showed how the distribution from which the original data arose can be estimated using only the perturbed data and the distribution of the noise. However, Kargupta *et al.* [2] and Huang *et al.* [3] pointed out how, in many cases, the noise can be filtered off leaving a reasonably good estimation of the original data (further investigated by Guo *et al.* [4]). These results point to the fact that unless the variance of the additive noise is sufficiently large, original data records can be recovered unacceptably well. However, this increase in variance reduces the accuracy with which the original data distribution can be estimated. This privacy/accuracy trade-off is not limited to additive noise; some other data transformation techniques suffer from a similar problem, *e.g.* k-anonymity [5].

Recently, Euclidean distance-preserving data perturbation for the *census model* [1]has gained attention ([7, 8, 9, 10, 11, 12, 13, 14]) because it mitigates the privacy/accuracy trade-off by guaranteeing perfect accuracy. The census model using Euclidean distance-preserving data perturbation can be illustrated as follows. An organization has a private, real-valued dataset X (represented as a matrix where each column is a data record) and wishes to make it publicly available for data analysis while keeping the individual records (columns) private. To accomplish this, $Y = T(X)$ is released to the public where $T(.)$ is a function, known only to the data owner that preserves Euclidean distances between columns. With this nice property, many useful data mining algorithms, with only minor modification, can be applied to Y and produce *exactly the same* patterns that would be extracted if the algorithm was applied directly to X. For example, assume single-link, agglomerative hierarchical clustering (using Euclidean distance) is applied directly to Y [15]. The cluster memberships in the resulting dendrogram will be identical to those in the dendrogram produced if the same algorithm is applied to X.

However, the issue of how well the private data is hidden after Euclidean distance-preserving data perturbation needs careful study. Without any prior knowledge, the attacker can do very little (if anything) to accurately recover the private data. However, no prior knowledge seems an unreasonable assumption in many situations. Consideration of prior knowledge-based attack techniques against Euclidean distance-preserving transformations is an important avenue of study. In this paper, we engage in this study by considering *known input* prior knowledge wherein the attacker knows a small set of original data tuples (inputs), but does not know their associated perturbed data tuples. As pointed out in [13, 14], this knowledge could be obtained through insider information. For example, consider a dataset where each record corresponds to

[1]The census model is widely studied in the field of security control for statistical databases [6].

information about an individual (*e.g.* medical data, census data). It is reasonable to assume that the individuals know (1) that a record for themselves appears in the dataset, and (2) the attributes of the dataset. As such, each individual knows one record in the original dataset. A small group of malicious individuals could then combine their insider information to produce a larger set of known original data tuples.

Summary of our contributions: The goal of the attacker is to use the perturbed data tuples and known original data tuples to produce good estimates of *unknown* original data tuples along with links to their perturbed counterparts. To achieve this, we develop an attack technique called the *known input attack* which proceeds in three steps.

1. The attacker links as many of the known original data tuples (columns in X) to their corresponding perturbed counterparts (columns in Y).
2. For each unlinked perturbed data tuple, the attacker computes the breach probability of the associated unknown original data tuple. This is the probability that the following stochastic procedure will result in an accurate enough estimate of the associated unknown original data tuple to be considered a privacy breach (the probability calculation is done by applying a closed-form expression we derive later).
   (a) A Euclidean distance-preserving transformation is uniformly chosen from the space of such transformations that satisfy the original-perturbed (input-output) constraints from step 1.
   (b) The inverse of the chosen transformation is used to estimate original data tuples from their perturbed counterparts.
3. The attacker chooses the perturbed data tuples which are most vulnerable to breach based their probabilities from step 2, *e.g.* chooses the one with the maximum probability or chooses all whose probability exceeds a threshold, and generates estimates of their associated known original data tuples.

When the number of linked, linearly independent known original data tuples exceeds the number of data dimensions, the privacy breach probability, for all unknown original data tuples, equals one as the estimates are guaranteed to be error-free. However, to our knowledge, little work has been done for the case where the number of known original data tuples is less than the number of data dimensions. This is an important case, since obtaining original data tuples is likely difficult. The attacker ought to be able to utilize however as many as she can get. Our results demonstrate how the attacker can do this and with increasing probability of success with respect to the number original data tuples obtained. Experiments on real and synthetic data show that even with the number of known original data tuples significantly smaller than the number of data dimensions, privacy can be breached with high probability. For example, on a real 16-dimensional dataset, we show that the attacker can use 4 known original data tuples to estimate an unknown original tuple with less than 7% error with probability exceeding 0.8.

Paper organization: Section 2 describes related work in data perturbation for privacy-preserving data analysis. Section 3 discusses some background material - the definition of T, a Euclidean distance-preserving data perturbation, and the definition of a privacy breach. Section 4 describes the main contribution of the paper - the known input attack outlined above. Section 5 discusses the results of experiments on real and synthetic data to evaluate the behavior of the attack. Section 6 provides a brief summary of the paper and a pointer to an idea for future work. Proofs and some detailed derivations are included in an appendix.

2. Related Work

In this section, we give a brief overview of a wide variety of data-perturbation techniques. We first introduce methods that do not preserve Euclidean distance between data tuples. Then we focus on research most relevant to this paper, a majority of which aim to preserve Euclidean distance by projecting private data to a new space.

2.1. General Data Perturbation and Transformation Methods

Additive perturbation: Adding *i.i.d.* white noise to protect data privacy is one common approach for statistical disclosure control [6]. The perturbed data allows the retrieval of aggregate statistics of the original data (*e.g.* sample mean and variance) without disclosing values of individual records. Moreover, additive white noise perturbation has received attention in the data mining literature [1, 2, 3, 4]. Clearly, additive noise does not preserve Euclidean distance and, therefore, is fundamentally different than the data perturbation we consider. An interesting example along these lines is given by Mukherjee *et al.* [16]. They considered additive noise to the most dominate principal components of the dataset along with a modification of k-nearest-neighbor classification [17] on the perturbed data to improve accuracy. Moreover, they nicely extend to additive noise the ρ_1-to-ρ_2 privacy breach measure originally introduced for categorical data in [18]. Another example is Liu *et al.* [19]. They argued that the level of additive noise ought to be flexible per record. They developed a modified addative noise approach allowing the level of noise to be varied per record based on data owner preference.

Multiplicative perturbation: Two traditional multiplicative data perturbation schemes were studied in the statistics community [20]. One scheme multiplies each data element by a random number that has a truncated Gaussian distribution with mean one and small variance. The other takes a logarithmic transformation of the data first, adds multivariate Gaussian noise, then takes the exponential function *exp(.)* of the noise-added data. These perturbations allow summary statistics (*e.g.*, mean, variance) of the attributes to be estimated, but do not preserve Euclidean distances among records.

To assess the security of traditional multiplicative perturbation together with additive perturbation, Trottini *et al.* [21] proposed a Bayesian intruder model that considers both prior and posterior knowledge of the data. Their overall strategy of attacking the privacy of perturbed data using prior knowledge is the same as ours. However, they particularly focused on linkage privacy breaches, where an intruder tries to identify the identity (of a person) linked to a specific record; while we are primarily interested in data record recovery. Moreover, they did not consider Euclidean distance-preserving perturbation as we do.

k-anonymization: Samarati and Sweeney [5, 22] originally developed the *k-anonymity* model to transform person-specific data. Their work shows that an attacker can link a subset of data attributes (called quasi-identifiers) with third-party information to uniquely identify a person even when his personally identifiable information is not present in the original data. To mitigate the risk, the authors proposed the suppression or generalization of values of these quasi-identifiers so that any records in the database, when projected onto the quasi-identifiers, cannot be distinguished from at least *k-1* others. This model has drawn much of attention because of its simple privacy definition. Since its initial appearance, a variety of extensions have been developed to anonymize transactional data [23], sequential data [24], and mobility data [25]. We refer interested readers to the survey book [26] for more details. It should be noted that none of these approaches consider Euclidean distance-preserving perturbation as we do.

Data micro-aggregation: Two multivariate micro-aggregation approaches have been proposed by researchers in the data mining area. The technique presented by Aggarwal and Yu [27] partitions the original data into multiple groups of predefined size. For each group, a certain level of statistical information (*e.g.*, mean and covariance) is maintained. This statistical information is used to create anonymized data that has similar statistical characteristics to the original dataset. Li *et al.* [28] proposed a kd-tree based perturbation method, which recursively partitions a dataset into subsets which are progressively more homogeneous after each partition. The private data in each subset is then perturbed using the subset average. The relationships between attributes are argued to be preserved reasonably well. However, neither of these two approaches preserves Euclidean distance between the original data tuples.

Data swapping and shuffling: Data swapping transforms a database by exchanging values of sensitive attributes among individual records. Records are exchanged in such a way that the lower-order frequency counts or marginals are maintained. A variety of refinements and applications of data swapping have been addressed since its initial appearance. We refer readers to [29] for a thorough treatment. Data shuffling [30] is similar to swapping, but is argued to improve on many of the shortcomings of swapping for numeric data. However, neither swapping or shuffling preserves Euclidean distance, which is the focus of this paper.

Other techniques: Evfimievski *et al.* [18], Rizvi and Haritza [31] considered the use of categorical data perturbation in the context of association rule mining. Their algorithms delete real items and add bogus items to the original records. Association rules present in the original data can be estimated from the perturbed data. Along a related line, Verykios *et al.* [32] considered perturbation techniques which allow the discovery of some association rules while hiding others considered to be sensitive. We refer interested readers to Chapter 11 of the survey book [26] for a nice overview of association rule hiding methods.

Oliveira and Zaiane [9] consider the application of a rotation, additive noise, and multiplicative noise, separately to each original data *attribute*. As such, their transformation is not guaranteed to preserve Euclidean distance between data *tuples*. However, Oliveira and Zaiane argue, through experiments, that their overall data perturbation technique preserves the accuracy of two clustering algorithms.

Similar to [9], Ting *et al.* [33] considered perturbation of the data attributes by an orthogonal transformation. More precisely, Ting *et al.* considered left-multiplication of the original data matrix by a randomly generated orthogonal matrix. However, they assume the original data tuples are *rows* rather than columns, as we do. As a result, Euclidean distance between original data tuples is not preserved, but, sample mean and covariance is. If the original data arose as independent samples from multi-variate Gaussian distribution, then the perturbed data allows inferences to be drawn about this underlying distribution just as well as the original data. For all but small or very high-dimensional datasets, their approach is more resistant to prior knowledge attacks than Euclidean distance-preserving perturbations. Their perturbation matrix is $m \times m$ (m is the number of original data tuples), much bigger than Euclidean distance-preserving perturbation matrices, $n \times n$ (n is the number of data dimensions).

A survey: Fung *et al.* [34] provided a detailed survey of work related to this paper (using the descriptive term "privacy-preserving data publishing"). They discussed a wide range of data perturbation and transformation techniques, as well as, approaches to breach privacy. They also discussed scenarios other than the census model, *e.g.* multiple release data publishing and statistical database querying.

2.2. Euclidean Distance-Preserving Data Perturbation

In this part, we describe research most related to this paper. The majority of the work focuses on Euclidean distance-preserving data perturbation.

Chen and Liu [7] observe that some classifiers are invariant with respect to Euclidean distance between the training tuples. The authors quantify the privacy offered by a Euclidean distance preserving perturbation in terms of the empirical covariance matrix with respect to the difference between the original and perturbed data attributes. The authors' privacy quantification does not take into account prior knowledge, hence, the attack based on prior knowledge presented in our paper applies directly to the Euclidean distance preserving data perturbation method of Chen and Liu. An important issue not discussed by Chen and Liu is how the classifier learned from perturbed data will be used to classify new tuples. Perturbing the new tuples and applying the classifier would produce the same result as if a classifier built from the unperturbed training data was applied to the unperturbed new tuples. But, the process of perturbing the new tuples and applying the classifier need be done with great care to not leak information that could be used to recover the original training tuples.

Oliveira and Zaiane [8] observe that some clustering algorithms are invariant with respect to Euclidean distance between data tuples. The authors quantify privacy using an approach related to that in Chen and Liu. Like Chen and Liu, Oliveira and Zaiane do not consider prior knowledge, hence, the attack based on it presented in our paper applies directly to the Euclidean distance preserving data perturbation method of Oliveira and Zaiane.

Liu *et al.* [10] developed two types of attacks to breach the privacy of distance-preserving data perturbation.

1. Liu developed the *known-sample attack* which assumes that the attacker has a moderate-sized collection of independent samples chosen i.i.d. from the same distribution as the private data. By mapping the principal components of the perturbed data to the principle components of the original data (estimated from the sample), the attacker can reconstruct the perturbation matrix and consequently recover the private data. The prior knowledge assumption made by this attack is different than the assumption in our manuscript of a very small set of known original tuples. For example, the known sample prior knowledge of Liu requires the original dataset and known sample be drawn i.i.d. (from the same distribution), while the assumption in our manuscript requires no i.i.d. or any other distribution assumptions. If, in our manuscript, we make the additional assumption that the original data is drawn i.i.d., then the known sample attack of Liu can, in theory, be applied. But, the attack's accuracy will be very low as the attack requires a much larger sample than the size of the known tuples we are considering.
2. Liu developed the *known input-output attack* which assumes that the attacker knows a very small subset of the original (private) data tuples *and* their correspondences to perturbed tuples (i.e. for each known original tuple, the attacker is assumed to know which is its corresponding perturbed tuple). Their attack technique is the same as a part of our attack – choose an orthogonal matrix randomly from the set of those that satisfy the input-output constraints. Then use a closed-form expression for the breach probability for each private tuple to choose the best one to re-estimate. However, we significantly weaken and make more realistic (providing an explicit scenario) the prior knowledge assumption. We assume only that the attacker knows a very small subset of original (private) data tuples, but does not know their correspondences to perturbed tuples. We extend the attack algorithm of Liu to first infer the correspondences between the known original tuples and the perturbed

tuples. Also, we provide a complete and rigorous mathematical analysis of the attack (Liu did not do this). We also correct a mistake in $\rho(x_j, \epsilon)$, the probability closed-form expression. Finally, we provide experimental results (run-time and accuracy) for the attack (Liu did not do this).

Chen *et al.* [12] also discussed a known input attack technique. Unlike ours, they considered a combination of distance-preserving data perturbation followed by additive noise. They also assumed a stronger form of known input prior knowledge: the attacker knows a subset of private data records *and* knows to which perturbed tuples they correspond. Finally, they assume that the number of linearly independent known input data records is no smaller than the number of data dimensions. They pointed out that linear regression can be used to re-estimate private data tuples.

Mukherjee *et al.* [11] considered the use of discrete Fourier transformation (DFT) and discrete cosine transformation (DCT) to perturb the data. Only the high energy DFT/DCT coefficients are used, and the transformed data in the new domain approximately preserves Euclidean distance. The DFT/DCT coefficients were further permuted to enhance the privacy protection level. Note that DFT and DCT are (complex) orthogonal transforms. Hence, their perturbation technique can be expressed as left multiplication by a (complex) orthogonal matrix (corresponding to the DFT/DCT followed by a perturbation of the resulting coefficients), then a left multiplication by an identity matrix with some zeros on the diagonal (corresponding to dropping all but the high-energy coefficients). They did not consider attacks based on prior knowledge. For future work, it would be interesting to do so.

Turgay *et al.* [13] extended some of the results in [10]. They assume that the similarity matrix of the original data is made public rather than, Y, the perturbed data itself. They describe how an attacker, given at least $n + 1$ linearly independent original data tuples *and* their corresponding entries in the similarity matrix, can recover the private data (n is the number of data dimensions). Like Chen *et al.*, this differs from our known input attack in two main ways: (i) we do not require prior knowledge beyond the known input tuples; (ii) our attack analysis smoothly encompasses the case where the number of linearly independent known input tuples is greater than n as well as less.

Wong *et al.* [14] considered data perturbation as a solution to privacy problems introduced by data outsourcing wherein an un-trusted party holds the perturbed data and computes k-nearest-neighbor queries against it on behalf of other parties. Among other things, they examined the vulnerabilities of the perturbed data against an attacker armed with known input prior knowledge (their "level 2" prior knowledge). Independently of us, they briefly discussed a basic idea for linking the known inputs to their perturbed counterparts that is similar to our linking technique (although they provide only a cursory description omitting many details).[2] They point out how a distance-preserving data perturbation can be undone if the number of linearly independent known inputs that can be linked to perturbed tuples exceeds the number of data dimensions. Their work differs from ours in that it says nothing about the case where the number of linearly independent, linked known tuples is less than the number of data dimensions.

Kaplan *et al.* [35] considered the estimation of private trajectories (vectors of real numbers) given various kinds of prior knowledge like Euclidean distances from the private trajectories to

[2] We described our linking technique in an earlier, unpublished, technical report version of this paper (citation omitted because of the double-blind nature of this submission). This report appeared 3 months after Wong's paper, and, at the time we were unaware of Wong's work.

a known one. They develop an innovative algorithm that can incorporate a wide variety of types of prior knowledge and produce estimates. The primary differences between Kaplan's and our work are as follows. Our work applies to the more general problem where the attacker has only a collection of known inputs and does not know their perturbed counterparts. We develop a novel technique for linking the known inputs to their perturbed counterparts. Once this is done, Kaplan's algorithm can be applied to estimate unknown private tuples from the known input-output pairs. However, unlike Kaplan's, our approach provides precise estimation error guarantees, namely, the precise value of the estimation error probability. Thus, with our approach, the attacker can know (in probability) how good each of the estimates is, and, for example, pick the best one. Through experiments, we found our approach to be significantly more accurate than Kaplan's. On the other hand, Kaplan's approach has the advantage over ours of being more general in the sense that it can incorporate a larger variety of prior knowledge into the attack. Our approach is tailored to known input-output prior knowledge.

Before we briefly describe another two attacks based on independent component analysis (ICA) [36], it is necessary to give a brief ICA overview.

2.2.1. ICA Overview

Given an n'-variate random vector $\mathcal{V}$, one common ICA model posits that this random vector was generated by a linear combination of independent random variables, *i.e.*, $\mathcal{V} = A\mathcal{S}$ with $\mathcal{S}$ an n-variate random vector with independent components. Typically, $\mathcal{S}$ is further assumed to satisfy the following additional assumptions: (i) at most one component is distributed as a Gaussian; (ii) $n' \geq n$; and (iii) A has rank n (full rank).

One common scenario in practice: there is a set of unobserved samples (the columns of $n \times q$ matrix S) that arose from $\mathcal{S}$ which satisfies (i) - (iii) and whose components are independent. But observed is $n' \times q$ matrix V whose columns arose as linear combination of the rows of S. The columns of V can be thought of as samples that arose from a random vector $\mathcal{V}$ which satisfies the above generative model. There are ICA algorithms whose goal is to recover S and A from V up to a row permutation and constant multiple. This ambiguity is inevitable due to the fact that for any diagonal matrix (with all non-zeros on the diagonal) D, and permutation matrix P, if A, S is a solution, then so is (ADP), $(P^{-1}D^{-1}\mathcal{S})$.

2.2.2. ICA Based Attacks

Liu *et al.* [37] considered matrix multiplicative data perturbation, $Y = MX$, where M is an $n' \times n$ matrix with each entry generated independently from the same distribution with mean zero and variance σ^2. They discussed the application of the above ICA approach to estimate X directly from Y: $\mathcal{S} = \mathcal{X}$, $\mathcal{V} = \mathcal{Y}$, $S = X$, $V = Y$, and $A = M$. They argued the approach to be problematic because the ICA generative model imposes assumptions not likely to hold in many practical situations: the components of $\mathcal{X}$ are independent with at most one such being Gaussian distributed. Moreover, they pointed out that the row permutation and constant multiple ambiguity further hampers accurate recovery of X. A similar observation is made later by Chen *et al.* [12].

Guo and Wu [38] considered matrix multiplicative perturbation assuming only that M is an $n \times n$ matrix (orthogonal or otherwise). They assumed the attacker has known input prior knowledge, *i.e.* she knows, $\widetilde{X}$, a collection of original data columns from X. They develop an ICA-based attack technique for estimating the remaining columns in X. To avoid the ICA problems described in the previous paragraph, they instead applied ICA *separately* to $\widetilde{X}$ and Y producing representations $(A_{\widetilde{X}}, S_{\widetilde{X}})$ and (A_Y, S_Y). They argued that these representations are

related in a natural way allowing X to be estimated. Their approach, however, will be quite inaccurate for extremely small numbers of known inputs. Moreover, their approach does not provide the attacker with any sort of error information and she will thus not know which (if any) of her original data tuple estimates are accurate.

3. Euclidean Distance-Preserving Perturbation and Privacy Breaches

This section provides: some common notation used throughout the article, the definition of T a Euclidean distance-preserving data perturbation, the definition of a privacy breach, and a small example illustrating a Euclidean distance-preserving perturbation.

3.1. Notation and Conventions

In the rest of this paper, unless otherwise stated, the following notations and conventions are used. "Euclidean distance-preserving" and "distance-preserving" are used interchangeably. All matrices and vectors discussed are assumed to have real entries (unless otherwise stated). All vectors are assumed to be column vectors and M' denotes the transpose of any matrix M. Given a vector x, $||x||$ denotes its Euclidean norm. An $m \times n$ matrix M is said to be *orthogonal* if $M'M = I_n$, the $n \times n$ identity matrix.[3] The set of all $n \times n$, orthogonal matrices is denoted by $\mathbb{O}_n$.

Given $n \times p$ and $n \times q$ matrices A and B, let $[A|B]$ denote the $n \times (p + q)$ matrix whose first p columns are A and last q are B. Likewise, given $p \times n$ and $q \times n$ matrices A and B, let $\begin{bmatrix} A \\ B \end{bmatrix}$ denote the $(p + q) \times n$ matrix whose first p rows are A and last q are B.

The data owner's private dataset is represented as an $n \times m$ matrix X, with each column a record and each row an attribute (each record is assumed to be non-zero). The data owner applies a Euclidean distance-preserving perturbation to X to produce an $n \times m$ data matrix Y, which is then released to the public or another party for analysis. That Y was produced from X by a Euclidean distance-preserving data perturbation (but not which one) is also make public.

3.2. Euclidean Distance-Preserving Perturbation

A function $H : \Re^n \to \Re^n$ is Euclidean distance-preserving if for all $x, y \in \Re^n$, $||x - y|| = ||H(x) - H(y)||$. Here H is also called a *rigid motion*. It has been shown that any distance-preserving function is equivalent to an orthogonal transformation followed by a translation [39, pg. 128]. In other words, H may be specified by a pair $(M, v) \in \mathbb{O}_n \times \Re^n$, in that, for all $x \in \Re^n$, $H(x) = Mx + v$. If $v = 0$, H preserve Euclidean length: $||x|| = ||H(x)||$, as such, it moves x along the surface of the hyper-sphere with radius $||x||$ and centered at the origin.

Recall that columns of X (denoted $x_1, \ldots, x_m$) refer to private data records. And, columns of $T(X) = Y$ (denoted $y_1, \ldots, y_m$) refer to perturbed data records. The correspondence between the private and perturbed data records is not assumed known, *e.g.* the perturbed version of x_i is not necessarily y_i. Instead, the columns of X are transformed using a Euclidean distance-preserving function, then are permuted to produce the columns of the perturbed dataset Y. Formally, the perturbed dataset Y, is produced as follows. The private data owner chooses (M_T, v_T), a secret Euclidean distance-preserving function, and π, a secret permutation of $\{1, \ldots, m\}$. Then, for $1 \leq i \leq m$, the data owner produces $y_{\pi(i)} = M_T x_i + v_T$.

Euclidean distance between the private data tuples is preserved in the perturbed dataset: for all $1 \leq i, j \leq m$, $||x_i - x_j|| = ||y_{\pi(i)} - y_{\pi(j)}||$. Moreover, if $v_T = 0$, then length of the private data tuples is also preserved: for all $1 \leq i \leq m$, $||x_i|| = ||y_{\pi(i)}||$.

[3] If M is square, it is orthogonal if and only if $M' = M^{-1}$ [39, pg. 17].

3.3. Privacy Breach

For simplicity, we assume the attacker produces an estimate for a single unknown original data tuple.[4] Formally, the attacker will employ a stochastic procedure and produce $1 \leq j \leq m$ and non-zero, $\hat{x} \in \Re^n$. Here, $\hat{x}$ is an estimate of $x_{\hat{j}}$ (with $\hat{j}$ denoting $\pi^{-1}(j)$), the private original data tuple that was perturbed to produce y_j.[5] Given $\epsilon > 0$, we define a privacy breach as follows.

Definition 3.1. *An* ϵ-privacy breach *occurs if* $\|\hat{x} - x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon$, i.e. *if the attacker's estimate is wrong with Euclidean relative error no more than* ϵ.

In the next section, we describe and analyze the known input attack. The main focus of analysis concerns, $\rho(\epsilon)$, the probability that an ϵ-privacy breach occurred.

3.4. Example

Figure 1 illustrates a small private dataset (left) and the result of applying a simple Euclidean distance-preserving perturbation (a 90-degree clockwise rotation and identity π). In general, Euclidean distance-preserving perturbations can be much more complex than the one illustrated here.

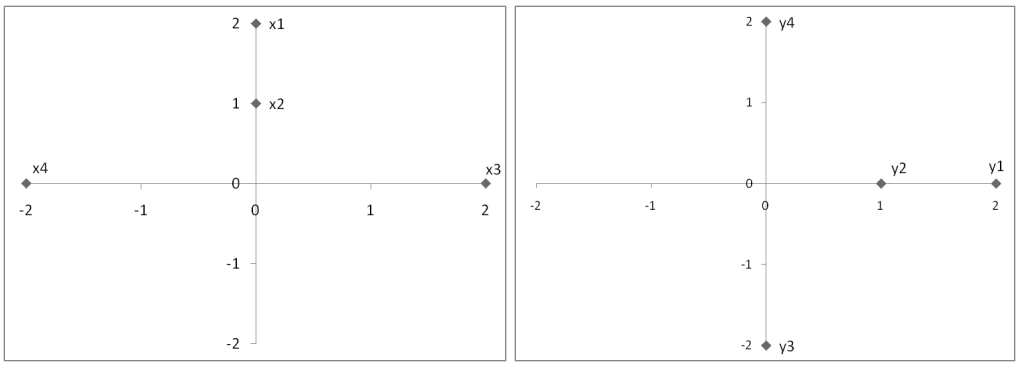


Figure 1: A four record, original private dataset (left) and the result of applying a simple Euclidean distance-preserving perturbation: 90-degree clockwise rotation (the perturbed records permutation, π, is the identity).

4. Known Input Attack

For $1 \leq a \leq m-1$, let X_a denote the first a columns of X. The attacker is assumed to know X_a and her attack proceeds in three steps. For the remainder of the paper we use interchangeably "known inputs" and "known original data tuples".

1. Infer as many as possible of the input-output mappings in π_a (the restriction of π to $\{1, \ldots, a\}$), that is, find as many as possible perturbed counterparts of X_a in Y.

[4] As described in Section 1, this can easily be extended to produce estimates for as many unknown original data tuples as desired.

[5] The attacker does not need to know $\hat{j}$; she is merely producing an estimate of the private data tuple that was perturbed to produce y_j.

2. For each perturbed tuple y_j in Y which is not mapped onto by π_a, compute the probability that the following stochastic procedure will result in an ϵ-privacy breach when estimating the original tuple associated with y_j (the probability calculation is done using a closed-form expression derived later).
   (a) Estimate M_T by choosing a matrix, $\hat{M}$, uniformly from the space of all orthogonal matrices that map the tuples in X_a to their π_a counterparts in Y (as computed in step 1).
   (b) Estimate the original tuple associated with y_j as $\hat{M}'y_j$.
3. Choose the y_j with the highest probability from step 2 and produce $\hat{x} = \hat{M}'y_j$.

The bulk of our work involves the development and analysis of an attack technique in the case where the data perturbation is assumed to be orthogonal (does not involve a fixed translation, $v_T = 0$). The majority of this section is dedicated to developing and analyzing an attack in this case. Then, in Subsection 4.5, we briefly describe how the attack and analysis can be extended to arbitrary Euclidean distance-preserving perturbation ($v_T \neq 0$).

4.1. Inferring π_a

The attacker may not have enough information to infer π_a, so, her goal is to infer π_I (the restriction of π to $I \subseteq \{1, \ldots, a\}$), for as large an I as possible. Next, we describe how this goal can be precisely stated as an algorithmic problem that the attacker can address given her available information.

Given $I \subseteq \{1, \ldots, a\}$, an *assignment on* I is a 1-1 function $\beta : I \to \{1, \ldots, m\}$. An assignment β on I is *valid* if it satisfies both of the following conditions for all $i, j \in I$, (1) $\|x_i\| = \|y_{\beta(i)}\|$ and (2) $\|x_i - x_j\| = \|y_{\beta(i)} - y_{\beta(j)}\|$. Importantly, if β is not valid, it cannot be a correct linkage between tuples in X_a and Y, *i.e.* $\beta \neq \pi_I$. As such, there is at least one valid assignment on I, namely π_I, but, there may be more. If β is the only valid assignment on I, then it must equal π_I.

For notational convenience, we say that I is *uniquely valid* if there is only one valid assignment on I. The attacker's goal is to find a *maximal* uniquely valid I, *i.e.* a uniquely valid I such that there does not exist uniquely valid J with $|J| > |I|$. It can be shown that there exists only one maximal uniquely valid subset of $\{1, \ldots, a\}$. Thus, the attacker's goal is to find the maximal uniquely valid subset of $\{1, \ldots, a\}$ along with its corresponding assignment.

The following straight-forward algorithm will meet the attacker's goal by employing a top-down, level-wise search of the subset space of $\{1, \ldots, a\}$. The inner for-loop uses an implicit linear ordering to enumerate the size ℓ subsets without repeats and requiring $O(1)$ space.

Algorithm 1 Overall Algorithm For Finding the Maximal Uniquely Valid Subset

```
For ℓ = a, ..., 1, do
    For all I ⊆ {1, ..., a} and |I| = ℓ, do
        If I is uniquely valid, then output I along with its corresponding assignment and terminate the algorithm.
Otherwise output ∅.
```

Example revisited – part 1: consider the dataset and its perturbed version illustrated in Figure 1 and assume that $X_3 = [x_1 x_2 x_3]$ are the known original data tuples ($a = 3$). Algorithm 1 proceeds as follows.

- Check if $I = \{1, 2, 3\}$ is uniquely valid. Since the distances of y_3 to y_2 and y_1 are the same as those of y_4 to y_2 and y_1, then the assignment $\beta : 1 \mapsto 1, 2 \mapsto 2, 3 \mapsto 4$ is valid. The identity assignment on I is also valid because, in this example, π is the identity permutation. Thus, I has more than one valid assignment (is not uniquely valid).

- Check if $I = \{1, 2\}$ is uniquely valid. To see that I is uniquely valid note that any valid assignment, β, must assign 2 to itself or else $\|x_2\| \neq \|y_{\beta(2)}\|$. And, it can be checked that $\beta(1) \neq 3, 4$ in order to satisfy $1 = \|x_1 - x_2\| = \|y_{\beta(1)} - y_{\beta(2)}\|$. Therefore, β must be the identity assignment on I.

- The algorithm terminates and outputs $I = \{1, 2\}$ as the maximal uniquely valid subset of $\{1, 2, 3\}$ with assignment $1 \mapsto 1, 2 \mapsto 2$. Note: any ordering on the subsets of size two may be considered. We chose lexicographic order for simplicity.

□

Now we develop an algorithm that, given $I \subseteq \{1, \ldots, a\}$, determines if I is uniquely valid, and, if so, also computes the corresponding assignment. The idea is to search the space of all assignments on I for valid ones. Once more than one valid assignment is identified, the search is cut-off and the algorithm outputs that I is not uniquely valid. Otherwise, exactly one valid assignment, π_I, will be found. In this case, the algorithm outputs that I is uniquely valid and returns the corresponding assignment. The algorithm performs a depth-first search with each node, $\mathcal{N}_1$, in the search tree representing $\emptyset \subseteq I_1 \subseteq I$ and β_1 a valid assignment on I_1. The search proceeds by considering all $\hat{I}_1 = I_1 \cup \{\hat{i}_1\}$ where $\hat{i} \in (I \setminus I_1)$ and all possible ways of extending β_1 to be a valid assignment, $\hat{\beta}_1$ on $\hat{I}_1$. In turn, $\hat{I}_1$ and $\hat{\beta}_1$ represent a node, $\hat{\mathcal{N}}_1$, in the search tree immediately below $\mathcal{N}_1$. If $\hat{I}_1 = I$, then a *NumValidAssignFound* counter is incremented. If the counter exceeds one, then I has more than one valid assignment, and the search is terminated.

To make this search efficient, we employ a simple, but effective, pruning rule to quickly eliminate possible extensions of β_1 that are not valid assignments on $\hat{I}_1 = I_1 \cup \{\hat{i}_1\}$. Let $C(I_1, \beta_1, \hat{i})$ denote the set of all extensions of β_1, $\hat{i} \mapsto j$, which appropriately preserve Euclidean distances; formally put, all $j \in (\{1, \ldots, m\} \setminus \beta_1(I_1))$ which satisfies both of the following conditions: (1) $\|x_{\hat{i}}\| = \|y_j\|$, and (2) for all $i_1 \in I_1$, $\|x_{i_1} - x_{\hat{i}}\| = \|y_{\beta_1(i_1)} - y_j\|$. It can be shown that $j \notin C(I_1, \beta_1, \hat{i})$ does not represent a valid assignment. Therefore, to enumerate all possible, valid, extensions of β_1 on $\hat{I}_1$, it suffices to consider those assignments $\hat{\beta}_1$ on $\hat{I}_1$ which are of the following form: (i) for all $\ell \in I_1$, $\hat{\beta}_1(\ell) = \beta_1(\ell)$ and (ii) $\hat{\beta}_1(\hat{i}_1) = j$ for some j in $C(I_1, \beta_1, \hat{i})$.

Algorithms 2 and 3 describe the precise details of the determination whether I is uniquely valid (namely, the details of the search discussed in the previous two paragraphs).

Algorithm 2 Determining Unique Validity Main

Inputs: $I \subseteq \{1, \ldots, a\}$.

1: Set global variable *NumValidAssignFound* = 0.
2: Call Algorithm 3 on inputs $\emptyset$ and $\beta_\emptyset$ ($\beta_\emptyset$ denotes the unique valid assignment on $\emptyset$).
3: If *NumValidAssignFound* > 1, then return "I IS NOT UNIQUELY VALID". Else, return "I IS UNIQUELY VALID WITH ASSIGNMENT" β_I.

Comment: The order by which the elements of $(I \setminus I_1)$ and $C(I_1, \beta_1, \hat{i})$ are chosen in iterating through the for loops in Algorithm 3 does not affect the correctness of the algorithm. However, it may affect efficiency. For simplicity, the loops order the elements in these sets from smallest to largest index number.

Algorithm 1 has worst-case computational complexity $O(m^a)$. While this is no better than a simple brute-force approach, in our experiments, quite reasonable running times are observed because few original data tuples will have the same length and/or few pairs of original data tuples will have the same Euclidean distance.

Algorithm 3 Determining Unique Validity Recursive

Inputs: $I_1 \subseteq I$ and β_1 a valid assignment on I_1.

If $I_1 = I$, then
    $NumValidAssignFound = NumValidAssignFound + 1$
    If $NumValidAssignFound == 1$, then set β_I to β_1.
    End If.
Else, do
    For $\hat{i} \in (I \setminus I_1)$ and as long as $NumValidAssignFound \leq 1$, do
        For $j \in C(I_1, \beta_1, \hat{i})$ and as long as $NumValidAssignFound \leq 1$, do
            Extend β_1 to $\hat{\beta_1}$ s.t. $\hat{\beta_1}(\hat{i}) = j$. Let $\hat{I_1} = I_1 \cup \hat{i}$.
            Call algorithm 3 on inputs $\hat{I_1}$ and $\hat{\beta_1}$.

4.2. Known Input-Output Attack

Assume, without loss of generality, that the attacker applies Algorithm 1 and learns π_q ($0 \leq q \leq a$), *i.e.* $\{1, \ldots, q\}$ is the maximal uniquely valid subset of $\{1, \ldots, a\}$. Further, to simplify notation, we may also assume that $\pi_q(i) = i$.[6] Let Y_q denote the first q columns of Y. As such, the attacker is assumed to know X_q and the fact that $Y_q = M_T X_q$ where M_T is an unknown orthogonal matrix. Based on this, she will apply an attack, called the *known input-output attack*, to produce $q < j \leq m$, and $\hat{x}$, which is an estimate of $x_{\hat{j}}$, the private tuple that was perturbed to produce y_j. The known input-output attack was described in the last two steps in the algorithm at the beginning of Section 4. More formally, the known input-output attack is as follows. Let $\mathbb{M}(X_q, Y_q)$ denote the set of all $M \in \mathbb{O}_n$ such that $MX_q = Y_q$.

1. For each $q < j \leq m$, compute the probability that the following stochastic procedure will result in an ϵ-privacy breech when estimating $x_{\hat{j}}$.
   (a) Estimate M_T by choosing a matrix, $\hat{M}$, uniformly from $\mathbb{M}(X_q, Y_q)$.
   (b) Estimate $x_{\hat{j}}$ as $\hat{M}' y_j$.[7]
2. Choose the y_j with the highest probability from step 2 and produce $\hat{x} = \hat{M}' y_j$.

A key component of the known input-output attack is the computation of $\rho(x_{\hat{j}}, \epsilon) = Pr(\|\hat{M}' y_j - x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon)$, the probability that an ϵ-privacy breach will result from the attacker estimating $x_{\hat{j}}$ as $\hat{M}' y_j$. In Section 4.4, we will develop a closed-form expression for $\rho(x_{\hat{j}}, \epsilon)$. This expression will only involve information known to the attacker; therefore, she can choose $q < j \leq m$ so as to maximize $\rho(x_{\hat{j}}, \epsilon)$. Another key component of the known input-output algorithm is in choosing $\hat{M}$ uniformly from $\mathbb{M}(X_q, Y_q)$. In most cases, $\mathbb{M}(X_q, Y_q)$ is uncountable and it is not obvious how to choose $\hat{M}$. We will develop and algorithm for doing so in Section 4.4. Before getting to Section 4.4, we discuss some important linear algebra background.

4.3. Linear Algebra background

Let $Col(X_q)$ denote the column space of X_q and $Col_\perp(X_q)$ denote its orthogonal complement, *i.e.*, $\{z \in \Re^n : z'w = 0, \forall w \in Col(X_q)\}$. Likewise, let $Col(Y_q)$ denote the column space of Y_q and $Col_\perp(Y_q)$ denote its orthogonal compliment. Let k denote the dimension of $Col(X_q)$. The "Fundamental Theorem of Linear Algebra" [40, pg. 95] implies that the dimension of $Col_\perp(X_q)$

[6] This can be achieved by the attacker appropriately reordering the columns of X_a and Y.
[7] This is equivalent to a maximum likelihood estimate of $x_{\hat{j}}$.

is $n-k$. Since $Y_q = M_T X_q$ and M_T is orthogonal, then it can be shown that $Col(Y_q)$ has dimension k. Thus, $Col_\perp(Y_q)$ has dimension $n-k$.

Let U_k and V_k denote $n \times k$ matrices whose columns form an orthonormal basis for $Col(X_q)$ and $Col(Y_q)$, respectively. It can easily be shown that $Col(M_T U_k) = Col(Y_q) = Col(V_k)$. Let U_{n-k} and V_{n-k} denote $n \times (n-k)$ matrices whose columns form an orthonormal basis for $Col_\perp(X_q)$ and $Col_\perp(Y_q)$, respectively. It can easily be shown that $Col(M_T U_{n-k}) = Col_\perp(Y_q) = Col(V_{n-k})$.

4.4. A Closed-Form Expression for $\rho(x_{\hat{j}}, \epsilon)$

Now we return to the issue of how to choose $\hat{M}$ uniformly from $\mathbb{M}(X_q, Y_q)$ and how to compute $\rho(x_{\hat{j}}, \epsilon) = Pr(\|\hat{M}' y_j - x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon) = Pr(\|\hat{M}' M_T x_{\hat{j}} - x_{\hat{j}}\|)$.

To choose $\hat{M}$ uniformly from $\mathbb{M}(X_q, Y_q)$, the basic idea is to utilize standard algorithms for choosing a matrix P uniformly from $\mathbb{O}_{n-k}$, the set of all $(n-k)\times(n-k)$ orthogonal matrices, then apply an appropriately designed transformation to P. The transformation will be an affine, bijection from $\mathbb{O}_{n-k}$ to $\mathbb{M}(X_q, Y_q)$.[8] The following technical result, proven in Appendix A, provides this transformation.[9]

Theorem 4.1. *Let L be the mapping $P \in \mathbb{O}_{n-k} \mapsto M_T U_k U_k' + V_{n-k} P U_{n-k}'$. Then, L is an affine bijection from $\mathbb{O}_{n-k}$ to $\mathbb{M}(X_q, Y_q)$. And, L^{-1} is the mapping $M \in \mathbb{M}(X_q, Y_q) \mapsto V_{n-k}' M U_{n-k}$.*

Algorithm 4 Uniform Choice From $\mathbb{M}(X_q, Y_q)$

Inputs: U_k, an $n \times k$ matrix whose columns form an orthonormal basis of $Col(X_q)$, and $M_T U_k$ (M_T is unknown); U_{n-k} and V_{n-k}, $n \times (n-k)$ matrices whose columns form an orthonormal basis of $Col_\perp(X_q)$ and $Col_\perp(Y_q)$, respectively.
Outputs: $\hat{M}$ a uniformly chosen matrix from $\mathbb{M}(X_q, Y_q)$.
1: Choose P uniformly from $\mathbb{O}_{n-k}$ using algorithm [41].
2: Set $\hat{M} = L(P)$, *i.e.*, $M_T U_k U_k' + V_{n-k} P U_{n-k}'$.

Two comments are in order regarding Algorithm 4. First, some special cases are interesting to highlight: when $k = n$, $\hat{M}$ is chosen as M_T; when $k = n-1$, $\hat{M}$ is one of two choices (one of which equals M_T); otherwise, $\hat{M}$ is, in theory, chosen from an uncountable set (containing M_T). Second, it is not obvious how the attacker can compute the inputs to the algorithm, *e.g.* $M_T U_k$. This issue will be discussed later when spelling out the details of the Known Input Attack Algorithm, Algorithm 5.

Now we develop a closed-form expression for $\rho(x_{\hat{j}}, \epsilon)$. The key points are outlined, while a more rigorous justification is provided in Appendix A. First of all, from Algorithm 4, $\hat{M} = M_T U_k U_k' + V_{n-k} P U_{n-k}'$ where P is chosen uniformly from $\mathbb{O}_{n-k}$. Therefore,

$$\begin{aligned} \rho(x_{\hat{j}}, \epsilon) &= Pr(\|\hat{M}' M_T x_{\hat{j}} - x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon) \\ &= Pr(\|U_k U_k' x_{\hat{j}} + U_{n-k} P' V_{n-k}' M_T x_{\hat{j}} - x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon). \end{aligned}$$

Since $\begin{bmatrix} U_k' \\ U_{n-k}' \end{bmatrix} \in \mathbb{O}_n$, then it can left-multiply each term in the left $\|\ldots\|$ of the second probability without changing the equality. As a result, the derivation continues

[8] That the resulting $\hat{M}$ was chosen *uniformly* from $\mathbb{M}(X_q, Y_q)$ could be more rigorously justified using left-invariance of probability measures and the Haar probability measure over $\mathbb{O}_{n-k}$. But, such a discussion is not relevant to this paper and is omitted.

[9] We define $\mathbb{O}_0$ to contain a single, empty matrix. And, for $P \in \mathbb{O}_0$, we define $V_{n-k} P U_{n-k}'$ to be the $n \times n$ zero matrix.

$$\begin{aligned}\cdots &= Pr\left(\left\|\begin{bmatrix} U'_k x_{\hat{i}} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ P'V'_{n-k}M_T x_{\hat{j}} \end{bmatrix} - \begin{bmatrix} U'_k x_{\hat{j}} \\ U'_{n-k}x_{\hat{j}} \end{bmatrix}\right\| \leq \|x_{\hat{j}}\|\epsilon\right) \\ &= Pr(\|P'V'_{n-k}M_T x_{\hat{j}} - U'_{n-k}x_{\hat{j}}\| \leq \|x_{\hat{j}}\|\epsilon).\end{aligned}$$

Since $Col(M_T U_{n-k}) = Col(V_{n-k})$, then there exists $(n-k)\times(n-k)$ matrix B such that $M_T U_{n-k}B = V_{n-k}$. It follows that (i) $V'_{n-k} = B'U'_{n-k}M'_T$, (ii) $B = U'_{n-k}M'_T V_{n-k}$. Thus, B is orthogonal.[10] Using (i), the derivation continues

$$\begin{aligned}\cdots &= Pr(\|P'B'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon) \quad (1)\\ &= Pr(\|P'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon) \quad (2)\end{aligned}$$

where the second equality is due to the fact that $B' \in \mathbb{O}_{n-k}$, and thus $(P'B')$ can be regarded as having been uniformly chosen from $\mathbb{O}_{n-k}$ just like P' (a rigorous proof of the second equality is provided in Appendix A). Putting the whole derivation together,

$$\rho(x_{\hat{j}}, \epsilon) = Pr(P \text{ uniformly chosen from } \mathbb{O}_{n-k} \text{ satisfies } \|P'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon). \quad (3)$$

Let $S_{n-k}(\|U'_{n-k}x_{\hat{j}}\|)$ denote the hyper-sphere in $\Re^{n-k}$ with radius $\|U'_{n-k}x_{\hat{j}}\|$ and centered at the origin. Since P is chosen uniformly from $\mathbb{O}_{n-k}$, then any point on the surface of $S_{n-k}(\|U'_{n-k}x_{\hat{j}}\|)$ is equally likely to be $P'(U'_{n-k}x_{\hat{j}})$. Let $S_{n-k}(U'_{n-k}x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ denote the "hyper-sphere cap" consisting of all points in $S_{n-k}(\|U'_{n-k}x_{\hat{j}}\|)$ with distance from $U'_{n-k}x_{\hat{j}}$ no greater than $\|x_{\hat{j}}\|\epsilon$. Therefore, (3) becomes

$$\begin{aligned}\rho(x_{\hat{j}}, \epsilon) &= Pr(\text{a uniformly chosen point on } S_{n-k}(\|U'_{n-k}x_{\hat{j}}\|) \text{ is also in } S_{n-k}(U'_{n-k}x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon))\\ &= \frac{SA(S_{n-k}(U'_{n-k}x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon))}{SA(S_{n-k}(\|U'_{n-k}x_{\hat{j}}\|))} \quad (4)\end{aligned}$$

where $SA(.)$ denotes the surface area of a subset of a hyper-sphere.[11] Based on equations (4), we prove, in Appendix A, the following closed form expression, for $\rho(x_{\hat{j}}, \epsilon)$, where, $\Gamma(.)$ denotes the standard gamma function, $ac_{[]-1}(x)$ denotes $arccos\left(\left[\frac{\|x_{\hat{j}}\|\epsilon}{\|U'_{n-k}x_{\hat{j}}\|\sqrt{2}}\right]^2 - 1\right)$, and $ac_{1-[]}(x)$ denotes $arccos\left(1 - \left[\frac{\|x_{\hat{j}}\|\epsilon}{\|U'_{n-k}x_{\hat{j}}\|\sqrt{2}}\right]^2\right)$.

[10] $B'B = B'U'_{n-k}M'_T V_{n-k} = V'_{n-k}V_{n-k} = I_{n-k}$.

[11] $S_1(\|U'_1 x_{\hat{j}}\|)$ consists of two points. We define $\frac{SA(S_1(U'_1 x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon))}{SA(S_1(\|U'_1 x_{\hat{j}}\|))}$ as 0.5 if $S_1(U'_1 x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ is one point, and as 1 otherwise. Moreover, we define $\frac{SA(S_0(U'_0 x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon))}{SA(S_0(\|U'_0 x_{\hat{j}}\|))}$ as 1.

$$\rho(x_{\hat{j}}, \epsilon) = \begin{cases} 1 & \text{if } n-k=0; \\ 1 & \text{if } \|x_{\hat{j}}\|\epsilon \geq \|U'_{n-k}x_{\hat{j}}\|2 \text{ and } n-k \geq 1; \\ 0.5 & \text{if } \|x_{\hat{j}}\|\epsilon < \|U'_{n-k}x_{\hat{j}}\|2 \text{ and } n-k=1; \\ 1-(1/\pi)ac_{[]-1}(x) & \text{if } \|U'_{n-k}x_{\hat{j}}\|\sqrt{2} < \|x_{\hat{j}}\|\epsilon < \|U'_{n-k}x_{\hat{j}}\|2 \text{ and } n-k=2; \\ 1-\frac{(n-k-1)\Gamma([n-k+2]/2)}{(n-k)\sqrt{\pi}\Gamma([n-k+1]/2)}\int_{\theta_1=0}^{ac_{[]-1}(x)} sin^{n-k-1}(\theta_1)\,d\theta_1 & \text{if } \|U'_{n-k}x_{\hat{j}}\|\sqrt{2} < \|x_{\hat{j}}\|\epsilon < \|U'_{n-k}x_{\hat{j}}\|2 \text{ and } n-k \geq 3; \\ (1/\pi)ac_{1-[]}(x) & \text{if } \|x_{\hat{j}}\|\epsilon \leq \|U'_{n-k}x_{\hat{j}}\|\sqrt{2} \text{ and } n-k=2; \\ \frac{(n-k-1)\Gamma([n-k+2]/2)}{(n-k)\sqrt{\pi}\Gamma([n-k+1]/2)}\int_{\theta_1=0}^{ac_{1-[]}(x)} sin^{n-k-1}(\theta_1)\,d\theta_1 & \text{if } \|x_{\hat{j}}\|\epsilon \leq \|U'_{n-k}x_{\hat{j}}\|\sqrt{2} \text{ and } n-k \geq 3. \end{cases} \tag{5}$$

Comment: it can be shown that $\|U'_{n-k}x_{\hat{j}}\|$ is the distance from $x_{\hat{j}}$ to its closest point in $Col(X_q)$ (the column space of X_q). Thus, the sensitivity of a tuple to breach is dependent upon its length relative to its distance to the column space of X_q. In particular, if the distance from $x_{\hat{j}}$ to the column space of X_q is sufficiently small, less than $(\|x_{\hat{j}}\|\epsilon)/2$, then the breach probability is one from the second case in equation (5).

Recall that the attacker seeks to use the closed-form expressions for $\rho(x_{\hat{j}}, \epsilon)$ to decide for which $q < j \leq m$ does $\hat{x} = \hat{M}'y_j$ produce the best estimation of $x_{\hat{j}}$. This is naturally done by choosing j to maximize $\rho(x_{\hat{j}}, \epsilon)$. To allow for this, observe that $\|x_{\hat{j}}\|\epsilon$ and $\|U'_{n-k}x_{\hat{j}}\|$ equal[12] $\|y_j\|\epsilon$ and $\|V'_{n-k}y_j\|$, respectively, which are known to the attacker. Therefore, (5) can be rewritten as follows, where $ac_{[]-1}(y)$ denotes $arccos\left(\left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2 - 1\right)$, and $ac_{1-[]}(y)$ denotes $arccos\left(1 - \left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2\right)$.

$$\rho(x_{\hat{j}}, \epsilon) = \begin{cases} 1 & \text{if } n-k=0; \\ 1 & \text{if } \|y_j\|\epsilon \geq \|V'_{n-k}y_j\|2 \text{ and } n-k \geq 1; \\ 0.5 & \text{if } \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k=1; \\ 1-(1/\pi)ac_{[]-1}(y) & \text{if } \|V'_{n-k}y_j\|\sqrt{2} < \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k=2; \\ 1-\frac{(n-k-1)\Gamma([n-k+2]/2)}{(n-k)\sqrt{\pi}\Gamma([n-k+1]/2)}\int_{\theta_1=0}^{ac_{[]-1}(y)} sin^{n-k-1}(\theta_1)\,d\theta_1 & \text{if } \|V'_{n-k}y_j\|\sqrt{2} < \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k \geq 3; \\ (1/\pi)ac_{1-[]}(y) & \text{if } \|y_j\|\epsilon \leq \|V'_{n-k}y_j\|\sqrt{2} \text{ and } n-k=2; \\ \frac{(n-k-1)\Gamma([n-k+2]/2)}{(n-k)\sqrt{\pi}\Gamma([n-k+1]/2)}\int_{\theta_1=0}^{ac_{1-[]}(y)} sin^{n-k-1}(\theta_1)\,d\theta_1 & \text{if } \|y_j\|\epsilon \leq \|V'_{n-k}y_j\|\sqrt{2} \text{ and } n-k \geq 3. \end{cases} \tag{6}$$

Now we put together all the parts and provide the pseudo-code of the full known input attack algorithm (Algorithm 5). Before doing so, first note that U_k, U_{n-k}, V_k, and V_{n-k} can be computed from X_q and Y_q using standard procedures [40]. Second, $M_T U_k = Y_q A$ where A is an $q \times k$ matrix that can be computed[13] from U_k and X_q. Third, a recursive procedure for computing (6) is described in Appendix A.

Comment: The ϵ-privacy breach probability $\rho(\epsilon)$ equals $\max_{q<j\leq m}\rho(x_{\hat{j}}, \epsilon)$.

Example revisited - part 2: consider the dataset and its perturbed version illustrated in Figure 1 with known original tuples x_1, x_2, x_3. Part 1 of this example showed how Algorithm 1 inferred the following mappings to perturbed tuples $x_1 \mapsto y_1$ and $x_2 \mapsto y_2$, hence $X_q = [x_1 x_2]$ and $Y_q = [y_1 y_2]$. Consider perturbed tuple y_4. The Known Input Attack will compute $\hat{x}$ an estimate of the original tuple $x_{\hat{4}}$ associated with y_4 ($x_{\hat{4}} = x_4$ in this case) and $\rho(x_{\hat{4}}, \epsilon)$, the ϵ-privacy breach probability. Since x_1 and x_2 are linearly dependent, $k = 1$, so, $n - k = 1$ and the second or third

[12] $M_T x_{\hat{j}} = y_j$, so, $\|x_{\hat{j}}\| = \|M_T x_{\hat{j}}\| = \|y_j\|$. Moreover, as shown earlier, there exists $B \in \mathbb{O}_{n-k}$ such that $V'_{n-k} = B'U'_{n-k}M'_T$. Thus, $\|U'_{n-k}x_{\hat{j}}\| = \|B'U'_{n-k}M'_T M_T x_{\hat{j}}\| = \|V'_{n-k}y_j\|$.

[13] Since $Col(U_k) = Col(X_q)$, then by solving k systems of linear equations (one for each column of U_k), a $q \times k$ matrix A can be computed such that $X_q A = U_k$.

Algorithm 5 Known Input Attack Algorithm

Inputs: Y, $\epsilon \geq 0$, and X_a.
Outputs: $a < j \leq m$ and $\hat{x} \in \Re^n$ the corresponding estimate of $x_{\hat{j}}$.
1: Compute $Y_q = M_T X_q$ (where $1 \leq q \leq a$) using Algorithm 1.
2: Compute $U_k, V_k, U_{n-k}, V_{n-k}$, and $M_T U_k$ as described earlier.
3: For each $q < j \leq m$ do
4: Compute $\rho(x_{\hat{j}}, \epsilon)$ using (6) as described in Appendix A.
5: End For.
6: Choose the j from the previous loop producing the largest $\rho(x_{\hat{j}}, \epsilon)$.
7: Choose $\hat{M}$ uniformly from $\mathbb{M}(X_q, Y_q)$ by applying Algorithm 4.
8: Set $\hat{x} \leftarrow \hat{M}' y_j$.

cases of Equation 6 apply. It can be shown that $V'_{n-k} = V'_1 = [0, 1]$. So, $\|y_4\|\epsilon = 2\epsilon$ and $\|V'_{n-k} y_4\|2 = 4$. Therefore, if $\epsilon \geq 2$, the second case applies and $\rho(x_{\hat{4}}, \epsilon) = 1$, else, the third case applies and $\rho(x_{\hat{4}}, \epsilon) = 0.5$.

There are only two Euclidean distance preserving transformations fixing the origin that satisfy the input-output constraints $x_1 \mapsto y_1$ and $x_2 \mapsto y_2$: the 90-degree clockwise rotation (the actual perturbation applied) and the 90-degree counter-clockwise rotation, these are the elements of $\mathbb{M}(X_2, Y_2)$. So $\hat{x}$ is chosen randomly between the inverse of these transformations applied to y_4 resulting in $\hat{x} = [2, 0]'$ or $[-2, 0]'$. If $\epsilon \geq 2$, then either of these choices represent an $\epsilon-$privacy breach so $\rho(x_{\hat{4}}, \epsilon) = 1$. If $\epsilon < 2$, then only one of these choices, $[2, 0]'$, represent a breach so $\rho(x_{\hat{4}}, \epsilon) = 0.5$. □

4.5. Known Input Attack on General Distance-Preserving Data Perturbation

Previously, we considered the case where the data perturbation is assumed to be orthogonal (does not involve a fixed translation, $v_T = 0$). Now we briefly discuss how the attack technique and its analysis can be extended to arbitrary Euclidean distance-preserving perturbation ($v_T \neq 0$).

Extending the algorithms for inferring π_a: Since the length of the private data tuples may not be preserved, then the definition of validity in Section 4.1 must be changed: β on I is *valid* if $\forall i, j \in I$, $\|x_i - x_j\| = \|y_{\beta(i)} - y_{\beta(j)}\|$. As well, the definition of $C(I_1, \beta_1, \hat{i})$ (given $I_1 \subseteq I$, β_1 a valid assignment on I_1, and $\hat{i} \in (I \setminus I_1)$), must change: the set of all $j \in (\{1, \ldots, m\} \setminus \beta_1(I_1))$ such that for all $i_1 \in I_1$, $\|x_{i_1} - x_{\hat{i}}\| = \|y_{\beta_1(i_1)} - y_j\|$. With these changes, Algorithms 1, 2, and 3 work correctly as stated.

Extending the known input attack: The basic idea is simple and relies on the fact that the same v_T is added to all tuples in the perturbation of X_q. Fix one tuple, say x_1 and y_1, and consider the following differences $x_1^- = (x_q - x_1)$, …, $x_{q-1}^- = (x_q - x_{q-1})$ and $y_1^- = (y_q - y_1)$, …, $y_{q-1}^- = (y_q - y_{q-1})$. Let X_{q-1}^- denote the matrix with columns $x_1^-, \ldots, x_{q-1}^-$ and Y_{q-1}^- denote the matrix with columns $y_1^-, \ldots, y_{q-1}^-$. Observe that $Y_{q-1}^- = M_T X_{q-1}^-$, hence, the attack and its analysis from the orthogonal data perturbation case can be applied. The details are straight-forward and are omitted for brevity. However, a caveat is in order. The attack depends on the choice of the tuple to fix. Therefore, the attacker examines them all and chooses the highest privacy breach probability.

5. Experiments and Discussion

The experiments are designed to assess the computational efficiency of the overall known input attack and its effectiveness at breaching privacy. We performed two sets of experiments: (a)

those involving only the known input attack, and (b) those comparing the known input attack with the attack of Kaplan *et al.* [35]. In both sets of experiments, we used two datasets as the original, private data tuples X: 1) a 100,000 tuple synthetic dataset generated from a 100-variate Gaussian distribution[14]; 2) the Letter Recognition dataset, 20,000 tuples and 16 numeric attributes, from UCI machine learning repository [42] – we removed tuples which were duplicated over the numeric attributes yielding a final dataset of 18,668 tuples. The attacks were implemented in Matlab 7 (R14) and all experiments were carried out on a Thinkpad laptop with 1.83GHz Intel Core 2 CPU, 1.99GB RAM, and WindowsXP system. We did not compare our attack technique against the ICA-based attack in [38] and the known sample attack in [10] because the extremely small size of the known inputs will render these attacks ineffective. In all Figures, the error bars show one standard deviation above and below the average.

5.1. Experiments Only Involving the Known Input Attack

The first experiment fixes X and its perturbed version Y, but changes the number of known input tuples, a. It proceeds by carrying out ten trials as follows. Select a linearly independent tuples randomly from X (these become the know inputs). Use Algorithm 1 to compute I, the maximal uniquely valid assignment. Use steps 2-5 in Algorithm 5 to compute the $\rho(\epsilon)$, the ϵ-privacy breach probability (a closed-form was given immediately above Algorithm 5).

To measure the accuracy of the attack, we report the average of $\rho(\epsilon)$ and $|I|$ over all ten trials. To measure the efficiency, we report the average time taken to compute I (the rest is ignored as the overall attack computation time is dominated by Algorithm 1). In Figures 2 and 3, results are shown with $\epsilon = 0.15$. In Figure 4, accuracy results are shown with varying ϵ and a fixed at four.

The second experiment fixes the number of known input tuples (and ϵ at 0.15) but changes the size of the original data X in order to assess the computational efficiency of the attack. For the Gaussian data, it uses the first k tuples as X where k takes a value in $\{10000, 20000, \ldots, 100000\}$. Then, the attack proceeds by carrying out the following operations ten times. Select $a = 50$ linearly independent tuples randomly from X and use Algorithm 1 to compute the maximal uniquely valid assignment I. The average time taken to compute I is given in Figure 5 top. For the Letter Recognition data, k takes a value in $\{2000, 4000, \ldots, 18000\}$ and the attack randomly select $a = 10$ linearly independent tuples as the known inputs. The average time taken to find I is given in Figure 5 bottom.

Regarding the known input attack accuracy, the linking phase of the attack (Algorithm 1), exhibits excellent performance. For synthetic data, its performance is perfect in that all known input tuples have their corresponding perturbed tuple inferred (see Figure 2 top). For real data, its performance is nearly perfect – see Figure 3 top. As expected, $\rho(\epsilon)$ approaches one as a increases see Figures 2 and 3 bottom. Interestingly on the synthetic dataset, the transition from $\rho(\epsilon) = 0 \rightarrow 1$ occurs very sharply around $a = 60$. Moreover, on the real dataset, $\rho(\epsilon) = 1$ with a as small as 4 (and we also observe in Figure 4 that the probability remains fairly high for ϵ as small as 0.07).

Regarding computational efficiency, the algorithm appears to require quite reasonable time in all cases observed, *e.g.* less that 450 seconds on the synthetic dataset with 100 known tuples (see Figure 2 middle) and less than 45 seconds on the real dataset with 16 known inputs (see Figure 3

[14]The mean vector is specified by independently generating 100 numbers from a univariate Gaussian with mean zero and variance one. The covariance matrix is specified by (i) independently generating 100 data tuples each with 100 independently generated entries a from a univariate Gaussian with mean zero and variance one, (ii) computing the empirical covariance of this 100 tuple dataset.

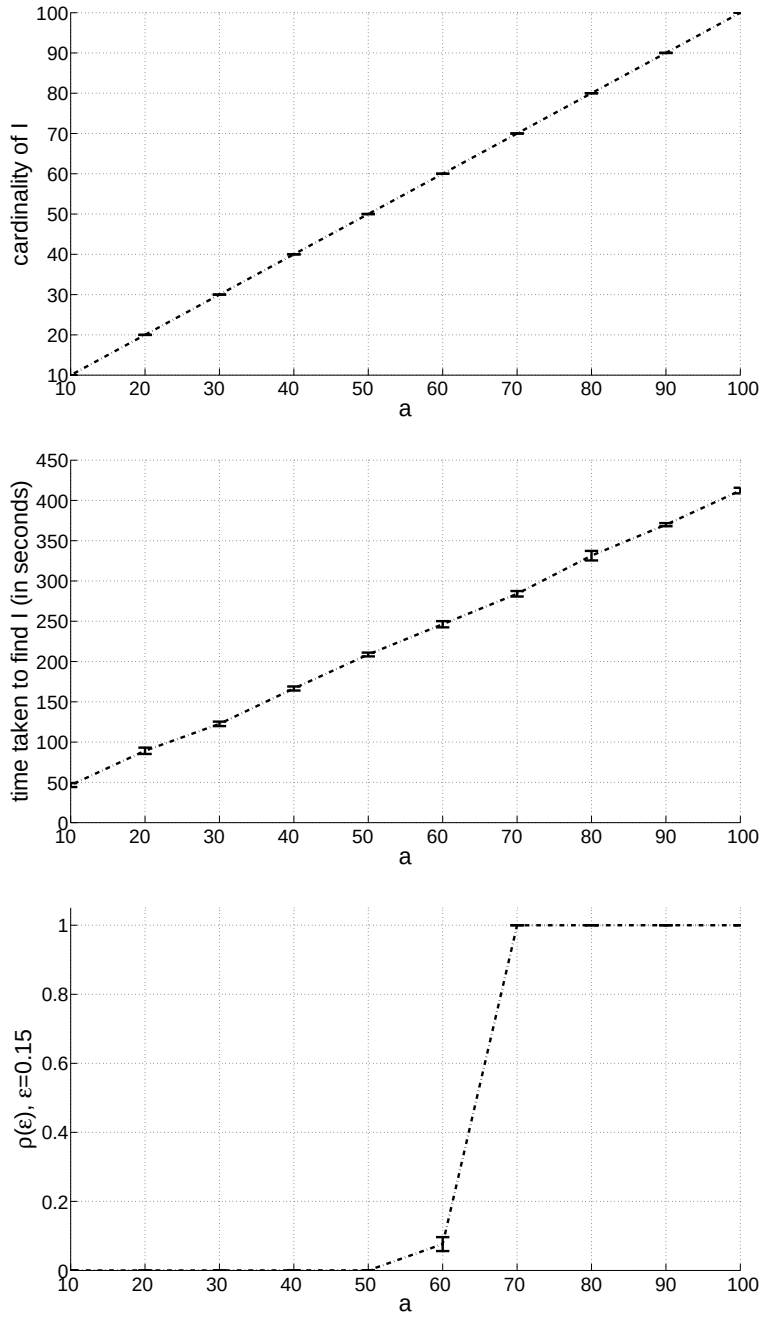


Figure 2: Known input attack on Gaussian data with different number of known inputs and $\epsilon = 0.15$.

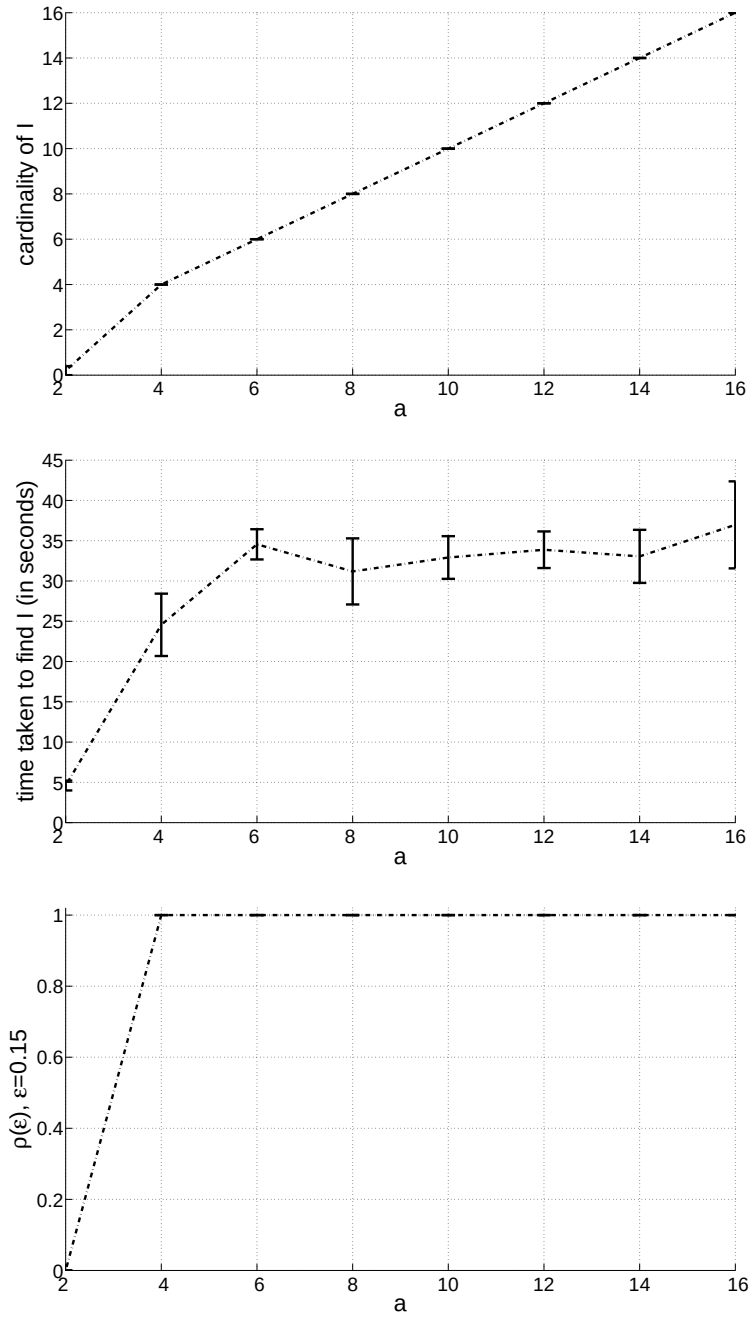


Figure 3: Known input attack on Letter Recognition data with different number of known inputs and $\epsilon = 0.15$.

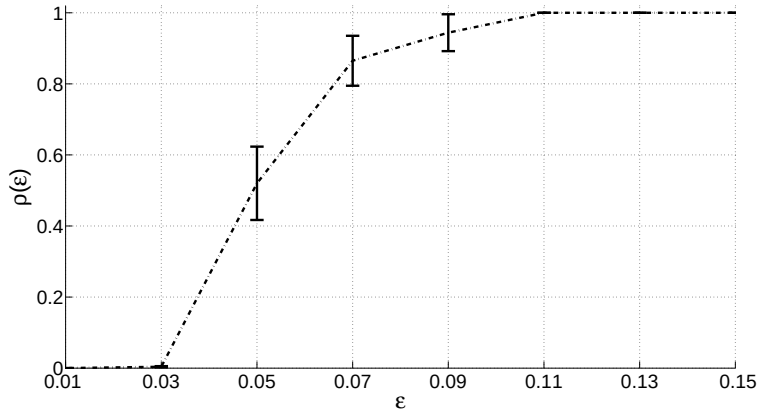


Figure 4: Known input attack on Letter Recognition data with fixed data size, fixed number of known inputs $a = 4$, but varying ϵ.

middle). With respect to known input set size (a), the average computation time exhibits a linear (synthetic data) or slower (real data) trend (see Figure 2 and Figure 3 middle). With respect to dataset size (number of private data tuples), the average computation time exhibits a clear linear trend for both synthetic and real data (see Figure 5). These results demonstrate that, despite the high worst-case computational complexity, the computation times on both real and synthetic data are quite reasonable.

The experimental results support the conclusion that the attack can breach privacy in plausible situations. For example, on the 16-dimensional, 18688 tuple real dataset, the known input attack achieves a privacy breach with probability one using four known inputs and less than 30 seconds of run-time.

5.2. Experiments Comparing the Known Input-Output Attack with Kaplan's Attack

We compare the accuracy of the attacks with respect to an attacker's goal of producing a single perturbed tuple and an estimate of its unperturbed counterpart. Since Kaplan's attack does not provide the attacker with any means to know how good the estimate is, the attacker has no reason to choose one perturbed tuple over another, hence, we assume the attacker picks randomly. On the other hand, our attack provides the attacker with breech proabilities, so, the attacker chooses the perturbed tuple to maximize the breech probability (as done in Algorithm 5).

The experiments proceed as follows. X and Y are fixed and a, the number of known input tuples, is varied. 100 trials are carried out as follows. Select a linearly independent tuples randomly from X (these become the know inputs) and do both of the following. (i) Choose a tuple y_ℓ randomly from Y whose unperturbed counterpart $x_{\hat{\ell}}$ in X is not among the a known inputs. Use Kaplan's attack[15] to produce an estimate, $\hat{x}$, of $x_{\hat{\ell}}$. Record the Euclidean relative error of the estimate, $||x_{\hat{\ell}} - \hat{x}||/||x_{\hat{\ell}}||$. (ii) Use our Algorithm 5[16] to choose the tuple y_j from Y with

[15]With a learning rate of 0.05 and 500 iterations, values observed empirically to produce the best results

[16]With $\epsilon = 0.15$ and 0.1 for the Gaussian and Letter data, respectively.

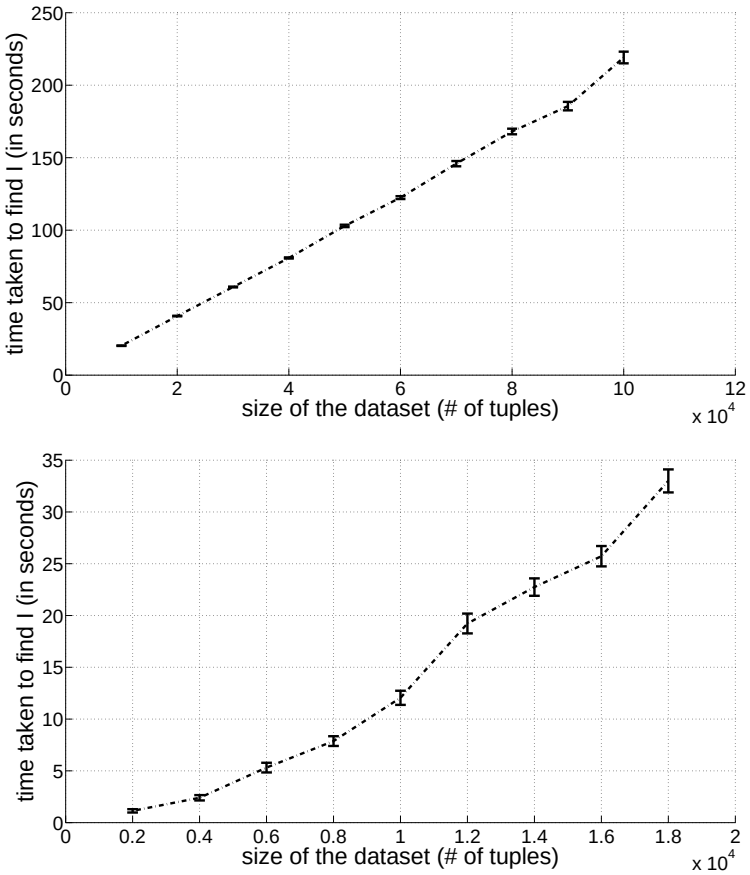


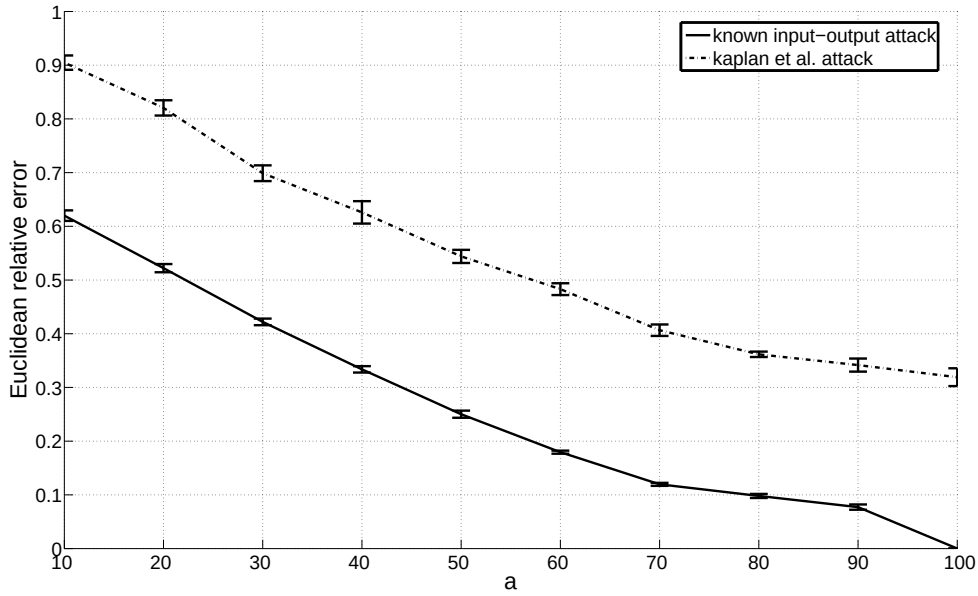
Figure 5: Known input attack on Gaussian (left) and Letter Recognition data (right) with varying size, but fixed number of known inputs $a = 50, 10$ (respectively) and fixed $\epsilon = 0.15$.

maximum $\epsilon-$privacy breech probability and whose unperturbed counterpart $x_{\hat{j}}$ does not appear among the a known inputs, then produce an estimate $\hat{x}$ of $x_{\hat{j}}$. Record the Euclidean relative error of the estimate: $\|x_{\hat{j}} - \hat{x}\|/\|x_{\hat{j}}\|$.

Figure 6 shows the average relative error of the attacks on both datasets. From the Figure it is clear that our approach allows the attacker to produce a significantly more accurate estimate. We do not provide a figure comparing the run-times of the attacks because Kaplan's computes an estimate from only one perturbed tuple while ours, in effect, computes an estimate from all perturbed tuples. However, the time required by our algorithm to produce an estimate from a single, randomly chosen, perturbed tuple is 100 to 1000 times faster than Kaplan's.

6. Conclusion

We examined the vulnerability of Euclidean distance-preserving data perturbation when a small set of original data tuples are known to the attacker. We developed a stochastic technique



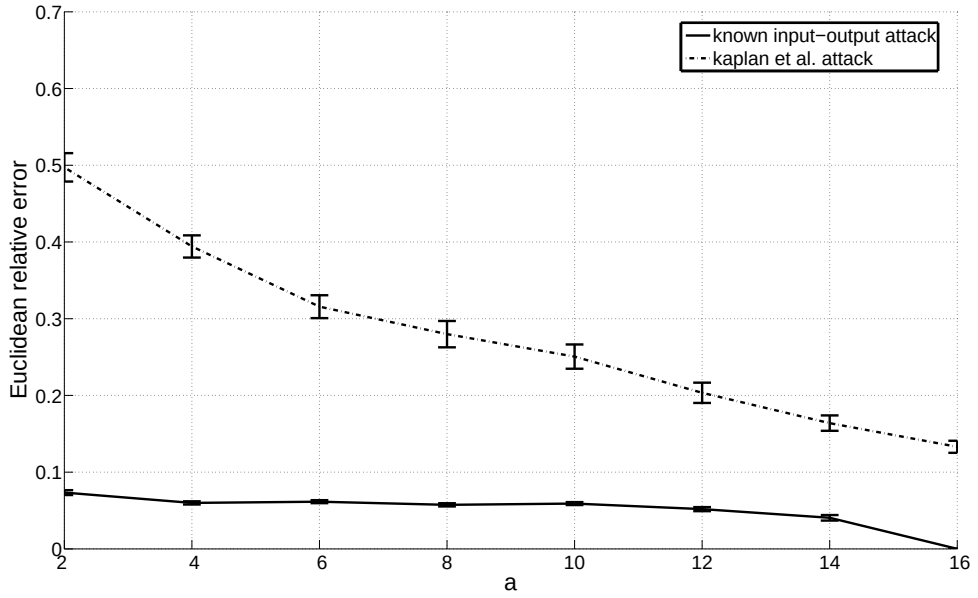


Figure 6: A comparison of Kaplan's attack and our Algorithm 5) in terms of average relative error over 100 trials. The top and bottom charts show the average error on the Gaussian and Letter recognition datasets, respectively.

allowing the attacker to estimate, for each perturbed tuple, the original unknown data tuple and calculate the probability that the estimation results in a privacy breach. For perturbations which fix the origin, this probability is dependent on the length of the original tuple relative to its distance from the column space of the known inputs. Therefore, the probability increases as the number of known original tuples does, reaching one when the number of linearly independent known original tuples reaches the number of data dimensions. The assumption of fixing the origin can be dropped, resulting in a slightly more complicated breach probability calculation. Our experiments on real and synthetic data showed that even with the number of known original tuples significantly smaller than the number of data dimensions, privacy is breached with high probability. For example, on a real 16-dimensional dataset, 4 known original tuples is enough for the attacker to estimate an unknown original tuple with less than 7% error with probability exceeding 0.8.

We conclude the paper by pointing to an interesting direction for future work, extending techniques in this paper to apply to random projection data perturbation: $Y = \ell^{-1/2}\hat{R}X$ where $\hat{R}$ is an $\ell \times n$ matrix with each entry generated independently and from a standard normal distribution (this type of data perturbation for $\ell \leq n$ was discussed in [37]). It can be shown that matrix R is orthogonal on expectation and the probability of orthogonality approaches one exponentially fast with ℓ. By increasing ℓ, the data owner can guarantee that distances are preserved with arbitrarily high probability. However, such an increase intuitively would seem to increase the vulnerability with respect to a known input attack. Some preliminary results along these lines can be found in [43].

References

[1] R. Agrawal, R. Srikant, Privacy preserving data mining, in: Proc. ACM SIGMOD, 2000, pp. 439–450.

[2] H. Kargupta, S. Datta, Q. Wang, K. Sivakumar, On the privacy preserving properties of random data perturbation techniques, in: Proc. IEEE ICDM, 2003.

[3] Z. Huang, W. Du, B. Chen, Deriving private information from randomized data, in: Proc. ACM SIGMOD, 2005, pp. 37–48.

[4] S. Guo, X. Wu, Y. Li, Determining error bounds for spectral filtering based reconstruction methods in privacy preserving data mining, Knowledge and Information Systems 17 (2) (2008) 217–240.

[5] L. Sweeney, k-anonymity: a model for protecting privacy, International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10 (5) (2002) 557–570.

[6] N. R. Adam, J. C. Worthmann, Security-control methods for statistical databases: a comparative study, ACM Computing Surveys 21 (4) (1989) 515–556.

[7] K. Chen, L. Liu, Privacy preserving data classification with rotation perturbation, in: Proc. IEEE ICDM, 2005, pp. 589–592.

[8] S. R. M. Oliveira, O. R. Zaïane, Privacy preserving clustering by data transformation, in: Proc. Brazilian Symposium on Databases, 2003, pp. 304–318.

[9] S. R. M. Oliveira, O. R. Zaïane, Achieving privacy preservation when sharing data for clustering, in: Proceedings of the International Workshop on Secure Data Management in a Connected World, Toronto, Canada, 2004, pp. 67–82.

[10] K. Liu, C. Giannella, H. Kargupta, An attacker's view of distance preserving maps for privacy preserving data mining, in: Proc. PKDD, 2006, pp. 297–308.

[11] S. Mukherjee, Z. Chen, A. Gangopadhyay, A privacy preserving technique for euclidean distance-based mining algorithms using fourier-related transforms, The VLDB Journal 15 (4) (2006) 293–315.

[12] K. Chen, G. Sun, L. Liu, Towards attack-resilient geometric data perturbation, in: Proc. SIAM SDM, 2007.

[13] E. Turgay, T. Pedersen, Y. Saygin, E. Savas, A. Levi, Disclosure risks of distance preserving data transformations, in: Lecture Notes in Computer Science, Springer-Verlag, Vol. 5069, 2008, pp. 79–94.

[14] W. Wong, D. Cheung, B. Kao, N. Mamoulis, Secure knn computation on encrypted databases, in: Proc ACM SIGMOD, 2009, pp. 139–152.

[15] J. Han, M. Kamber, Data Mining: Concepts and Techniques, 2nd ed., The Morgan Kauffman Series in Data Management Systems, Morgan Kauffman, 1991.

[16] S. Mukherjee, M. Banerjee, Z. Chen, A. Gangopadhyay, A privacy preserving technique for distance-based classification with worst case privacy gaurantees, Data & Knowledge Engineering 66 (2) (2008) 264–288.
[17] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition, Springer, 2011, Ch. 13.3, 5th printing.
[18] A. Evfimevski, J. Gehrke, R. Srikant, Limiting privacy breaches in privacy preserving data mining, in: Proc. ACM PODS, 2003.
[19] L. Liu, M. Kantarcioglu, B. Thuraisingham, The applicability of the perturbation based privacy preserving data mining for real-world data, Data & Knowledge Engineering 65 (1) (2008) 5–21.
[20] J. J. Kim, W. E. Winkler, Multiplicative noise for masking continuous data, Tech. Rep. Statistics #2003-01, Statistical Research Division, U.S. Bureau of the Census, Washington D.C. (April 2003).
[21] M. Trottini, S. E. Fienberg, U. E. Makov, M. M. Meyer, Additive noise and multiplicative bias as disclosure limitation techniques for continuous microdata: A simulation study, Journal of Computational Methods in Sciences and Engineering 4 (2004) 5–16.
[22] P. Samarati, Protecting respondents identities in microdata release, IEEE Transactions on Knowledge and Data Engineering 13 (6) (2001) 1010–1027.
[23] K. LeFevre, D. J. DeWitt, R. Ramakrishnan, Workload-aware anonymization, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06), 2006, pp. 277–286.
[24] K. Wang, B. C. M. Fung, Anonymizing sequential releases, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06), 2006, pp. 414–423.
[25] O. Abul, F. Bonchi, M. Nanni, Never walk alone: Uncertainty for anonymity in moving objects databases, in: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE'08), 2008, pp. 376–385.
[26] C. C. Aggarwal, P. S. Yu, Privacy-Preserving Data Mining: Models and Algorithms, 1st Edition, Advances in Database Systems, Springer, 2008, Ch. 5.
[27] C. C. Aggarwal, P. S. Yu, A condensation based approach to privacy preserving data mining, in: Proc. EDBT, 2004, pp. 183–199.
[28] X.-B. Li, S. Sarkar, A tree-based data perturbation approach for privacy-preserving data mining, IEEE Transactions on Knowledge and Data Engineering 18 (9) (2006) 1278–1283.
[29] S. E. Fienberg, J. McIntyre, Data swapping: Variations on a theme by dalenius and reiss, Tech. rep., National Institute of Statistical Sciences (2003).
[30] K. Muralidhar, R. Sarathy, Data shuffling – a new masking approach for numeric data, Management Science 52 (5) (2006) 658–670.
[31] S. J. Rizvi, J. R. Haritsa, Maintaining data privacy in association rule mining, in: Proc. VLDB, 2002.
[32] V. S. Verykios, A. K. Elmagarmid, B. Elisa, Y. Saygin, D. Elena, Association rule hiding, in: IEEE Transactions on Knowledge and Data Engineering, Vol. 16, 2004, pp. 434–447.
[33] D. Ting, S. Fienberg, M. Trottini, Random orthogonal matrix masking methodology for microdata release, International Journal on Information and Computer Security 2 (1) (2008) 86–105.
[34] B. Fung, K. Wang, R. Chen, P. Yu, Privacy-preserving data publishing: A survey of recent developments, ACM Computing Surveys 42 (4) (2010) 14:1–14:53.
[35] E. Kaplan, T. Pedersen, E. Savas, Y. Saygin, Discovering private trajectories using background information, Data & Knowledge Engineering 69 (7) (2010) 723–736.
[36] A. Hyvärinen, E. Oja, Independent component analysis: Algorithms and applications, Neural Networks 13 (4) (2000) 411–430.
[37] K. Liu, H. Kargupta, J. Ryan, Random projection-based multiplicative data perturbation for privacy preserving distributed data mining, IEEE Transactions on Knowledge and Data Engineering 18 (1) (2006) 92–106.
[38] S. Guo, X. Wu, Deriving private information from arbitrarily projected data, in: Proc. PAKDD, 2007.
[39] M. Artin, Algebra, Prentice Hall, 1991.
[40] G. Strang, Linear Algebra and Its Applications (3rd Ed.), Harcourt Brace Jovanovich College Publishers, New York, 1986.
[41] R. Heiberger, Algorithm as 127: Generation of random orthogonal matrices, Applied Statistics 27 (2) (1978) 199–206.
[42] A. Frank, A. Asuncion, UCI machine learning repository (2010).
URL `http://archive.ics.uci.edu/ml`
[43] K. Liu, C. Giannella, H. Kargupta, A survey of attack techniques on privacy-preserving data perturbation methods, in: Privacy-Preserving Data Mining: Models and Algorithms, Vol. 53 of Advances in Information Security, Springer Verlag, 2008.
[44] Z. Huang, B. He, Volume of unit ball in an n-dimensional normed space and its asymptotic properties, J. Shanghai Univ. 12 (2) (2008) 107–109.

Appendix A. Supplementary Material

Appendix A.1. Known Input Attack: Proof of Theorem 4.1

Theorem 4.1: Let L be the mapping $P \in \mathbb{O}_{n-k} \mapsto M_T U_k U_k' + V_{n-k} P U_{n-k}'$. Then, L is an affine bijection from $\mathbb{O}_{n-k}$ to $\mathbb{M}(X_q, Y_q)$. And, L^{-1} is the mapping $M \in \mathbb{M}(X_q, Y_q) \mapsto V_{n-k}' M U_{n-k}$.

To prove this theorem we rely upon the following key technical result.

Lemma Appendix A.1. *Let* $\mathbb{P}$ *denote the set* $\{M_T U_k U_k' + V_{n-k} P U_{n-k}' : P \in \mathbb{O}_{n-k}\}$. *Then* $\mathbb{M}(X_q, Y_q) = \mathbb{P}$.

Proof: Let $\mathbb{M}(U_k, M_T U_k)$ denote the set of all $M \in \mathbb{O}_n$ such that $MU_k = M_T U_k$. First we show that $\mathbb{M}(X_q, Y_q) = \mathbb{M}(U_k, M_T U_k)$. Since $Col(X_q) = Col(U_k)$, then there exists $k \times p$ matrix A such that $U_k A = X_q$. Since A has k columns, then $rank(A) \leq k$. Furthermore, [40, pg. 201] implies that $k = rank(U_k A) \leq \min\{k, rank(A)\}$, thus, $rank(A) = k$. Therefore, from [40, pg. 90], A has a right inverse.

For any $M \in \mathbb{O}_n$, we have

$$\begin{aligned} M \in \mathbb{M}(X_q, Y_q) &\Leftrightarrow MU_k A = M_T U_k A \\ &\Leftrightarrow MU_k = M_T U_k. \end{aligned}$$

The last $\Leftrightarrow$ follows from the fact that A has a right inverse. We conclude that $\mathbb{M}(X_q, Y_q) = \mathbb{M}(U_k, M_T U_k)$. Now we complete the proof by showing that $\mathbb{M}(U_k, M_T U_k) = \mathbb{P}$.

(1) For any $M \in \mathbb{P}$, there exists $P \in \mathbb{O}_{n-k}$ such that $M = \{M_T U_k U_k' + V_{n-k} P U_{n-k}'\}$. We have then

$$\begin{aligned} MU_k &= M_T U_k U_k' U_k + V_{n-k} P U_{n-k}' U_k \\ &= M_T U_k. \end{aligned}$$

If we can show that M is orthogonal, then $M \in \mathbb{M}(U_k, M_T U_k)$, so, $\mathbb{P} \subseteq \mathbb{M}(U_k, M_T U_k)$, as desired. Let U denote $[U_k | U_{n-k}]$ (clearly $U \in \mathbb{O}_n$). Observe

$$\begin{aligned} M'M &= U_k U_k' M_T' M_T U_k U_k' + U_k U_k' M_T' V_{n-k} P U_{n-k}' \\ &+ U_{n-k} P' V_{n-k}' M_T U_k U_k' + U_{n-k} P' V_{n-k}' M_T U_{n-k} P U_{n-k}' \\ &= U_k U_k' + 0 + 0 + U_{n-k} U_{n-k}' \\ &= UU' = I_n. \end{aligned}$$

where the first zero in the second equality is due to the fact that $Col(M_T U_k) = Col(Y_q)$, so, $V_{n-k}' M_T U_k = 0$.

(2) Now consider $M \in \mathbb{M}(U_k, M_T U_k)$. It can be shown that $Col(V_{n-k}) = Col(MU_{n-k})$.[17] Thus, there exists $(n-k) \times (n-k)$ matrix P with $V_{n-k} P = MU_{n-k}$. Observe that

$$\begin{aligned} P'P &= P'(V_{n-k}' V_{n-k}) P \\ &= (V_{n-k} P)'(V_{n-k} P) \\ &= (MU_{n-k})'(MU_{n-k}) = I_{n-k}. \end{aligned}$$

[17] Since $(MU_{n-k})' MU_k = 0$, then $Col(MU_{n-k}) = Col_\perp(MU_k)$. Since $MU_k = M_T U_k$ and $Col(M_T U_k) = Col(Y_q)$, then it follows that $Col_\perp(MU_k) = Col_\perp(M_T U_k) = Col_\perp(Y_q) = Col(V_{n-k})$.

Thus, $P \in \mathbb{O}_{n-k}$. Moreover,

$$\begin{aligned} MU &= M[U_k|U_{n-k}] \\ &= [M_T U_k|MU_{n-k}] \\ &= [M_T U_k|V_{n-k}P]. \end{aligned}$$

Thus,

$$\begin{aligned} M &= [M_T U_k|V_{n-k}P]\begin{bmatrix} U_k' \\ U_{n-k}' \end{bmatrix} \\ &= M_T U_k U_k' + V_{n-k}PU_{n-k}'. \end{aligned}$$

Therefore, $M \in \mathbb{P}$, so, $\mathbb{M}(U_k, M_T U_k) \subseteq \mathbb{P}$, as desired. □

Now we prove Theorem 4.1.

Proof: Clearly L is an affine map. Moreover, Lemma Appendix A.1 directly implies that L maps $\mathbb{O}_{n-k}$ onto $\mathbb{M}(X_q, Y_q)$. To see that L is one-to-one, consider $P_1, P_2 \in \mathbb{O}_{n-k}$ such that $L(P_1) = L(P_2)$. By definition, $M_T U_k U_k' + V_{n-k}P_1 U_{n-k}' = M_T U_k U_k' + V_{n-k}P_2 U_{n-k}'$, thus, $V_{n-k}P_1 U_{n-k}' = V_{n-k}P_2 U_{n-k}'$. Therefore $P_1 = V_{n-k}' V_{n-k} P_1 U_{n-k}' U_{n-k} = V_{n-k}' V_{n-k} P_2 U_{n-k}' U_{n-k} = P_2$.

To complete the proof, consider $P \in \mathbb{O}_{n-k}$. We have, $V_{n-k}' L(P) U_{n-k} = V_{n-k}' M_T U_k U_k' U_{n-k} + V_{n-k}' V_{n-k} P U_{n-k}' U_{n-k} = 0 + P$. Moreover, consider $M \in \mathbb{M}(X_q, Y_q)$. By Lemma Appendix A.1, there exists $P_M \in \mathbb{O}_{n-k}$ such that $M = M_T U_k U_k' + V_{n-k} P_M U_{n-k}'$. We have $L(V_{n-k}' M U_{n-k}) = L(P_M) = M$. Therefore, the inverse of L is $M \in \mathbb{M}(X_q, Y_q) \mapsto V_{n-k}' M U_{n-k}$. □

Appendix A.2. Known Input Attack: A Rigorous Development of the Closed-Form Expression for $\rho(x_{\hat{j}}, \epsilon)$

Up to (1), we had derived the following result (for P chosen uniformly from $\mathbb{O}_{n-k}$):

$$\rho(x_{\hat{j}}, \epsilon) = Pr(\|P'B'(U_{n-k}' x_{\hat{j}}) - (U_{n-k}' x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon), \tag{A.1}$$

where $B \in \mathbb{O}_{n-k}$ and satisfies $M_T U_{n-k} B = V_{n-k}$. Now we provide a rigorous proof of (2), *i.e.* the r.h.s. above equals $Pr(\|P'(U_{n-k}' x_{\hat{j}}) - (U_{n-k}' x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon)$. To do so, we need some material from measure theory.

Because $\mathbb{O}_{n-k}$ is a locally compact topological group [39, pg. 293], it has a Haar probability measure, denoted by μ, over $\mathbb{B}$, the Borel algebra on $\mathbb{O}_{n-k}$. This is commonly regarded as the standard uniform probability measure over $\mathbb{O}_{n-k}$. Its key property is *left-invariance*: for all $\mathcal{B} \in \mathbb{B}$ and all $M \in \mathbb{O}_{n-k}$, $\mu(\mathcal{B}) = \mu(M\mathcal{B})$, *i.e.*, shifting $\mathcal{B}$ by a rigid motion does not change its probability assignment.

Let $\mathbb{O}_{n-k}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ denote the set of all $P \in \mathbb{O}_{n-k}$ such that $\|P'(U_{n-k}' x_{\hat{j}}) - (U_{n-k}' x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon)$. Let $\mathbb{O}_{n-k}^{B'}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ denote the set of all $P \in \mathbb{O}_{n-k}$ such that $\|P'B'(U_{n-k}' x_{\hat{j}}) - (U_{n-k}' x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon$.[18] By definition of μ we have,

$$\begin{aligned} \mu(\mathbb{O}_{n-k}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)) &= Pr(P \text{ uniformly chosen from } \mathbb{O}_{n-k} \text{ lies in } \mathbb{O}_{n-k}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)) \\ &= Pr(\|P'(U_{n-k}' x_{\hat{j}}) - (U_{n-k}' x_{\hat{j}})\| \leq \|x_{\hat{j}}\|\epsilon), \end{aligned}$$

[18] Since $\mathbb{O}_{n-k}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ and $\mathbb{O}_{n-k}^{B'}(U_{n-k}' x_{\hat{j}}, \|x_{\hat{j}}\|\epsilon)$ are topologically closed sets, then they are Borel subsets of $\mathbb{O}_{n-k}$, therefore, μ is defined on each of these.

and,

$$\begin{aligned}\mu(\mathbb{O}^{B'}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) &= Pr(P \text{ uniformly chosen from } \mathbb{O}_{n-k} \text{ lies in } \mathbb{O}^{B'}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) \\ &= Pr(||P'B'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})|| \leq ||x_{\hat{j}}||\epsilon),\end{aligned}$$

Therefore,

$$\begin{aligned}Pr(||P'B'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})|| \leq ||x_{\hat{j}}||\epsilon) &= \mu(\mathbb{O}^{B'}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) \\ &= \mu(B\mathbb{O}^{B'}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) \\ &= \mu(\mathbb{O}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) \\ &= Pr(||P'(U'_{n-k}x_{\hat{j}}) - (U'_{n-k}x_{\hat{j}})|| \leq ||x_{\hat{j}}||\epsilon)\end{aligned} \tag{A.2}$$

where the second equality is due to the left-invariance of μ and the third equality is due to the fact that $B\mathbb{O}^{B'}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)$ can be shown to equal $\mathbb{O}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)$.

Since the last equality above was for intuitive purposes only, we will ignore it in completing the derivation of a closed form expression. (A.1) and (A.2) imply

$$\rho(x_{\hat{j}}, \epsilon) = \mu(\mathbb{O}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)).$$

Recall that $S_{n-k}(||U'_{n-k}x_{\hat{j}}||)$ denotes the hyper-sphere in $\Re^{n-k}$ with radius $||U'_{n-k}x_{\hat{j}}||$ and centered at the origin and $S_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)$ denotes the points contained by $S_{n-k}(||U'_{n-k}x_{\hat{j}}||)$ whose distance from $U'_{n-k}x_{\hat{j}}$ is no greater than $||x_{\hat{j}}||\epsilon$. Using basic principles from measure theory, it can be shown that[19]

$$\mu(\mathbb{O}_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)) = \frac{SA(S_{n-k}(U'_{n-k}x_{\hat{j}}, ||x_{\hat{j}}||\epsilon))}{SA(S_{n-k}(||U'_{n-k}x_{\hat{j}}||))}$$

We have arrived at Equation (4) from Section 4.4. Next, we derive the desired closed-form expression (5). To simplify exposition, we prove the following result for $m \geq 0$, $z \in \Re^m$, and $c \geq 0$ (by plugging in $m = n - k$, $z = U'_{n-k}x_{\hat{j}}$, and $c = ||x_{\hat{j}}||\epsilon$, (5) follows).

$$\frac{SA(S_m(z,c))}{SA(S_m(||z||))} = \begin{cases} 1 & \text{if } m = 0; \\ 1 & \text{if } c \geq ||z||2 \text{ and } m \geq 1; \\ 0.5 & \text{if } c < ||z||2 \text{ and } m = 1; \\ 1 - (1/\pi)arccos([c/(||z||\sqrt{2})]^2 - 1) & \text{if } ||z||\sqrt{2} < c < ||z||2 \text{ and } m = 2; \\ 1 - \frac{(m-1)\Gamma([m+2]/2)}{m\sqrt{\pi}\Gamma([m+1]/2)} \int_{\theta_1=0}^{arccos([c/(||z||\sqrt{2})]^2-1)} sin^{m-1}(\theta_1)\, d\theta_1 & \text{if } ||z||\sqrt{2} < c < ||z||2 \text{ and } m \geq 3; \\ (1/\pi)arccos(1 - [c/(||z||\sqrt{2})]^2) & \text{if } c \leq ||z||\sqrt{2} \text{ and } m = 2; \\ \frac{(m-1)\Gamma([m+2]/2)}{m\sqrt{\pi}\Gamma([m+1]/2)} \int_{\theta_1=0}^{arccos(1-[c/(||z||\sqrt{2})]^2)} sin^{m-1}(\theta_1)\, d\theta_1 & \text{if } c \leq ||z||\sqrt{2} \text{ and } m \geq 3. \end{cases} \tag{A.3}$$

Before proving (A.3) we establish:

[19] $S_1(||U'_1x_{\hat{j}}||)$ consists of two points. Recall that we define $\frac{SA(S_1(U'_1x_{\hat{j}}, ||x_{\hat{j}}||\epsilon))}{SA(S_1(||U'_1x_{\hat{j}}||))}$ as 0.5 if $S_1(U'_1x_{\hat{j}}, ||x_{\hat{j}}||\epsilon)$ is one point, and as 1 otherwise. Moreover, we define $\frac{SA(S_0(U'_0x_{\hat{j}}, ||x_{\hat{j}}||\epsilon))}{SA(S_0(||U'_0x_{\hat{j}}||))}$ as 1.

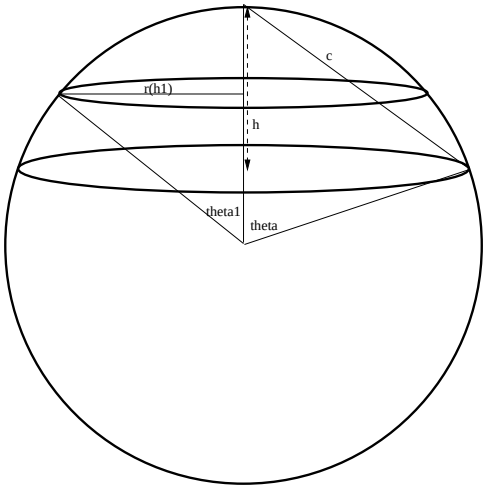


Figure A.7: The hyper-sphere $S_m(||z||)$ and two "north pole" caps ($c \leq ||z|| \sqrt{2}$).

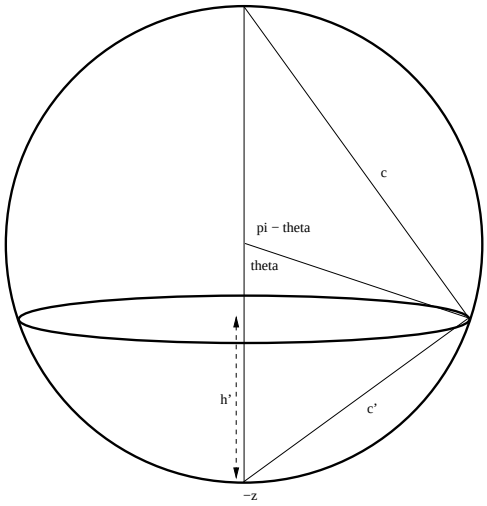


Figure A.8: The hyper-sphere $S_m(||z||)$ and one "south pole" cap ($||z|| \sqrt{2} < c < ||z||2$).

$$\text{For } b \geq 2 \text{ and } r > 0,\ SA(S_b(r)) = \frac{br^{b-1}\pi^{b/2}}{\Gamma((b+2)/2)}. \tag{A.4}$$

Indeed, with $Vol(.)$ denoting volume, it can be shown that $SA(S_b(r)) = \frac{dVol(S_b(r))}{dr} = Vol(S_b(1))\frac{dr_b}{dr}$ $= \frac{\pi^{b/2} b r^{b-1}}{\Gamma((b+2)/2)}$. The last equality follows from [44]. Now we return to proving (A.3).

If $m = 0$, then the surface area ratio equals 1 by definition. If $c \geq ||z||2$ and $m \geq 1$, then the ratio equals 1 since $S_m(z, c) = S_m(||z||)$. If $c < ||z||2$ and $m = 1$, then, the ratio equals 0.5 since $S_1(z, c) = \{z\}$ and $S_1(||z||) = \{z, -z\}$. For the remainder of the derivation, we assume that $m \geq 2$ and, without loss of generality, z is at the "north pole" of the hyper-sphere $S_m(||z||)$, *i.e.* z $= (1, 0, 0, \cdots, 0)$.

Case $c \leq ||z|| \sqrt{2}$: The set of points on $S_m(||z||)$ whose distance from z equals c is the intersection of $S_m(||z||)$ with the hyper-plane whose perpendicular to z is of length h as seen in Figure A.7. Thus, $S_m(z, c)$ are all those points on $S_m(||z||)$ not below that hyper-plane.

Sub-case $m = 2$: Since $S_2(||z||)$ is an ordinary circle, then the angle θ in Figure A.7 determines the surface area ratio as follows $\frac{(2\theta/2\pi)SA(S_2(||z||))}{SA(S_2(||z||))} = \theta/\pi$. Moreover, since θ is the top angle of an isosceles triangle with sides of length $||z||$ and base of length c, then $sin(\theta/2) = c/(2||z||)$. The half-angle formula implies that $\theta = arccos(1 - [c/(||z|| \sqrt{2})]^2)$. Therefore, as desired,

$$\frac{SA(S_2(z, c))}{SA(S_2(||z||))} = (1/\pi) arccos(1 - [c/(||z|| \sqrt{2})]^2). \tag{A.5}$$

Sub-case $m \geq 3$: Here, computing the surface area ratio is more complicated and requires an appeal to the integral definition of the cap surface area. Consider the intersection of $S_m(||z||)$ with the hyper-plane whose perpendicular to z is of length $0 \leq h_1 \leq h$ as seen in Figure A.7. The surface area of this intersection equals the surface area of $S_{m-1}(r(h_1))$. Thus, (A.4) implies

$$\begin{aligned} SA(S_m(z,c)) &= \int_{h_1=0}^{h} SA(S_{m-1}(r(h_1)))\,dh_1 \\ &= \left(\frac{(m-1)\pi^{(m-1)/2}}{\Gamma((m+1)/2)}\right)\int_{h_1=0}^{h}(r(h_1))^{m-2}\,dh_1. \end{aligned}$$

To evaluate the integral, we change coordinates with $h_1 = ||z||(1 - cos(\theta_1))$. So, $h_1 = 0, h$ implies that $\theta_1 = 0, arccos(1 - h/||z||)$. And, $r(||z||(1 - cos(\theta_1)) = ||z||sin(\theta_1) = \frac{dh_1}{d\theta_1}$. Therefore,

$$\begin{aligned} \int_{h_1=0}^{h}(r(h_1))^{m-2}\,dh_1 &= \int_{\theta_1=0}^{arccos(1-h/||z||)} r(||z||(1-cos(\theta_1))^{m-2}\frac{dh_1}{d\theta_1}\,d\theta_1 \\ &= \int_{\theta_1=0}^{arccos(1-h/||z||)} ||z||^{m-2}sin^{m-2}(\theta_1)||z||sin(\theta_1)\,d\theta_1 \\ &= ||z||^{m-1}\int_{\theta_1=0}^{arccos(1-h/||z||)} sin^{m-1}(\theta_1)\,d\theta_1. \end{aligned}$$

Plugging this into the previous equations for $SA(S_m(z,c))$ and using (A.4), we get

$$\begin{aligned} \frac{SA(S_m(z,c))}{SA(S_m(||z||))} &= \left(\frac{(m-1)\pi^{(m-1)/2}||z||^{m-1}\Gamma((m+2)/2)}{\Gamma((m+1)/2)m||z||^{m-1}\pi^{m/2}}\right)\int_{\theta_1=0}^{arccos(1-h/||z||)} sin^{m-1}(\theta_1)\,d\theta_1 \\ &= \left(\frac{(m-1)\Gamma((m+2)/2)}{\Gamma((m+1)/2)m\sqrt{\pi}}\right)\int_{\theta_1=0}^{arccos(1-h/||z||)} sin^{m-1}(\theta_1)\,d\theta_1. \end{aligned}$$

Since $h = \frac{c^2}{2||z||}$, then, as desired, we get

$$\frac{SA(S_m(z,c))}{SA(S_m(||z||))} = \left(\frac{(m-1)\Gamma((m+2)/2)}{\Gamma((m+1)/2)m\sqrt{\pi}}\right)\int_{\theta_1=0}^{arccos(1-[c/||z||\sqrt{2}]^2)} sin^{m-1}(\theta_1)\,d\theta_1. \tag{A.6}$$

Case $||z||\sqrt{2} < c < ||z||2$: As depicted in Figure A.8, $S_m(z,c)$ contains the entire northern hemisphere of $S_m(||z||)$. Let $S_n(-z,c)$ denote the "south pole" cap defined by h' (and c') in Figure A.8 (clearly $c' \leq ||z||\sqrt{2}$). We have

$$\frac{SA(S_m(z,c))}{SA(S_m(||z||))} = 1 - \frac{SA(S_m(-z,c')}{SA(S_m(||z||))}. \tag{A.7}$$

By replacing "c" with "c'" in (A.5) and (A.6) then plugging the resulting expression into (A.7) we get,

$$\frac{SA(S_m(z,c))}{SA(S_m(||z||))} = \begin{cases} 1 - (1/\pi)arccos(1 - [c'/(||z||\sqrt{2})]^2) & \text{if } m = 2; \\ 1 - \frac{(m-1)\Gamma([m+2]/2)}{m\sqrt{\pi}\Gamma([m+1]/2)}\int_{\theta_1=0}^{arccos(1-[c'/(||z||\sqrt{2})]^2)} sin^{m-1}(\theta_1)\,d\theta_1 & \text{if } m \geq 3. \end{cases} \tag{A.8}$$

From Figure A.8, it can be seen that θ is the top angle on an isosceles triangle with sides of length $||z||$ and base of length c'. So, $sin(\theta/2) = \frac{c'}{2||z||}$. The half-angle formula implies $cos(\theta) = 1 - [c'/(||z||\sqrt{2})]^2$. Similar reasoning shows $cos(\pi - \theta) = 1 - [c/(||z||\sqrt{2})]^2$. Since $0 \leq \theta \leq \pi/2$, then $cos(\pi - \theta) = -cos(\theta)$. Thus, $[c/(||z||\sqrt{2})]^2 - 1 = 1 - [c'/(||z||\sqrt{2})]^2$. Plugging $2 - [\frac{c}{||z||\sqrt{2}}]^2$ in for $[\frac{c'}{||z||\sqrt{2}}]^2$ in (A.8) yields the desired results.

Appendix A.3. Known Input Attack: Computing the Closed-Form Expression for $\rho(x_{\hat{j}}, \epsilon)$

Next we develop recursive procedures for computing (6). This amounts to computing the following two functions: (i) $GR(m) = \Gamma([m+2]/2)/\Gamma([m+1]/2)$ for $m \geq 1$; (ii) $SI(z,m) = \int_{\theta_1=0}^{arccos(z)} sin^{m-1}(\theta_1)\, d\theta_1$ for $1 \geq z \geq 0$ and $m \geq 1$. Indeed, (6) is equivalent to

$$\rho(x_{\hat{j}}, \epsilon) = \begin{cases} 1 & \text{if } n-k=0; \\ 1 & \text{if } \|y_j\|\epsilon \geq \|V'_{n-k}y_j\|2 \text{ and } n-k \geq 1; \\ 0.5 & \text{if } \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k=1; \\ 1-(1/\pi)arccos\left(\left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2 - 1\right) & \text{if } \|V'_{n-k}y_j\|\sqrt{2} < \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k=2; \\ 1-\frac{(n-k-1)GR(n-k)}{(n-k)\sqrt{\pi}} SI\left(\left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2 - 1, n-k\right) & \text{if } \|V'_{n-k}y_j\|\sqrt{2} < \|y_j\|\epsilon < \|V'_{n-k}y_j\|2 \text{ and } n-k \geq 3; \\ (1/\pi)arccos\left(1-\left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2\right) & \text{if } \|y_j\|\epsilon \leq \|V'_{n-k}y_j\|\sqrt{2} \text{ and } n-k=2; \\ \frac{(n-k-1)GR(n-k)}{(n-k)\sqrt{\pi}} SI\left(1-\left[\frac{\|y_j\|\epsilon}{\|V'_{n-k}y_j\|\sqrt{2}}\right]^2, n-k\right) & \text{if } \|y_j\|\epsilon \leq \|V'_{n-k}y_j\|\sqrt{2} \text{ and } n-k \geq 3. \end{cases} \tag{A.9}$$

To compute $GR(m)$ for $m \geq 1$, we use the following facts: $\Gamma(z+1) = z\Gamma(z)$ for $z > 0$, $\Gamma(1/2) = \sqrt{\pi}$, and $\Gamma(1) = 1$. Thus, we get a recursive procedure for computing $GR(m)$.

$$GR(m) = \begin{cases} \frac{\sqrt{\pi}}{2} & \text{if } m=1; \\ \frac{\sqrt{\pi}}{1} & \text{if } m=2; \\ \left(\frac{m}{m-1}\right) GR(m-2) & \text{if } m \geq 3. \end{cases} \tag{A.10}$$

To compute $SI(z,m)$ for $1 \geq z \geq 0$ and $m \geq 1$, we use the following facts. $sin^{m-2}(arccos(z)) = [1-z^2]^{(m-2)/2}$ if $m \geq 3$. And, $SI(z,m) = \left[\int sin^{m-1}(\theta_1)\, d\theta_1\right](arccos(z)) - \left[\int sin^{m-1}(\theta_1)\, d\theta_1\right](0)$. And,

$$\left[\int sin^{m-1}(\theta_1)\, d\theta_1\right](w) = \begin{cases} w & \text{if } m-1=0; \\ -cos(w) & \text{if } m-1=1; \\ \frac{m-2}{m-1}\left[\int sin^{m-3}(\theta_1)\, d\theta_1\right](w) - \frac{sin^{m-2}(w)cos(w)}{m-1} & \text{if } m-1 \geq 2; \end{cases} \tag{A.11}$$

Therefore,

$$SI(z,m) = \begin{cases} arccos(z) & \text{if } m=1; \\ 1-z & \text{if } m=2; \\ \frac{m-2}{m-1} SI(z, m-2) - \frac{z[1-z^2]^{(m-2)/2}}{m-1} & \text{if } m \geq 3; \end{cases} \tag{A.12}$$

Chris Giannella is an artificial intelligence engineer with the MITRE corporation in Annapolis Junction Maryland. His current research interests include machine learning and natural language processing. Prior to that, he held faculty positions at New Mexico State University, Loyola University in Maryland, and Goucher College. Prior to that he was a postdoctoral research associate at the University of Maryland, Baltimore County and completed his Ph.D. in Computer Science at Indiana University, Bloomington, Indiana in 2004.

Kun Liu, Ph.D. is working at LinkedIn as a Staff Software Engineer and Applied Researcher. He is primarily focusing on user intent and interest modeling for personalization and ad targeting. Prior to that, he was a Scientist at Yahoo! Labs, leading several successful research projects such as social targeting and commercial mail monetization. Before joining Yahoo, he was a Postdoctoral Researcher at IBM Almaden Research Center, working on privacy-preserving social-network analysis and text analytics. Dr. Liu received his Ph.D. in Computer Science from University of Maryland Baltimore County. His research interests include behavioral ad targeting, privacy-preserving data mining and social-network analysis. He has co-authored over 25 peer-reviewed research papers and book chapters. He also regularly serves on the program committee of many data mining conferences (*e.g.*, KDD, ICDM, PKDD, PAKDD), and as a reviewer of many scientific journals (*e.g.*, IEEE TKDE, ACM TKDD).

Hillol Kargupta is a Professor of Computer Science at the University of Maryland, Baltimore County. He is also a co-founder of AGNIK, a vehicle performance data analytics company for mobile, distributed, and embedded environments. He received his Ph.D. in Computer Science from University of Illinois at Urbana-Champaign in 1996. His research interests include mobile and distributed data mining. Dr. Kargupta is an IEEE Fellow. He won the IBM Innovation Award in 2008 and a National Science Foundation CAREER award in 2001 for his research on ubiquitous and distributed data mining. He and his team received the 2010 Frost and Sullivan Enabling Technology of the Year Award for the MineFleet vehicle performance data mining product and the IEEE Top-10 Data Mining Case Studies Award. His other awards include the best paper award for the 2003 IEEE International Conference on Data Mining for a paper on privacy-preserving data mining, the 2000 TRW Foundation Award, and the 1997 Los Alamos Award for Outstanding Technical Achievement. His dissertation earned him the 1996 Society for Industrial and Applied Mathematics annual best student paper prize.He has published more than one hundred peer-reviewed articles. His research has been funded by the US National Science Foundation, US Air Force, Department of Homeland Security, NASA and various other organizations. He has co-edited several books. He serve(s/d) as an associate editor of the IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Systems, Man, and Cybernetics, Part B and Statistical Analysis and Data Mining Journal. He is/was the Program Co-Chair of 2009 IEEE International Data Mining Conference, General Chair of 2007 NSF Next Generation Data Mining Symposium, Program Co-Chair of 2005 SIAM Data Mining Conference and Associate General Chair of the 2003 ACM SIGKDD Conference, among others.