Institute of Education Sciences
National Center for Education Statistics

NATIONAL INSTITUTE OF STATISTICAL SCIENCES

WHITE PAPER

SECURE STATISTICAL ANALYSIS OF DISTRIBUTED DATA

TABLE OF CONTENTS

SECURE STATISTICAL ANALYSIS OF DISTRIBUTED DATA

EXECUTIVE SUMMARY

This white paper is intended to lay out for the National Center for Education Statistics (NCES) available algorithms and technology solutions for secure, principled statistical analysis of distributed data, as well as to identify gaps in our current understanding.

For concreteness, the following problem served to frame the key issues. Consider a set of "similar" databases, such as statewide longitudinal data systems (SLDS), containing student-level data, in this instance, owned by multiple state education authorities (SEAs). For some reason – possibilities include law, regulation, policy, scale, distrust among the databases owners, lack of a trusted third party, and simple unwillingness – it is not possible to consolidate the data into a single database. Notwithstanding the restriction, the goal is to perform sound statistical analyses without having a consolidated database. Much of this white paper is devoted to explaining ways in which secure and principled analysis is possible, provided only that the database owners are willing to share anonymously certain summaries of their data, such as means and covariances.

The underlying literature spans both statistics and computer science, but the perspective here is statistical. First, the analyses must be *correct*, about which a computer science and statistics are in full agreement. Second, analyses must be *complete*, which is often not true in the computer science literature. Especially, those objects that statisticians use to quantify and characterize uncertainty must be provided. For a linear regression, for instance, in addition to estimated regression coefficients, the analysis must produce estimated standard errors, key test statistics and significance levels ("*p*-values"), information about model fit, model diagnostics and some information about residuals. Some of these objects can be produced using the same methods used to calculate estimators, but others cannot.

From a starting point of concepts and tools from computer science, the approach to secure statistical analyses is laid out using horizontally partitioned data. Brief discussion of privacy-preserving record linkage, analysis of vertically partitioned data, and complex partitions follows.

The paradigm described here and used to frame the problem suffers from a massive shortcoming in many practical applications: the database owners *must prescribe in advance what analysis is to be performed*. Exploratory data analyses are almost completely precluded, unless they are identified one-by-one in advance. Any adaptive human interaction with the data currently seems impossible.

PREFACE

This white paper prepared by the National Institute of Statistical Sciences (NISS) is intended to lay out for the National Center for Education Statistics (NCES) available algorithms and technology solutions for secure, principled statistical analysis of distributed data, assuming it is not possible to consolidate the data into a single database. Much of the paper is devoted to explaining ways in which secure and principled analysis is possible, if the database owners are willing to share anonymously certain summaries of their data, such as means and covariances.

The paper begins by formulating the problem and introducing concepts and tools from Computer Science. This is followed by a discussion of how secure statistical analyses of horizontally partitioned data can be conducted (with a focus on secure summation), as horizontally partitioned data comprise the framing example for NCES. A brief discussion of privacy-preserving record linkage, analysis of vertically partitioned data, and complex partitions are also included, however such analysis are less relevant to the framing example. This is followed by a discussion of implementation issues, including preprocessing, communication, dishonesty, and opting out. Gaps in current understanding are discussed at the end.

NATIONAL INSTITUTE OF STATISTICAL SCIENCES
WHITE PAPER REPORT

SECURE STATISTICAL ANALYSIS OF DISTRIBUTED DATA

I. INTRODUCTION AND PROBLEM FORMULATION

Consider a global database that is partitioned among a number of owners, such as SEAs, other government agencies or companies. These database owners wish to perform a statistically valid analysis of the *integrated database* – that is, the database that would result from merging the individual databases in a single location – but without ever creating the integrated database.

The protocols described here are designed principally to protect the owners from one another. Each owner can compare the global analysis to the same analysis on its own data, but no owner can attribute any characteristics of the difference to specific other databases. The extent to which the individual data subjects are protected by such protocols is not understood: see §9.1.

The secure statistical analysis process must:

1) Yield the same answer as would have been obtained from performing the analysis on the integrated database.
2) Produce all objects that statisticians or other analysts customarily think of as part of the analysis.
3) Provide no database owner knowledge about the other owners' databases beyond what can be deduced from the results of the "combined" analysis. See also §3.5.

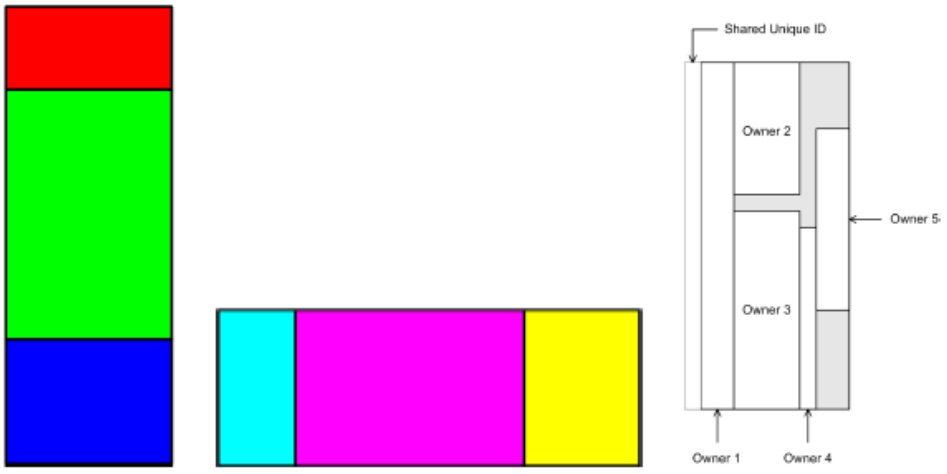


Figure 1: Models for data partitioning. Data subjects are rows, and attributes are columns. *Left*: horizontally partitioned data - data subjects are partitioned among database owners. *Center*: Vertically partitioned data - attributes are partitioned among database owners. *Right*: complex data partition with five database owners.

In addition, implementations must be computationally feasible, scalable and secure from tampering by malicious owners or external parties.

Throughout, a database is a flat file in which rows represent data subjects and columns represent attributes. Mainly, we focus on *horizontally partitioned data*: the data subjects are partitioned among the databases, and owner each has the same attributes for all of its subjects. This model fits the SLDS/SEA setting. Horizontal partitioning is illustrated in the left-hand panel of Figure 1. For *vertically partitioned data*, the attributes rather than the subjects are partitioned among the databases, as in the center panel in Figure 1. More complex partitions are also possible, as in the right-hand panel in Figure 1; see also §7. All partitioning models engender significant metadata and preprocessing issues; see §8.1.

II. CONCEPTS AND TOOLS FROM COMPUTER SCIENCE

In this section, we provide some background from the computer science – specifically, cryptography – literature.

2.1 Semi-Honesty

Parties involved in a distributed computation are termed *semi-honest* if they perform agreed-upon computations correctly *and* they use their true data. If the protocol is iterative, they are permitted to retain the results of intermediate computations. In §8.3, we examine the consequences of violations of the assumption of semi-honesty, and propose one way to mitigate them.

2.2 RSA Encryption

For our purposes, encryption can be thought of as a pair of functions, the public key f and the private key g, such that $g(f(x)) = x$ for each message x. As the names suggest, f is known publicly, so that anyone can encrypt a message x by calculating $f(x)$. However, only the holder of the private key is able to decrypt the message by calculating $f(x)$. In order for the process to be effective, it must be essentially impossible to recover g from knowledge of f.

Most modern encryption methods are based on RSA encryption (Rivest et al., 1978), which is named after its creators. Conceptually, a message is an integer m; procedures exists for converting actual text to messages, but in many of the procedures discussed below, the messages are numerical values, albeit not necessarily integers.

An RSA encryption system consists of

- A *modulus* $n = pq$, where p and q are large prime numbers, in practice, consisting of approximately the same number of digits.
- A *public exponent* e with the property that e and $\varphi(n) = n - (p + q - 1)$ are relatively prime.
- A *private exponent* $d = e^{-1} \bmod \varphi(n)$

The values of n and e are made public; p, q and therefore $\varphi(n)$ are kept secret. Breaking the code requires factoring n, which is impossibly difficult computationally.

Then, anyone can encrypt a message m using the modulus and public key:

$$E(m) \; m^e \qquad \bmod n.$$

However, only the holder of the private exponent can decrypt a message c, via

$$D(c) = c^d \qquad \text{mod } n\text{:}$$

That $D(E(m))\ D = m$ is obvious.

2.3 Homomorphic Encryption

Homomorphic encryption allows algebraic operations to be conducted on unencrypted objects by performing other algebraic computations on encrypted objects. One example is additive homomorphic encryption (Samet and Miri, 2011). Let p and q be large prime numbers, and let $n = pq$, and let λ be the least common multiple of $p - 1$ and $q - 1$. Let g be a random integer in $\{1, \ldots, n^2\}$ whose order is divisible by n. The public key is the pair (n, g), which is used to encrypt a message m as

$$E(m) = g^m \times r^n \qquad \text{mod } n^2,$$

where r is randomly chosen from $\{1, \ldots, n\}$. The private key is (λ, μ), where

$$\mu = \left[L(g^\lambda \mod n^2)\right]^{-1},$$

where $L(u) = (u - 1)/n$. An encrypted message c is decrypted as

$$D(c) = L\left(c^\lambda \mod n^2\right) \times \mu \qquad \text{mod } n.$$

The importance of this technique is the translation of algebraic operations: for messages m_1 and m_2,

$$D\left(E(m_1) \times E(m_2) \mod n^2\right) = m_1 + m_2 \qquad \text{mod } n$$

and

$$D\left(E(m_1)^{m_2} \mod n^2\right) = m_1 \times m_2 \qquad \text{mod } n.$$

2.4 Oblivious Computation

Another key concept is that of obliviousness, which allows queries to be made to a database without the owner's knowing exactly what they are. We illustrate with two relevant examples.

Oblivious Polynomial Evaluation. Consider two parties, one of whom – the sender – holds a polynomial (or other function) Q and the other of whom – the receiver – holds a value v. The goal is for the receiver to learn the value of $Q.(v)$ without full knowledge of Q, and without the sender's learning the value of v. Algorithms for oblivious polynomial evaluation are complex, and are typically based on binary gate representations or homomorphic encryption (Naor and Pinkas, 1999a,b). Rather than present details, we note one key application: determination of duplicates in multiple databases.[1]

Assume that the sender and receiver hold databases that have the same integer-valued primary key.[2] The sender constructs the polynomial

$$Q(x) = \prod_{i=1}^{n} (x - x_i),$$

[1] This is an essential part of the preprocessing for horizontally partitioned data; see §8.1.

[2] In labor force settings, the key could be the social security number; for SLDS, such a key seems unlikely to exist, but see §5 for further discussion.

where x_i is the value of the primary key for the ith record in the sender's database. Let y be the value of the key for a record in the receiver's database. Using oblivious polynomial evaluation, the receiver learns $Q(y) = 0$ without any knowledge of Q, and neither does the sender learn the value of y. If $Q(y) = 0$, the receiver knows that the sender's database contains a record with key y; if $Q(y) \neq 0$, the receiver knows that the sender's database does not contain a record with key y.

The value to SEAs and NCES of knowing the extent of duplication among the different SLDS seems apparent.

Oblivious Transfer. In this case, the sender holds a set of values $v_{1, \ldots,} v_N$ and the receiver holds an integer $I \in \{1, \ldots, N\}$. The goal is for the receiver to learn the value v_I but not any of the values v_j; $j \neq I$, and without the sender learning the value of I. Encryption-based algorithms exist for oblivious transfer (Naor and Pinkas, 1999b). Here is a simple illustration.

1. The sender generates an RSA encryption system with modulus n, public exponent e and private exponent d, and shares the first two of these with the receiver.
2. The sender generates N random integers $x_1, \ldots, x_N$, and sends these to the receiver.
3. The receiver generates a random integer y and sends the "encrypted value" $a = x_I + y^e \ (\text{mod } n)$ to the receiver.
4. The sender calculates the "decrypted value" $b_i = (a - x_i)^d$ for each i. Note that $b_1 = y$, which the receiver but not the sender can recognize, while every other b_j has no meaning.
5. The sender sends $c_i = b_i + v_i = a - x_i + v_i$, *for* $i = 1, \ldots, N$ to the receiver, who can then recover v_I since $v_I = c_I - y$. For $j \neq I$, the receiver cannot disentangle b_j from v_j.

Oblivious transfer can be used in conjunction with oblivious polynomial evaluation to exchange the contents of duplicate records in two databases without revealing any information about non-duplicates.

2.5 Secure Multiparty Computation

Here we give a brief introduction to secure multi-party computation (SMPC). General references are Goldwasser (1997) and Yao (1982). Much of the computer science literature on the theory of SMPC relates to representing function evaluation using binary gates.

The fundamental problem is that of K database owners with values $v_1, \ldots, v_K$ who wish to evaluate

$$f(v_1, \ldots, v_K),$$

where f is a known function, which is typically symmetric in its arguments, subject to three constraints:

Correctness. The correct value $f(v_1, \ldots, v_K)$ is obtained and made known to all owners.

Privacy. Each owner j learns no more about the other owners' values $V_{-j} = \{v_k : k \neq j\}$ than it can deduce from v_j and $f(v_1, \ldots, v_K)$.

Security. No trusted third party is part of the process.

Our model of a trusted third party is an organization, person or computer to which the owners would submit their values v_j, would calculate $f(v_1, \ldots, v_K)$ and would inform the owners of the result. The challenge is that the process must be as effective as if there were a trusted third party, while **Security** forbids this.

Nearly all protocols for SMPC assume that the owners are *semi-honest* in the sense of §3.1.

2.6 Secure Summation

In this paper we focus on one form of SMPC – secure summation: the database owners wish to compute

$$V = f(v_1, \ldots, v_K) = v_1 + \cdots + v_K$$

The secure summation protocol (Benaloh, 1987; Schneier, 1995) is depicted graphically in Figure 2. Assume for simplicity that the v_k are integers.

Initialization. Owner 1 generates (and retains) a very large random integer R, adds R to its value v_1, and sends the sum $R + v_1$ to owner 2.

Iteration. Since R is random, owner 2 learns effectively nothing about v_1 from $R + v_1$. It simply adds its value v_2 to $R + v_1$, sends the result to owner 3, and so on.

Sharing. Finally, owner 1 receives $R + v_1 + \cdots + v_K = R + V$ from owner K, subtracts R, and shares the result V with the other owners.

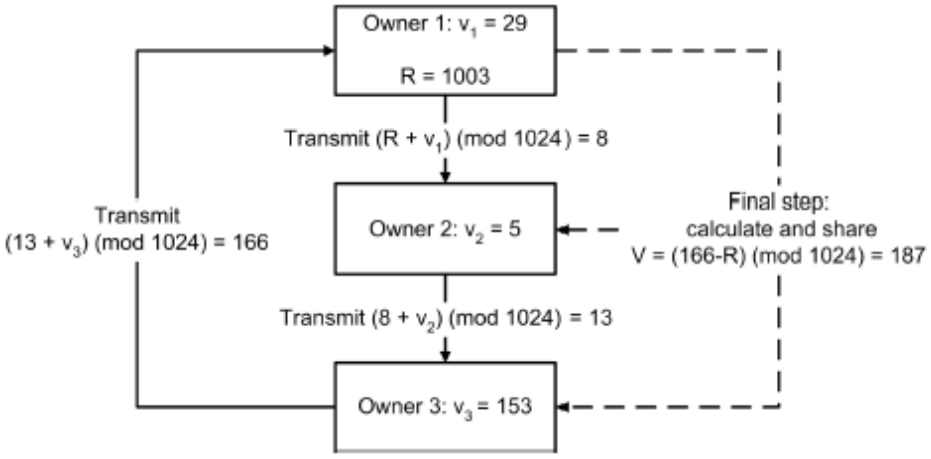


Figure 2: Pictorial representation of secure summation.

Following completion of the protocol, each owner j knows *only* v_j and V, from which it can calculate $V\text{-}_j = \Sigma_{k \neq j}\, {}^{v}k$, but nothing more, at least in the absence of external knowledge. Therefore, it cannot resolve $V\text{-}_j$ into its components v_k; $k \neq j$, nor can it associate these with specific other owners.

Figure 2 depicts an extra layer of protection. Suppose that V is known to lie in the range [0, m), where m is a very large number, say $2^{100} - 1$, known to all the owners. Then R can be chosen randomly from $\{0, \ldots, m - 1\}$ and all computations performed modulo m, which provides added protection to owners with large values of v_j that are early in the process.

While the secure summation protocol is simple, implementation presents challenges. For example, neither another owner nor an outsider should be able to masquerade as an owner, nor should the process be visible to, or corruptible by, outsiders. Also, secure summation is vulnerable to collusion among a subset of the database owners, although there are known ways (Benaloh, 1987) to address this problem.

2.7 Secure Matrix Products

Many statistical procedures involve matrix or vector computations.

The notation in this section matches that of §6, where we discuss secure regression on *vertically* partitioned data, in which the key step is to compute off-diagonal blocks of the full data covariance matrix.

For simplicity, consider agencies A and B, that there are n data subjects common to both databases, and that agency A (B) holds p_A (p_B) attributes. We write the data of agency A as

$$\mathbf{X}^{\mathrm{A}} = [X_1^{\mathrm{A}}\ X_2^{\mathrm{A}}\ \cdots\ X_{p_{\mathrm{A}}}^{\mathrm{A}}];$$

this is an unconventional representation in the sense that the X_i^{A} are *columns* in agency A's data matrix - each belongs to $\mathbb{R}^n$. Similarly, we write agency B's data as

$$\mathbf{X}^{\mathrm{B}} = [X_1^{\mathrm{B}}\ X_2^{\mathrm{B}}\ \cdots\ X_{p_{\mathrm{B}}}^{\mathrm{B}}].$$

Assume that X^{A} and X^{B} are of full rank; if not, each agency removes any linearly dependent columns.

Agency A and agency B wish to compute securely the (p_A x p_B)-dimensional matrix $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$. We first describe a generic protocol for computing $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$, and then show how it can be applied in such a way that the information exchanged between the two agencies is symmetric.

The protocol is as follows:

Step 1: Agency A generates a set of g n-dimensional vectors $\{\mathrm{Z}_1, \mathrm{Z}_2, \ldots, \mathrm{Z}_g\}$ such that

$$Z_i^T X_j^{\mathrm{A}} = 0 \text{ for all } i \text{ and } j, \tag{1}$$

and then forms and sends to agency B the matrix (n x g)-dimensional matrix

$$\mathbf{Z} = [\mathrm{Z}_1\ \mathrm{Z}_2\ \ldots\ \mathrm{Z}_g].$$

A method for generating **Z** is presented in Sanil et al. (2009), which yields Z_i that are orthonormal.

The choice of g is discussed below.

Step 2: Agency B computes

$$\mathbf{W} = (\mathbf{I} - \mathbf{Z}\mathbf{Z}^{\mathrm{T}})\ \mathbf{X}^{\mathrm{B}}, \tag{2}$$

where **I** is an (n x n)-dimensional identity matrix, and sends **W** to agency A.

Step 3: Agency A calculates

$$(\mathbf{X}^{\mathrm{A}})^T\mathbf{W} = (\mathbf{X}^{\mathrm{A}})^T(\mathbf{I} - \mathbf{Z}\mathbf{Z}^T)\mathbf{X}^{\mathrm{B}} = (\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}},$$

where the second equality holds since $(\mathbf{X}^{\mathrm{A}})^T\mathbf{Z} = 0$.

In view of **Step 2**, agency A could instead send the $n \times n$-dimensional matrix $\mathbf{Z}\mathbf{Z}^T$ to agency B. On the face of it, this seems safer, but in fact the loss of protection does not change.

Loss of Protection. The absolute and relative protection that the protocol provides to agencies A and B depends on the parameter g in **Step 1**. First, consider two extreme cases:

- $g = 0$: In this case, in (2),$\mathbf{W} = \mathbf{X}^{\mathrm{B}}$, so agency A has learned agency B's data exactly.
- $g = n - p$: In this case, agency B knows exactly the orthogonal complement of $\mathbf{X}^{\mathrm{A}}$ in $\mathbb{R}^n$. While this does not specify $\mathbf{X}^{\mathrm{A}}$ exactly, agency B does know the span of $\mathbf{X}^{\mathrm{A}}$.

Neither of these makes sense in general. Note also that the extreme cases are not precisely symmetric.

A principled method for choosing g is to consider the *loss of protection* incurred by the agencies, which we denote by LP(A)/ for agency α. We measure loss of protection to one agency by the number of (linearly independent) constraints the other agency has on its data. Thus, for agency A,

$$\mathrm{LP(A)} = p_{\mathrm{A}}\, p_{\mathrm{B}} + p_{\mathrm{A}} g. \tag{3}$$

The first term in (3) represents B's knowledge of the $p_{\mathrm{A}} \cdot_{\mathrm{B}}$ entries of $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$ at the end of the process. The second term reflects that B knows both $\mathbf{Z}$ and that $(\mathbf{X}^{\mathrm{A}})^T\mathbf{Z} = 0$ - that is, (1), which contains $p_{\mathrm{A}} \times g$ constraints. One can also view LP(A) relative to the total "degrees of freedom" in $\mathbf{X}^{\mathrm{A}}$, which is $n \times p_{\mathrm{A}}$.

Similarly,

$$\mathrm{LP(B)} = p_{\mathrm{A}}\, p_{\mathrm{B}} + p_{\mathrm{B}}(n - g). \tag{4}$$

The first term is the same as in (3) and the second term reflects that A knows that

$$\mathrm{rank}(\mathbf{W}) = n - g.$$

Indeed, this is why agency A cannot invert $\mathbf{W}$ to obtain $\mathbf{X}^{\mathrm{B}}$.

The total loss of protection, as a function of g, is then

$$\mathrm{LP}(g) = \mathrm{LP(A)} + \mathrm{LP(B)} = 2p_{\mathrm{A}}p_{\mathrm{B}} + np_{\mathrm{B}} + (p_{\mathrm{A}} - p_{\mathrm{B}})g. \tag{5}$$

Note that when $p_{\mathrm{A}} = p_{\mathrm{B}}$, $\mathrm{LP}(g) = 2p_{\mathrm{A}}p_{\mathrm{B}} + np_{\mathrm{B}}$, no matter what the value of g. Harking back to the two extreme cases, when $p_{\mathrm{A}} = p_{\mathrm{B}}$, the total loss of protection is constant, and g affects only how that loss is distributed between agency A and agency B.

In general, it seems desirable that the loss of protection should be shared equally by the two agencies. To measure this, we introduce the *inequity*

$$\mathrm{I}(g) = |\mathrm{LP(A)} - \mathrm{LP(B)}| = |(p_{\mathrm{A}} + p_{\mathrm{B}})g - np_{\mathrm{A}}|. \tag{6}$$

Setting I(g) to its minimal value of zero yields the optimal choice of g:

$$g^* = \frac{p_A}{p_A + p_B} n. \tag{7}$$

The value of g^* in (7) has a natural interpretation: agencies A and B together possess $p_A + p_B$ attributes, so $p_A/(p_A + p_B)$ is agency A's share of those attributes. When A has a larger share of attributes, it must surrender more information to B than *vice versa*.

III. SECURE STATISTICAL ANALYSES OF HORIZONTALLY PARTITIONED DATA

Because horizontally partitioned data comprise the framing example for NCES, this section contains some detail regarding how techniques from §3 can be applied in this setting. The predominant technique is secure summation (§3.6).

3.1 Generalities

Most of what follows is based on one principle: for horizontally partitioned data, any analysis whose sufficient statistics are additive across the databases can be performed securely using secure summation. "Additive" is of course, a broad term, since, for instance, sufficient statistics can be combined multiplicatively by combining their logarithms additively, as in §4.6.

3.2 Descriptive Statistics

Most descriptive statistics for numerical attributes are moments, such as the mean and variance. Moments are trivially based on additive sufficient statistics. As an illustration, suppose that the owners have income data and wish to compute the global average income. Let n_j be the number of records in owner j's database and I_j be the sum of their incomes. The quantity to be computed is

$$\bar{I} = \frac{\sum_j I_j}{\sum_j n_j}, \tag{8}$$

whose numerator can be computed using secure summation on the I_j's, and whose denominator can be computed using secure summation on the n_j's. Each owner can then calculate $\bar{I}$.

Generalized moments, such as characteristic functions, Laplace transforms and even kernel density estimates, can be calculated securely in precisely the same manner. Other descriptive statistics, such as maxima, minima, medians and quantiles, are straightforward conceptually,[3] but more challenging to implement efficiently.

Nearly all summary statistics for categorical data are, or are derived from, contingency tables, which are treated in §4.4.

3.3 Secure Data Integration

The formulation in §2 prohibits creation of the integrated database as a means of performing the analysis. However, there may be situations – see §4.4 for one – in which the owners are willing to

[3] Using bisection methods, for instance.

create and share the global database, provided that the *sources* of data elements are protected. For example, owners who are retailers may be willing to share information about their customers, including who the customers are, provided that they do not reveal who is whose customer.

We sketch a protocol for secure data integration (SDI). Conceptually it is straightforward, provided that the star topology and encryption mechanisms of the Secure Computation System described in §8.5 are employed: the owners incrementally contribute data in a random round-robin order known to a central server but not to them. Figure 7 illustrates the star topology of the system. The circulating database is assumed to be encrypted using a key known to the clients but not the server, and encryption/decryption steps are omitted from the description.

Initial round. Each owner receives from the server a circulating database. It records all elements in it, adds to the database a random number of its records – the need for this randomization and issues associated with it are discussed below – and sends the result back to the server.

Intermediate rounds. Each successive time an owner receives the circulating database, it first removes elements it put on the preceding round. It can recognize these, and it is safe to remove them because they have already been recorded by all the other owners. It then records all other data in the database, which have been added by other owners. Finally, it adds a random subset of its remaining data and returns the database to the server.

Penultimate round. Each owner removes its own previous data and records other owners' previous data as for **Intermediate Rounds**, but now puts in *all its remaining data*. Which round is the penultimate round may be owner-dependent.

Final round. Following its penultimate round, each owner removes the data it inserted then, and sends the database back to the server. The last owner to do this will be left with an empty database, and the process will be complete. Until that happens, each owner will continue to receive the database, will record data added by other owners, and send the database on.

Note that because of the round-robin system, between any two times an owner receives the database *all other owners* (who have not exhausted their own data) will have contributed to it, so there is no means of limiting which other owners may have contributed the new records. The "remove what was put in on the previous round" achieves computational feasibility by controlling the size of the database: the size of circulating database is approximately constant rather than growing linearly with the number of rounds. With n the size of the global database, this reduces the communication overhead from $O(n^2)$ to $O(n)$.

We may evaluate the effectiveness of the protocol using Bayesian methods. Assume that the database owners know the sizes n_j of each other's databases. (If they were state agencies, this would be plausible, since the sizes would often be public information.) Owner j's (naive) prior distribution on the source of any record not from its own database is

$$P\{\text{Source} = \text{owner } k\} = \frac{n_k}{\sum_{\ell \neq j} n_\ell}. \tag{9}$$

The extent to which observing the protocol enables an owner to improve its estimates of records' sources – posterior distributions given what is observed about the protocol – over the priors in (9) measures the lack of protection of sources.

In early rounds, randomizations that add a constant fraction of each owner's records are revealing – the larger the pool received by an owner, the more likely it is to contain records from large databases. At the other extreme, if each owner were to put in the same (expected) number of records on each round, then in late rounds only owners with large databases would be contributing. In empirical experiments, moderate numbers of rounds and multinomial distributions appear to address both considerations.

This protocol described above violates the strict interpretation of condition **Security** in §3, because the server is trusted by the owners to know the order in which they contribute to the data pool. The encryption described in §8.5 prevents the server from learning any data values. Moreover, no owner can know even its place (first, second, . . ., last) in the order. Without this protection, the second owner in the process would know that all records it receives in the initial round came from a single database, and might use relationships deduced from those records to identify their owner.

3.4 Contingency Tables

The algorithm for secure data integration described in §4.3 has an important application to securely constructing contingency tables containing counts or sums.

Let $\mathcal{D}$ be a database containing categorical attributes $A_1, \ldots, A_J$. The associated contingency table is the J-dimensional array T defined by

$$T(a_I, \ldots, a_J) = \#\{r \in \mathcal{D} : r_1 = a_1, \ldots, r_J = a_J\}, \tag{10}$$

where each a_i is a possible value of the categorical attribute A_i (For example, if A_1 corresponds to gender, then possible values of a_1 are "female" and "male."), $\#(\cdot)$ denotes "cardinality of $\cdot$" and r_i is the ith attribute of record r. The J-tuple $(a_1, \ldots, a_J)$ is called the cell coordinates. More generally, contingency tables may contain sums of numerical variables rather than counts; the procedure described below works in either case.

An array is not a feasible data structure for tables with large numbers of cells. Large tables are invariably sparse, however, with relatively few cells having non-zero counts. For instance, the table associated with the American Community Survey (ACS) has more than 10^{30} cells, but at most approximately 10^8 (the number of households in the U.S.) of these are non-zero. The sparse representation of a table is the data structure of (cell coordinate, cell count) pairs

$$\left\{(a_1, \ldots, a_J, T(a_1, \ldots, a_J)) : T(a_1, \ldots, a_J) \neq 0\right\}.$$

Algorithms that use this sparse representation data structure have been developed that support virtually all important table operations (Moore and Lee, 1998).

Consider now the problem of securely building a contingency table from databases $\mathcal{D}_1, \ldots, \mathcal{D}_K$ containing the same categorical attributes for disjoint sets of data subjects. Given the tools described in §4.3 and 3.6, this process is straightforward. The steps:

1. **List of Non-Zero Cells**: Use secure data integration to build the list $\mathcal{L}$ of cells with non-zero counts. The "databases" being integrated in this case are the owners' individual lists of cells with non-zero counts. The protocol in §4.3 allows each owner not to reveal in which cells it has data.
2. **Non-Zero Cell Counts**: For each cell in $\mathcal{L}$, use secure summation to determine the associated count or sum.

3.5 Regression and Similar Models

Assume that the data consist of $p + 1$ numerical attributes of each data subject, so that owner j's data on its n_j subjects consist of p predictors X^j and a response y^j. The owners wish to fit the linear model

$$y = X\beta + \epsilon, \tag{11}$$

to the "global" data

$$X = \begin{bmatrix} X^1 \\ \vdots \\ X^K \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} y^1 \\ \vdots \\ y^K \end{bmatrix}.$$

We embed the constant term of the regression in the first predictor by putting $X_1^j \equiv 1$ for all j. Typically, the predictors and response would be centered at mean values, or even standardized. Because the means and standard deviations required for centering or standardization are the global ones, a preliminary round of secure computation is necessary to compute them.

We assume for simplicity that $\mathrm{Cov}(\varepsilon) = \sigma^2 I$, in which case the least squares estimator for β is

$$\hat{\beta} = (X^T X)^{-1} X^T y. \tag{12}$$

The crucial point is that the global $(p + 1) \times (p + 1)$ matrix

$$[X\ y]^T [X\ y] = \begin{bmatrix} X^T X & X^T y \\ y^T X & y^T y \end{bmatrix}$$

from which $\hat{\beta}$ can be calculated using (12), is additive over the owners:

$$[X\ y]^T [X\ y] = \sum_{k=1}^{K} [X^k\ y^k]^T [X^k\ y^k]. \tag{13}$$

Therefore, $[X\,y]^T\,[X\,y]$ can be computed entry wise using secure summation. In the Secure Computation System described in §8.5, a single secure summation is performed on $(p + 1) \times (p + 1)$ matrices whose entries are initialized with independently generated random numbers.

We illustrate with a data set of 1,318 chemical compounds, in which the response is water solubility and the 91 predictors are a constant and 90 chemical features of the compounds (Karr et al., 2005a).

Four pharmaceutical companies were created, whose databases contain 499, 572, 16 (Not enough even to do the regression!) and 231 compounds, respectively. This example mimics real-world heterogeneity, where each company's database contains compounds with features that are absent from all compounds in the other companies' databases. This sharpens the incentive for each company to participate, because it can learn about features for which it has no data. Of course, company 3 has greatest incentive to participate, since it cannot even do the regression on its own.

There is no need to illustrate that the secure regression protocol produces the correct answer: it is guaranteed to do so if the database owners are semi-honest. It is more pertinent to ask what the database owners gain and lose from the global analysis. Figure 3 compares that analysis with single-company analyses. The first three panels are scatterplots of the 91 regression coefficients for companies 1, 2 and 4 (y-axis) against the coefficients for the global (four-company) regression (x-axis). Coefficients with y-values of zero correspond to features missing from each company's database. Not surprisingly, the match between each of company 1, 2 and 4's coefficients and the global coefficients depends on the size of its database: the larger the database, the better the match.

Calculation of $\hat{\beta}$ is only part of a valid, useful regression. A variety of other objects can be calculated from $[X\,y]^T\,[X\,y]$, or using secure summation directly. These include the coefficient of determination R^2, the least squares estimate S^2 of the error variance o^2 and the "hat" matrix $H = X^T X)^{-1} X^T$, which can be used to identify outliers (Karr et al., 2005b, 2006a). It is also possible to use the secure data integration algorithm of §4.3, together with methods for constructing privacy-preserving synthetic residuals in ordinary regressions (Reiter, 2003), to create secure synthetic residuals (Karr et al., 2006a).

Other analyses that are sufficiently similar to regression can be performed in the same way. For example, the constant variance assumption preceding can be relaxed. Analyses using adaptive regression splines are treated in Ghosh et al. (2007).

3.6 Maximum Likelihood Estimation

In this section, we describe two approaches to maximum likelihood estimation. The first – for exponential families – is a straightforward application of the principle of additive sufficient statistics in §4.1. The second, which is numerical likelihood function maximization using the Newton-Raphson algorithm, which also employs additive sufficient statistics, but is quite a bit more complicated, as well as illuminates the limitations of a secure summation-centric approach.

3.6.1 Exponential Families

Suppose now that the owners' databases partition a global database $\{x_i\}$ modeled as independent samples from an unknown density $f(\theta,\cdot)$ belonging to an exponential family:

$$\log f(\theta, x) = \sum_{\ell=1}^{L} c_\ell(x) d_\ell(\theta). \qquad (14)$$

Then under the independence assumption, the global log-likelihood function is

$$\log L(\theta, x) = \sum_{\ell=1}^{L} d_\ell(\theta) \left[\sum_{k=1}^{K} \sum_{x_i \in \mathcal{D}_k} c_\ell(x_i) \right], \tag{15}$$

where $\mathcal{D}_k$ is the database of owner k.

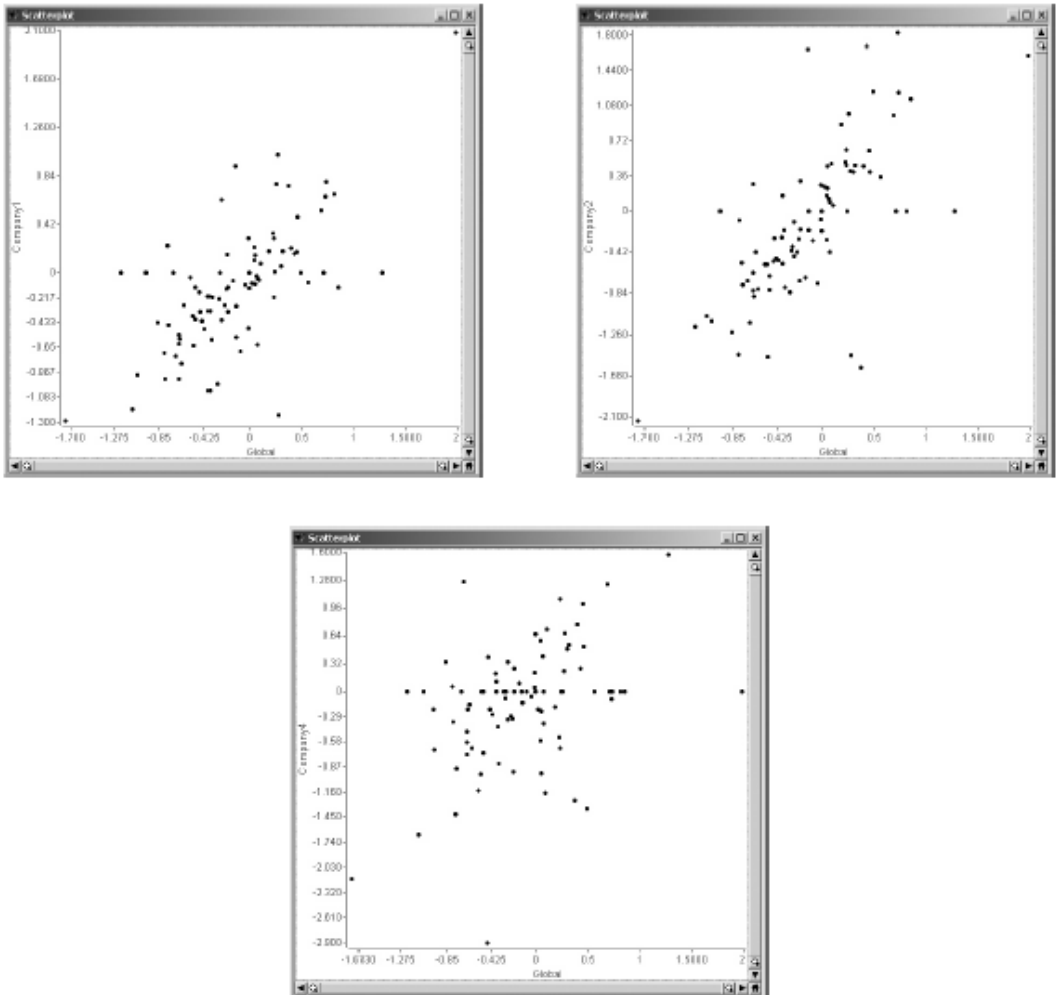


Figure 3: Scatterplots for the example discussed in §4.5. In all plots, the regression coefficients for the four-company regression appear on the x-axis. *Upper left:* y-axis contains regression coefficients for company 1 alone. *Upper right:* y-axis contains regression coefficients for company 2 alone. *Bottom:* y-axis contains regression coefficients for company 4 alone.

Assuming that the owners have agreed in advance on the model (14), they can use secure summation to compute each of the L terms within the brackets in (15), and then each can maximize the likelihood function by whatever means it wishes. No owner's success at maximization is impaired by another owner's lack of success.

3.6.2 Numerical MLE via Newton–Raphson

When analytical solution is not possible, iterative algorithms are used to compute maximum likelihood (ML) estimators (Karr and Lin, 2010). One of the most popular of these is the Newton-Raphson algorithm for root evaluation, which finds local maxima of the log-likelihood function $l(\theta|\mathbf{X})$ by locating a zero of its derivative. More specifically, assume that the first and second derivatives of the likelihood function with respect to θ exist. Then given an estimator $\hat{\theta}^{(s-1)} - \theta$ at step s – 1 , the estimator at step s is

$$\hat{\theta}^{(s)} = \hat{\theta}^{(s-1)} - [D^2\ell(\mathbf{X}|\hat{\theta}^{(s-1)})]^{-1}\nabla\ell(\mathbf{X}|\hat{\theta}^{(s-1)}), \tag{16}$$

where $D^2\ell(\mathbf{X}|\hat{\theta}^{(s-1)})$ is the Hessian matrix of the log-likelihood function evaluated at $\hat{\theta}^{(s-1)}$ and $\nabla\ell(\mathbf{X}|\hat{\theta}^{(s-1)})$ is the gradient.

Let $\theta = (\theta_1, \ldots, \theta_q)$. Then for each $1 \le j \le q$,

$$\nabla_\theta\ell(\mathbf{X}|\hat{\theta}^{(s-1)})(j) = \left(\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\frac{\frac{\partial f(X_i;\theta)}{\partial\theta_j}}{f(X_i;\theta)}\right)_{\hat{\theta}^{(s-1)}}, \tag{17}$$

and similarly, for each h and j,

$$D^2\ell(\mathbf{X}|\hat{\theta}^{(s-1)})(h,j) = \sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\left(\frac{\frac{\partial^2 f(X_i;\theta)}{\partial\theta_h\partial\theta_j}}{f(X_i;\theta)} - \frac{\frac{\partial f(X_i;\theta)}{\partial\theta_h}\frac{\partial f(X_i;\theta)}{\partial\theta_j}}{f^2(X_i;\theta)}\right)_{\hat{\theta}^{(s-1)}}, \tag{18}$$

Each of these is computable using secure summation, and so is the Newton-Raphson step, using (16).

To understand the security implications of this approach, let

$$L_k(j) = \left(\sum_{i\in\mathcal{I}_k}\frac{\frac{\partial f(X_i;\theta)}{\partial\theta_1}}{f(X_i;\theta)}\right)_{\hat{\theta}^{(s-1)}}$$

and

$$H_k(h,j) = \sum_{i\in\mathcal{I}_k}\left(\frac{\frac{\partial^2 f(X_i;\theta)}{\partial\theta_h\partial\theta_j}}{f(X_i;\theta)} - \frac{\frac{\partial f(X_i;\theta)}{\partial\theta_h}\frac{\partial f(X_i;\theta)}{\partial\theta_j}}{f^2(X_i;\theta)}\right)_{\hat{\theta}^{(s-1)}}.$$

Then, the approach uses secure summation to calculate $\sum_{k=1}^{K} L_k$ and $\sum_{k=1}^{K} H_k$, but all that is needed to calculate the Newton-Raphson update is $[\sum_{k=1}^{K} H_k]^{-1}\sum_{k=1}^{K} L_k$. Thus, more information is shared than is necessary.[4]

Thus, more information is shared than is necessary.[4]

A reasonable remedy uses the fact that matrix inversion amounts to solution of a system of linear equations. For simplicity, assume that $K = 2$.[5] So, to compute $Z = (H_1 + H_2)^{-1}(L_1 + L_2)$ we need to solve the linear system $(H_1 + H_2)Z = (L_1 + L_2)$.

[4] This same issue is noted in Karr et al. (2007) for regression: secure summation is used to compute $X^T X$ and $X^T y$, but all that is really needed is $(X^T X)^{-1} X^T y$. Moreover, a dishonest agency can exploit this by lying about its data.

[5] In fact, this is the most difficult case, because secure summation does not protect information when there are only two agencies.

Our protocol is as follows: agency A generates a $q \times q$ matrix M_1, which is of rank $[q/2]$ and sends M_1 to agency B. Agency B then computes M_1H_2 and M_1L_2, sends them back to agency A, which can then produce the linear system

$$M_1(H_1 + H_2)Z = M_1(L_1 + L_2).$$

Symmetrically, agency B generates and sends to agency A a matrix M_2 that is of rank $[q/2]$, and can then produce the linear system

$$M_2(H_1 + H_2)Z = M_2(L_1 + L_2).$$

Direct sharing of either$M_1(H_1 + H_2)$ of $M_1(L_1 + L_2)$ would divulge information. However, if T_1 and T_2 are full rank matrices generated by agencies A and B, respectively, then the systems

$$T_1M_1(H_1 + H_2)Z = T_1M_1(L_1 + L_2)$$

and

$$T_2M_1(H_1 + H_2)\,X = T_2M_1(L_1 + L_2)$$

can be combined into a system solvable for Z.

The degree of protection afforded by this protocol depends on the value of q: the larger the better.

There are other issues associated with the iterative nature of the Newton-Raphson algorithm. In particular, numerical inaccuracies associated with differentiation of the log-likelihood function, from matrix multiplication or from matrix inversion that arise at *any* agency affect computations at *all* agencies. These problems can be detected, if not circumvented, by having each agency calculate its proposed value for $\hat{\theta}^{s)}$, and using secure summation and a secure Boolean operation to terminate the process if any agency's value differs too drastically from the mean.

3.7 Linear Mixed Models

We focus on the model most relevant to distributed data, with database-dependent random effects – in the context of SLDS, state-level random effects. The model for horizontally partitioned data is then

$$Y_i = X_i\beta + U_{D(i)} + \varepsilon_i, \tag{19}$$

where $D(i)$ is the database to which record i belongs. We assume, for simplicity of exposition, that the U_j are uncorrelated random variables with mean 0 and variances τ_j^2, that the ε_i are uncorrelated random variables with mean 0 and common variance o^2, and that $\mathbf{U} = (U_1, \ldots, U_K)$ and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_{n1} + \cdots + n_K)$ are uncorrelated. (Recall that K is the number of databases and n_ℓ is the number of elements in the ℓ^{th} database.) The design matrix Z for the random effects then has the following structure:

$$Z(i, j) = \begin{cases} 1 & \text{if } j = D(i) \\ 0 & \text{otherwise.} \end{cases}$$

Let G = diag$(\tau_1^2, \ldots, \tau_N^2)$be the covariance matrix of the random effects and let $R = o^2 I$ be the covariance matrix of the ε_i. Then, estimation proceeds as follows (SAS Institute, Inc., 2014):

1. Given distributional assumptions, maximum likelihood estimators $\widehat{\tau_1^2}, \ldots, \widehat{\tau_N^2}$ and $\widehat{\sigma^2}$ are calculated using a Newton-Raphson procedure, as in §4.6.2. The applicable likelihood function for normally distributed random effects and errors is given in SAS Institute, Inc. (2014); see also McCulloch et al. (2008).
2. Let $V = ZGZ^T + R$ be the overall covariance matrix of the data, which has the following structure:

$$V(i,k) = \begin{cases} \tau^2_{D(i)} + 1(i=k)\sigma^2 & \text{if } D(i) = D(k) \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

In particular, all "off-diagonal" blocks of V are zero, and both $\hat{V}$ and, therefore $\hat{V}^{-1}$ also have this structure.

Then, in the obvious notation, the estimators of β are

$$\hat{\beta} = (X^T \hat{V}^{-1} X)^- X^T \hat{V}^{-1} Y, \tag{21}$$

where C^- is the generalized inverse of the matrix C.

3. The predicted random effects are then

$$\hat{U} = \hat{G} Z^T \hat{V}^{-1} \left(Y - X\hat{\beta} \right). \tag{22}$$

To perform this process securely is feasible, but not entirely straightforward. The Newton-Raphson calculation leading to $\hat{V}$ would be performed in the manner of §4.6.2. Given $\hat{V}$, for our model, $(X^T \hat{V}^{-1} X)$ and $\hat{V}^{-1} Y$ can be calculated using secure summation. (This is not true for model complex models in which the random effects are correlated.) The computation in (21) can then be performed independently by each database owner. Those with poor algorithms for calculating generalized inverses will arrive at inaccurate values for $\hat{\beta}$. The final step, in equation (22), requires calculation of residuals, which in general is problematic (Karr et al., 2007). However, in this case, the block structure of the matrices implies that for each j, $\hat{U}_j$ can be calculated entirely by the owner of database j. Consequently, all that is required is agreement by the database owners to share the predicted random effects with one another. Such sharing cannot, of course, be effected by means of secure summation. (Alternatively, it could be argued that $\hat{U}_j$ is of interest only to the owner of database j, and that no sharing is necessary. For some "administrative" purposes this argument may make sense, but it does not make sense for research purposes. In the context of SLDS, many researchers would be very interested in comparing the state-level random effects.)

3.8 Structural Equation Models and Causal Inference

Structural equation models (Bollen, 1989; Pearl, 2009), or SEMs, can, with some oversimplification, be construed as multivariate regression models in which some responses (the left-hand sides of equations) can appear as predictors of other responses. Such variables are termed *endogenous*; variables appearing only as predictors are *exogenous*. Widespread of SEMs reflects their ability to represent (formal or informal) thinking about causal relationships, especially in the social sciences (Heckman, 2005), as well as to accommodate latent variables. The widely cited Rubin causal model is a special case of SEMs (Pearl, 2009).

Estimation for SEMs mimics that for regression models, although there are substantial issues regarding identifiability. Purely for expository linkage, consider a single equation of the form

$$Y = X\delta + \zeta,$$

with n cases and p predictors (some of which may be responses in other equations), and let Z be an $n \times p$ matrix of *instrumental* variables that uncorrelated with the errors ζ. There is a strong assumption that the number of instrumental variables is equal to the number of predictors. Then, a consistent estimator of δ is

$$\hat{\delta} = (Z^T X)^{-1} Z^T Y. \tag{23}$$

In fact, it is possible to estimate δ using two-stage ordinary least squares:

$$\hat{X} = X(ZZ^T)^{-1} Z^T X \tag{24}$$

followed by

$$\hat{X} = (\hat{X}\hat{X}^T)^{-1} \hat{X}^T Y. \tag{25}$$

Therefore, the protocol in §4.5 is applicable.

3.9 Privacy-Preserving Data Mining

The computer science literature contains numerous references to "privacy-preserving data mining (PPDM);" an accessible source with broad coverage is Samet and Miri (2011). Sometimes the concept is construed as synonymous with "secure statistical analysis" in the sense that we use the term, but often the analyses are substantially incomplete. For instance, a fitted model may be calculated, but not measures of fit, model diagnostics or information about residuals. For completeness, we provide some comments and pointers to the literature.

Classification. Many classifiers, such as decision trees (see, e.g., Quinlan (1986) and Vaidya et al. (2006)) operate by dichotomously and recursively splitting a response on the basis of the values of one or more predictor variables. Splitting criteria include entropy and two-sample T -tests, and are often additive across databases, in which case secure summation is applicable. Further details are available in Lindell and Pinkas (2000) and Xiao et al. (2005). However, to our knowledge none of the extant algorithms provides output at the level of detail in, for instance, SAS® or JMP®.

Clustering. Perhaps the most commonly used clustering method for numerical data, *k*-means clustering, is compatible with secure summation in the context of horizontally partitioned data. Cluster centroids are moments (see §4.2) and assignment of points to clusters can occur separately and privately for each database.

Association Rules. Association rules are based on additive sufficient statistics known as *support* and *confidence*, and can be implemented using only secure summation. See Vaidya and Clifton (2002).

Neural Networks. Existing protocols, such as those in Samet and Miri (2008) and Samet and Miri (2011), are almost all inadequate from a statistical perspective, because they fail to provide essential outputs.

IV. PRIVACY-PRESERVING RECORD LINKAGE

There is a growing literature on privacy-preserving record linkage (PPRL), which can be accessed from sources that include Christen (2012), Clifton et al. (2003), Hall and Fienberg (2013) and Schadow et al. (2002). As articulated in Christen (2013), the motivating application is secure identification of duplicates in databases in the absence of a common primary key.[6] The techniques generalize, in some ways, the pioneering work in Fellegi and Sunter (1969). They are based on binary or distance-quantified matches of certain attributes.

V. ANALYSIS OF VERTICALLY PARTITIONED DATA

Because vertically partitioned data are less relevant than horizontally partitioned data to the framing example of SLDS, our treatment in this section, with the exception of linear regression – which exemplifies one approach – is brief. There may be other contexts relevant to NCES which vertically partitioned data do arise: examples include student and teacher data in different databases and student and workforce data in different databases.

5.1 Secure Computation of Covariance Matrices

Secure analyses of vertically partitioned data (Sanil et al., 2009) depends on the matrix multiplication protocol described in §3.7. Alternative approaches appear in Du et al. (2004) and Sanil et al. (2004)

We assume the database owners are willing to share sample means and covariances of their individual databases. The main issue, obviously, is computation of the full data covariance matrix which requires that the owners surrender some dimensions of their data to each other. Exactly what we mean by "surrender some dimensions" is explained below.

It is important to note that the loss of protection discussed in this paper applies only to the database owners vis-à-vis one another, and only in the aggregated sense of the span of their databases. The same is true in general for PPDM. The measure LP defined by (5) below does not address threats to the confidentiality of data records or the privacy of individual data subjects, the traditional focus of statistical disclosure limitation (SDL), arising from either computation of the full data covariance matrix or use of it to perform analyses such as regressions. One exception to this, whereby an agency could learn exact data values held by another agency for one subject, is discussed below.

From shared means and the securely computed full data covariance matrix, the owners can perform richer sets of analyses than estimating regression coefficients. These analyses include inference for the coefficients, model diagnostics and model selection. We note that the approach of Du et al. (2004) can be modified to share sample covariance matrices, although the protocol presented in §3.7 holds advantages over it. The approach of Sanil et al. (2004) cannot be so modified.

[6] When there is a common primary key, techniques based on oblivious polynomial evaluation can be employed; see §3.4.

We label the data owners as Agency A, agency B, . . . , agency Ω, even though they might be private companies or other data holders. The "global" database **X**, illustrated for three agencies in Figure 4, is partitioned vertically among the agencies:

$$\mathbf{X} = [\mathbf{X}^{\mathrm{A}}\ \mathbf{X}^{\mathrm{B}}\ \cdots \mathbf{X}^{\Omega}]. \tag{26}$$

Let n be the number of records in the global database, suppose that agency X has p_{X} attributes, and let $p = p_{\mathrm{A}} + \cdots + p_{\Omega}$.

A number of issues, some subtle and possibly difficult, underlie the process. First, the agencies must have a privacy–protecting method of determining which data subjects are common to all of their databases. The most straightforward way to do this is by means of a common primary key, such as social security numbers, possibly in conjunction with oblivious polynomial evaluation (§3.4). In other instances, however, it might be necessary to use record linkage methods; see §5. We further assume that the agencies have aligned their common data subjects in the same order. Finally, we assume that the sets of attributes in the X^a are disjoint; if not, the agencies coordinate so that each common attribute is included in only one agency's data.

The statistical analyses discussed below are all based on the (p x p)-dimensional full data covariance matrix $\mathbf{X}^T\mathbf{X}$, which is shown pictorially, also for three agencies, in Figure 5. The goal of the agencies is to compute and share $\mathbf{X}^T\mathbf{X}$ in a way that minimizes the information each reveals to the others about its data values. As that figure shows, $\mathbf{X}^T\mathbf{X}$ consists of:

On-diagonal blocks of the form $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$. Each of these must be computed by one agency and shared with the others.

Off-diagonal blocks of the form $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$, where A ≠ B. Each of these must be computed by two agencies, and the result shared with the others.

§3.7 describes a protocol for computation of the $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$.

The optimal value g^* in (6) applies only to computation of $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$. This is not the only equity issue associated with computation of $\mathbf{X}^T\mathbf{X}$.

When the agencies have different numbers of attributes, perhaps the most glaring inequity is associated with the on-diagonal blocks of $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{A}}$ of $\mathbf{X}^T\mathbf{X}$. The dimensions of $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{A}}$ are p_{A} x p_{A}, and is as shown in Figure 5, these blocks may differ substantially in size. Without even considering off-diagonal blocks, each agency A is surrendering p_{A}^2 constraints on its data X^{A}.

Also, the loss of privacy LP(A) in (3) is what agency A loses to agency B in the course of computing $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$. None of this is fully given up to any other agencies when $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$ is shared, because no agency other than B knows $\mathbf{X}^{\mathrm{B}}$. Other agencies, however, know different constraints on $\mathbf{X}^{\mathrm{B}}$, and second-order computations may be possible.

But, agency A must engage in calculation of $(\mathbf{X}^{\mathrm{A}})^T\mathbf{X}^{\mathrm{B}}$ with *every other* agency B, which does increase loss of privacy. However, under the assumption that agencies do not collude, the loss does not increase. Even if agencies do collude, agency A can mitigate the effects by making the **Z** matrices it sends to other agencies subsets of one matrix. This means that the agency B for which g^* in (7) is largest learns the most about $\mathbf{X}^{\mathrm{A}}$, and what any other agency learns is a subset of this.

The protocol in §3.7 is vulnerable to breaches of privacy. For instance, if the matrix **Z** in **Step 1** of the protocol is such that ($\mathbf{I} - \mathbf{Z}\mathbf{Z}^T$) contains a column with all zeros except for a non-zero constant in one row, then agency A learns from $(\mathbf{X}^A)^T\mathbf{W}$ the value of agency B's data for the data subject in that row. Of course, this problem is detectable by agency B, which could then simply not respond.

Even when the agencies are semi-honest, disclosures might be generated because of the values of the attributes themselves. Note that these issues are the result of computation of $\mathbf{X}^T\mathbf{X}$ by any method, and are neither caused not alleviated by use of the protocol in §3.7. As a simple example, suppose $\mathbf{X}^A$ includes an attribute that equals zero for all but one of the data subjects. Even with a legitimate **Z**, then $(\mathbf{X}^A)^T\mathbf{X}^B$ will reveal that subject's value of $\mathbf{X}^B$.

Similar problems arise if $\mathbf{X}^A$ is sparse and there is reliable prior information on the locations of non-zero entries. In this case, the effective number of degrees of freedom in $\mathbf{X}^A$ is less – perhaps much less – than $n \times p_A$.

Still other issues arise. For instance, attributes might satisfy constraints of the form "Gross income ≥ net income plus federal tax plus state tax," which have effectively unpreventable potential to reveal information. Similarly, if one record in $\mathbf{X}^A$ contains a dominant attribute value (Willenborg and de Waal, 2001), for instance to the extent that it exceeds 90% of the sum of all values of that attribute, then that value is revealed approximately in all of the $(\mathbf{X}^A)^T\mathbf{X}^B$.

Finally, there may be problems that are revealed only when analyses are conducted. For example, agency A can perform a regression of every other attribute on its attributes. Should one of those regressions have a high coefficient of determination (r^2), then A knows that has in its own data a good predictor of that attribute, even if it is owned by another agency.

One way in which agencies might attempt to deal with such issues would be to share only some of their attributes. How this would work and whether it would be effective in preserving privacy are subjects for future research.

5.2 Linear Regression

For clarity, we now write the ($n \times p$)-dimensional data matrix **X** of (26) as

$$\mathbf{X} = [X_1 \cdots X_p], \tag{27}$$

where each X_i has dimension $n \times 1$, and belongs to exactly one of the $\mathbf{X}^\alpha$. To account for intercepts in regressions, we assume the first column of **X** in (27) consists entirely of ones (and is owned by agency A, although that is not material). For simplicity assume that all other attribute means are zero. This is not restrictive, since if the agencies are willing to share the on-diagonal blocks of the covariance matrix, they would certainly be willing to share means.

Assume that $\mathbf{X}^T\mathbf{X}$ is calculated using the method described in §6.1. Following that computation, some checking may be necessary. For instance, it is possible that two agencies hold perfectly correlated attributes without knowing it. Simple issues of this sort are detectable from $\mathbf{X}^T\mathbf{X}$, and readily addressed. More complex issues (for instance, complex multi-agency multi-collinearity) are problematic.

A regression model of a response attribute $X_{\text{resp}} \in \{X_1;, \ldots, X_p\}$ on a set of predictors

$$\mathbf{X}_{\text{pred}} \subseteq \{X_0, \ldots, X_p\} \setminus \{X_{\text{resp}}\}$$

is of the form

$$X_{\text{resp}} = \mathbf{X}_{\text{pred}}\,\beta + \varepsilon \tag{28}$$

where $\varepsilon \sim N(0,\ \sigma^2)$. The maximum likelihood estimates of β and σ^2 as well as the standard errors of the estimated coefficients, can be easily obtained from $\mathbf{X}^T\mathbf{X}$, for example using the sweep algorithm (Beaton, 1964; Schafer, 1997). Indeed, only the restriction of $\mathbf{X}^T\mathbf{X}$ to $\mathbf{X}_{\text{pred}} \cup (X_{\text{resp}})$ is needed.

5.3 Model Diagnostics

Estimated regression coefficients are of limited value when a regression model such as (28) does not describe the data adequately. Hence, model diagnostics are essential. The types of diagnostic measures available in vertically partitioned data settings depend what additional information the agencies are willing to share. Diagnostics based on residuals require the predicted values

$$\widehat{X_{\text{resp}}} = \mathbf{X}_{\text{pred}}\hat{\beta} = \mathbf{X}_{\text{pred}}\left[\mathbf{X}_{\text{pred}}^T\mathbf{X}_{\text{pred}}\right]^{-1}\mathbf{X}_{\text{pred}}^T X_{\text{resp}}. \tag{29}$$

These can be calculated using the secure matrix multiplication protocol. Alternatively, since each agency can calculate $\hat{\beta}$ from $\mathbf{X}^T\mathbf{X}$, each can compute that portion of $\mathbf{X}_{\text{pred}}\hat{\beta}$ associated with its attributes, and these vectors can be summed across agencies using secure summation (§3.6).

Once the predicted values are known, the agency owning the response attribute X_{resp} can calculate the residuals $X_{\text{resp}} - X_{\text{pred}}\hat{\beta}$. If that agency is willing to share these with the other agencies, each agency can perform plots of residuals its predictor attributes and report the nature of any lack of fit to the other agencies. Sharing residuals $X_{\text{resp}} - X_{\text{pred}}\hat{\beta}$ also enables all agencies to obtain Cook's distance measures (Cook and Weisberg, 1982), since t

$$\mathbf{H} = \mathbf{X}_{\text{pred}}\left[\mathbf{X}_{\text{pred}}^T\mathbf{X}_{\text{pred}}\right]^{-1}\mathbf{X}_{\text{pred}}^T.$$

We note that the diagonal elements of $\mathbf{H}$ can be used as well to generate standardized and Studentized residuals. Additionally, the agency with X_{resp} can make a plot of the residuals versus predicted values, and a normal quantile plot of the residuals, and report any evidence of model violations to the other agencies. The number of residuals exceeding certain thresholds, i.e., outliers, also can be reported.

5.4 Other Analyses

The approach outlined in §6.2 and 6.3 extends readily to other classes of statistical analyses, although the equity issues raised in §6.1 need to be considered in detail in each instance.

A simple example is weighted least squares regression; see also the discussion of weights in §9.5. If $\mathbf{T}$ is the $n \times n$ (diagonal) matrix of weights, then each agency pre-multiplies its attributes by $\mathbf{T}^{1/2}$, and the analysis proceeds as described in §6.2 and 6.3.

To run semi-automatic model selection procedures such as stepwise regression, the agencies can calculate covariance matrices securely, then select models based on criteria that are functions of the full covariance matrix $\mathbf{X}^T\mathbf{X}$, such as the F-statistic or the Akaike Information Criterion (AIC).

It is also possible to perform ridge regression (Hoerl and Kennard, 1970) securely. Ridge regression shrinks estimated regression coefficients away from the maximum likelihood estimates by imposing a penalty on their magnitude. Written in matrix form, ridge regression seeks $\hat{\beta}$ that minimizes

$$\text{Ridge}(\beta;\, \lambda) = (X_{\text{resp}} - \mathbf{X}_{\text{pred}}\beta)^T\, (X_{\text{resp}} - \mathbf{X}_{\text{pred}}\beta) + \lambda\beta^T\beta, \tag{30}$$

where λ is a specified constant. The ridge regression estimate of the coefficients is

$$\hat{\beta}_{\text{Ridge}} = (\mathbf{X}^T_{\text{pred}}\mathbf{X}_{\text{pred}} + \lambda\mathbf{I})^{-1}\, \mathbf{X}^T_{\text{pred}}X_{\text{resp}}. \tag{31}$$

Once $\mathbf{X}^T_{\text{pred}}\mathbf{X}_{\text{pred}}$ and $\mathrm{X}^T_{\text{pred}}X_{\text{resp}}$ have been calculated securely, each agency can perform the calculation in (31).

5.5 Data Mining

The literature on PPDM for vertically partitioned data includes treatments of

Decision Trees: Samet and Miri (2011);

***k*-Means Clustering**: Vaidya and Clifton (2003);

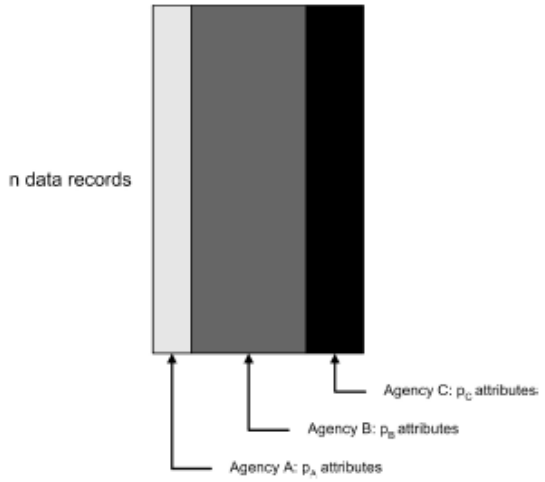


Figure 4: Pictorial representation of vertically partitioned data.
The blocks represent the data values held by three agencies.

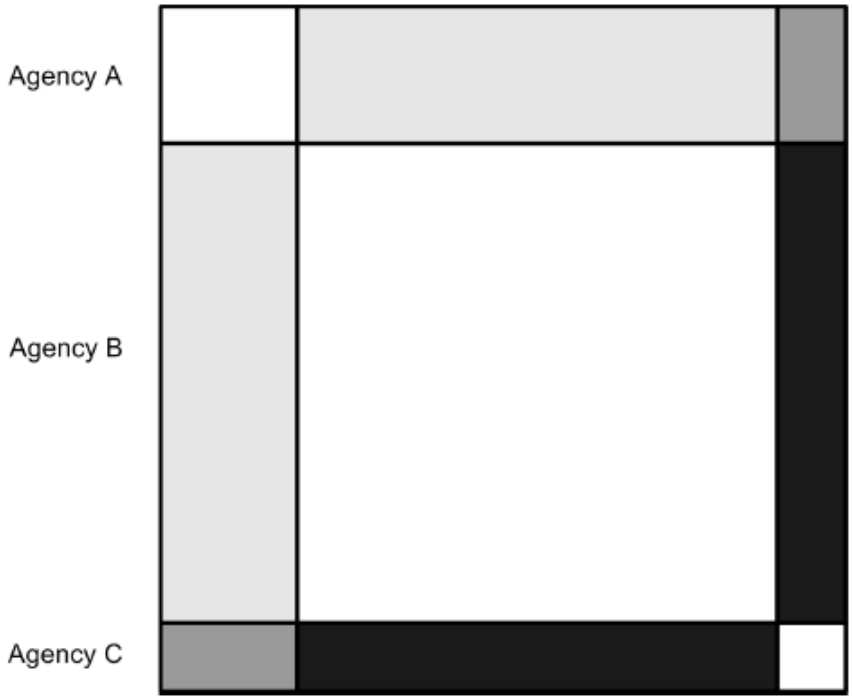


Figure 5. Pictorial representation of the full data covariance matric $\mathbf{X}^T\mathbf{X}$ for three agencies.

Association Rules: Vaidya and Clifton (2002);

Neural Networks: Samet and Miri (2011).

A good entry point to this literature is Vaidya et al. (2006).

VI. COMPLEX PARTITIONS

Little is known about complex partitions such as that in the right-hand panel in Figure 1. Here we simply outline some issues and initial approaches to them.

To begin, there may be problems associated with merely knowing which databases contain which attributes about which data subjects, and there are further issues involved in determining which subjects are common across the databases, a problem to which oblivious polynomial evaluation (§3.4) is relevant.

One approach (Reiter et al., 2004; Karr et al., 2007) is to view complicated data partitions as incomplete data sets – the global database is construed as a flat file with missing values in those records not common to all parties – and then to develop secure versions of techniques used for analyzing incomplete data sets. For instance, one can specify a joint distribution for the complete data, and then to use the EM algorithm (Dempster et al., 1977) to estimate the parameters of that distribution. If the associated sufficient statistics can be calculated using SMPC, then a secure EM algorithm is feasible, as we illustrate briefly for data following a multivariate normal distribution.

For simplicity, we assume that the owners share *globally unique* identifiers of the records in their databases, in order to identify records that are common to multiple databases, and that matching on these unique identifiers can be done without error. Finally, we assume that there are no duplicate attributes. In reality, means must be developed for dealing with failure of each of these assumptions.

The sufficient statistics – sums, sums of squares, and sums of cross-products of the data values – can be computed securely by the following protocol.

Let M be the number of data missingness patterns; for example, in Figure 6, $M = 5$: partitioning the attributes into four blocks (corresponding to agencies 1, {2,3}, 4 and 5), there are five patterns: blocks 3 and 4, block 3, blocks 2 and 3, block 4, and no blocks. For $m = 1, \ldots, M$, let $\mathcal{D}_m$ be the set of data elements with missingness pattern m.

To begin the secure EM protocol, the owners group records by missingness pattern, which is possible since they have shared unique identifiers. They next compute and share two tables of sufficient statistics needed by the EM algorithm. The first table has M rows corresponding to the missingness patterns and p columns corresponding to all of the attributes in the global database. The entry for row m and column j is the sum of the observed y_j for those records with the missingness pattern associated with row m. When there are no common attributes, each sum is computed by only one owner. When there are common attributes, it is computed using secure summation.

The second table has M rows corresponding to the missingness patterns and $p(p + 1)/2$ columns corresponding to the inner products of all p variables in the data set, including the sums of squares. The entry in the table for row m and the column associated with attributes (j, k) is the $\Sigma y_j y_k$ for those records with the missingness pattern of row m. With no common attributes, each cross-product entry in the table is derived from a single dot product involving two agencies, calculated using a secure dot product protocol, which is a special case of the secure matrix multiplication protocol in §3.7.

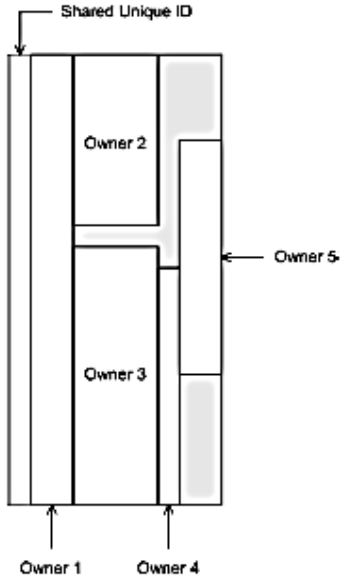


Figure 6: Example of complex data partitioning with five database owners.

Once each owner has these two tables, it has all the information needed to run the EM algorithm (Schafer, 1997) independently of others. Further inference from the data, for example, fitting regression models, is then possible without additional error.

VII. IMPLEMENTATION ISSUES

Implementation of the protocols described in the preceding sections is difficult and subtle.

7.1 Preprocessing

Implementation of the protocols described here entails significant pre-processing overhead. For instance, for horizontally partitioned data, the database owners need to ascertain that their sets of data subjects are in disjoint, and must have the attributes in the same units. Attributes must be in the same order, or at least (as in §8.5) have the same names. For vertically partitioned data, there are, in addition, the more difficult problem of securely linking records across the databases, and of duplicated or inconsistent attribute values and incomplete records.

7.2 Communication

We have ignored communication issues in this paper. The "round-robin" nature of secure summation and related protocols lead to passing of multiple encrypted messages, especially for the star topology in §8.5. Even so, for the kinds of analyses we have considered and the scale of the number of databases (states), the number of bytes being communicated is small by almost any standard. For additional discussion, see Samet and Miri (2011).

7.3 Dishonesty

Throughout this paper, we assume that the database owners are semi-honest: they adhere strictly to protocols designed to preserve privacy, and perform calculations using their real data. We believe the semi-honesty assumption is realistic for many data integration settings, including government agencies seeking to perform combined analyses using their data. Furthermore, we assume that the owners do not collude with each other. Here we consider more carefully problematic aspects of the semi-honesty assumption, and outline a possible path to mitigate them.

In the secure regression setting of §4.5, suppose that all owners other than j are semi-honest, but that instead of contributing $[X^j\ y^j]^T\ [X^j\ y^j]$ to the summation in (13), owner j puts in 0. Then what every other owner receives at the end of the process and thinks is $[X\ y]^T\ [X\ y]$ is in fact

$$([X\ y\}^T\ [X\ y])_{-j} = \sum_{k \neq j} [X^k\ y^k]^T\ [X^k\ y^k].$$

Owner j can then merely add $[X^j\ y^j]^T\ [X^j\ y^j]$ to obtain the correct $[X\,y]^T\,[X\,y]$, and proceed to perform the correct regression, leaving other owners unaware that they do not have the correct regression.

Active deception can cause more severe problems. If, instead of $[X^j\ y^j]^T\ [X^j\ j^j]$ or 0, owner j adds *false* values $[X^j_{\text{false}}\ y^j_{\text{false}}]^T [X^j_{\text{false}}\ y^j_{\text{false}}]$, then what each other owner thinks is $[X\ y]^T\ [X\ y]$ is now

$$\sum_{k=1,k\neq j}^{K} [X^k\ y^k]^T\ [X^k\ y^k]\ +\ [X^j_{\text{false}}\ y^j_{\text{false}}]^T\ [X^j_{\text{false}}\ y^j_{\text{false}}].$$

Owner j obtains the correct value of $[X\ y]^T\ [X\ y]$ by subtracting $[X^j_{\text{false}}\ y^j_{\text{false}}]^T\ [X^j_{\text{false}}\ y^j_{\text{false}}]$ and adding $[X^j\ y^j]^{\mathrm{T}}\ [X^j\ y^j]$, and then can calculate the correct regression. Unless $[X^j_{\text{false}}\ y^j_{\text{false}}]^T\ [X^j_{\text{false}}\ y^j_{\text{false}}]$ is egregiously false, the other owners will be unaware that they have a completely bogus regression.

These examples show that the secure regression protocol in §4.5, viewed as a multi-player game, is not a Nash equilibrium. Each owner has unilateral incentive not to be semi-honest, a behavior that is effective if the others are semi-honest.

The concept of *partially trusted third parties (PTTPs)* (Karr et al., 2007) may be able to reduce unilateral incentives to cheat in situations where the results of interest are (non-additive) functions of *two or more other quantities*. For secure regression, PTTPs work because $\hat{\beta}$ is a function of $X^T X$ and $X^T y$, which can be computed independently by secure summation. They also would work for secure averages such as (8), with the number and denominator computed using secure summation.

A PTTP is, in effect, a dataless owner that initializes secure summations, receives the results, calculates quantities of interest, and shares *only the final result of the computations* with the database owners. To illustrate in the setting of secure regression, we restrict attention to regression coefficients $\hat{\beta}$. The PTTP protocol is then as follows:

Initialization. The PTTP creates a matrix R of dimensions $(p + 1) \times (p + 1)$, where p is the number of predictors, each of whose entries is a very large random number.

Iteration. Using either an ordinary secure summation protocol (§3.6) or the Secure Computation System (SCS) of §8.5, the database owners each add their $[X^j \; y^j]^T \; [X^j \; y^j]$.

Computation. When it receives back $R + [X \; y]^T \; [X \; y]$ from the final owner to contribute, the PTTP subtracts R and then calculates $\hat{\beta}$ using (12).

Dissemination. The PTTP disseminates $\hat{\beta}$ to the owners. It does not disseminate $[X \; y]^T \; [X \; y]$.

The major advantage of the PTTP protocol is that, because the computation of $\hat{\beta}$ is hidden from the database owners, it is harder to undo the effects of cheating the effects of cheating. Let $\hat{\beta} - j$ denote the (true) coefficients for the regression involving all owners other than j. If instead of adding $[X^j \; y^j]^T \; [X^j \; y^j]$, owner j adds $[X^j_{\text{false}} \; y^j_{\text{false}}]^T \; [X^j_{\text{false}} \; y^j_{\text{false}}]$, and then receives false coefficients $\hat{\beta}_{\text{false}}$ in order to recover true coefficients, j must somehow remove the effects of $[X^j_{\text{false}} \; y^j_{\text{false}}]^T \; [X^j_{\text{false}} \; y^j_{\text{false}}]$, from $\hat{\beta}_{\text{false}}$ and then "add" the effects of $[X^j \; y^j]^T \; [X^j \; y^j]$.

The most significant drawback is that the PTTP ends up knowing more than the database owners. For secure regression, the PTTP knows $[X \, y]^T \; [X \, y]$, while the owners know only $\hat{\beta}$. This may be acceptable in some settings, but will not be in others. There are also ways around it. For instance, for secure regression, the owners could each replace their X^j by $Z^j = X^j B$, where B is an invertible $p \times p$ matrix not known to the PTTP, and the regression could be performed using the PTTP and the Z^j. Finally, each owner would multiply the $\hat{\beta}$ received from the PTTP by B to recover the true estimated coefficients.

7.4 Opting Out

As currently construed, the protocols presented here do not allow database owners to opt out of a secure computation based on the results of the analysis, should they feel that those results are too informative about their data. At some level, of course, this is an unsolvable problem: a decision that requires the results cannot be made without them. Cooperating owners could, obviously, agree after the fact not to release publicly the results an analysis. On the other hand, it is possible to use secure

summation to allow agencies to opt out beforehand on the basis of (k, p)-rules in SDL (Willenborg and de Waal, 1996, 2001) and even to make opting out anonymous. It seems likely that intermediate procedures exist, but they remain undeveloped.

7.5 Prototype Implementation: NISS Secure Computation System

As alluded to in §3.6, implementation of protocols such as that for secure regression presents significant challenges. A prototype Secure Computation System (SCS) developed by the National Institute of Statistical Sciences (NISS) (Karr et al., 2007) attempts to illuminate and address many of these.

The SCS uses a network-based server-client model implemented as a star topology, illustrated in Figure 7, and operates on the public internet. The database owners are the clients, but they never communicate directly with each other. Rather, they communicate only with the server. There are multiple reasons for this. First, the server can hide from the clients the order in which they provide information, increasing the difficulty of collusion – owners cannot easily figure out who should collude with whom against whom, although not preventing it entirely. (Methods that guarantee protection against collusion entail significantly more communication overhead (Ben-Or et al., 1988).) Second, authentication and encryption are much more efficient with the server. And finally, process management is vastly simplified.

On the other hand, since the server is not an owner, it must not be able to "read" any of the information it is passing between clients, which leads, as described below, to a dual encryption. In fact, in the SCS the server is not even aware which protocol the clients are performing.

Here is a simplified description of SCS functionality.

Set-Up. Prior to actually performing the analysis, the owners/clients must agree on who is participating, which protocol (for instance, secure regression) will be performed, metadata issues, a time at which the protocol is to be performed and a symmetric *clients-only* encryption key, which is not revealed to the server. The time and IP addresses of clients are transmitted to the server by means of a reservation system.

Log In. At the reserved time, the server listens for clients to log in. As each client logs in, it receives the server's public key – to use when sending messages to the server. It also generates a (private key, public key) pair for encryption of messages from the server to it, and sends the server the latter.

Initiation. When all clients have logged in, the server randomly selects an order for the clients, and sends an "Initiate protocol" message to the first client.

The structure and encryption of messages from the server to the client is shown in the top panel in Figure 8. The entire message is encrypted using the client's public key, which prevents its being read by the other clients or an outsider. The Tag is generated by the server and used for authentication: the server will not respond to any message not containing the correct tag. The Client Message contains information passed from the server to the client. Examples are "Initiate protocol" and "Protocol completed." The remainder of the message is empty at the start of the

process, or else, as described below, has been constructed and encrypted by another client. It is not readable by the server.

Client-Side Processing. When it receives a message from the server, a client decrypts it using its own private key, sets the Tag aside, parses the Client Message and acts accordingly. For example, if the message is "Initiate protocol" and the clients have agreed to perform secure summation, the client would generate the random number R, add its v_j to it, construct the Payload, which is simply $R + v_j$, set Operation to "Secure summation: add value," which tells the next client what to do, concatenate the two and encrypt the result with the clients-only key. It would then concatenate the saved Tag, a Server Message and the encrypted Operation and Payload, encrypt that entire object with the server's public key, and send it to the server.

When there are no problems, Server Message will be of the form "Client operation successful." Were there a problem, for example, if a client in secure regression could not locate its data file, Server Message would inform the server that there had been a problem. The prototype version of the SCS makes no attempt to recover from errors, and the server merely informs the clients that the process has terminated unsuccessfully.

Succeeding clients receive a Tag and Client Message of the form "Continue," decrypt the Operation and Payload, perform the indicated operation (Example: "Secure summation: add value") to generate a new payload (Example: the payload it received plus its v_j), encrypt the two with the clients-only key, and then proceed as described above.

The initiating client recognizes that the computation phase of the protocol is complete when it receives a second message from the server. Because it knows the protocol, it knows that it should then remove random initializers (Example: subtract R from $V + R$ to obtain V), set Operation to "Read", set Payload to V, and proceed.

The initiating client recognizes that the entire protocol is complete when it receives a third message from the server, in which case it sets Server Message to "Protocol complete" sends one final message to the server and shuts itself down. The server then tells the remaining clients that the protocol is complete, which they acknowledge and shut themselves down.

Server-Side Processing. When it receives a message (purporting to be) from a client, the server first decrypts it using its private key. It then compares the Tag to that of the last message it sent out, and if the two are not identical, either ignores the message and waits for the "correct" message, or declares a problem. If the Tag is correct, the server parses the Server Message, and acts accordingly. In most cases, the message would be "Client operation successful" or "Protocol complete." As noted previously, in the current SCS, if the Server message is "Client encountered problem," the server would terminate the protocol.

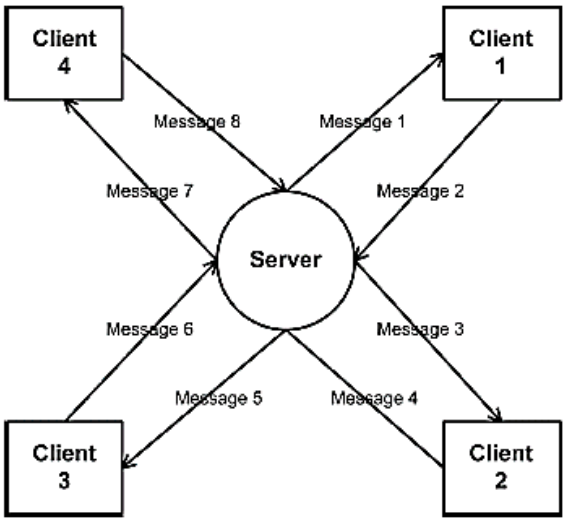


Figure 7: Star topology structure of the SCS illustrated with four clients. Messages are never passed directly from one client to another but rather only from the server to a client, or *vice versa*.

The server continues sending messages to clients in the same order (cyclically) until Server Message is "Protocol complete." Note that the server is never able to read either the Operation or the Payload.

The client software for the SCS consists of a number of graphical user interfaces (GUIs) and computational engines. Figure 9 shows those associated with secure regression.

7.6 Other Implementations

The SAS® system contains some capabilities for analysis of data in distributed – including Hadoop and no-SQL – databases. One early reference is McIntyre (1992). The SAS tools may be oriented to business rather than statistical analysis applications, and may involve more superstructure than SEAs can comfortably accommodate.

Some implementations of "cloud computing" may support statistical analyses that are performed by sharing data summaries (not necessarily sufficient statistics). In this context, the main goal is to avoid moving large volumes of data.

VIII. GAPS IN UNDERSTANDING

A broader discussion of "what we do not understand" appears in Karr (2010), some of which is excerpted here.

At a very high level, we lack a risk-utility paradigm (Cox et al., 2011) for secure analysis of distributed data. What do the database owners gain from the analysis? What do they lose? The discussion of heterogeneity in §9.3, raises – but only that – one possibility.

Moreover, the beneficiaries of the analysis may not be only the database owners. Indeed, in the framing example of SLDS, while SEAs do benefit from comparing analyses of their own data to analyses of the national data, the envisioned beneficiaries also include national policy formulators and the education sciences research community.

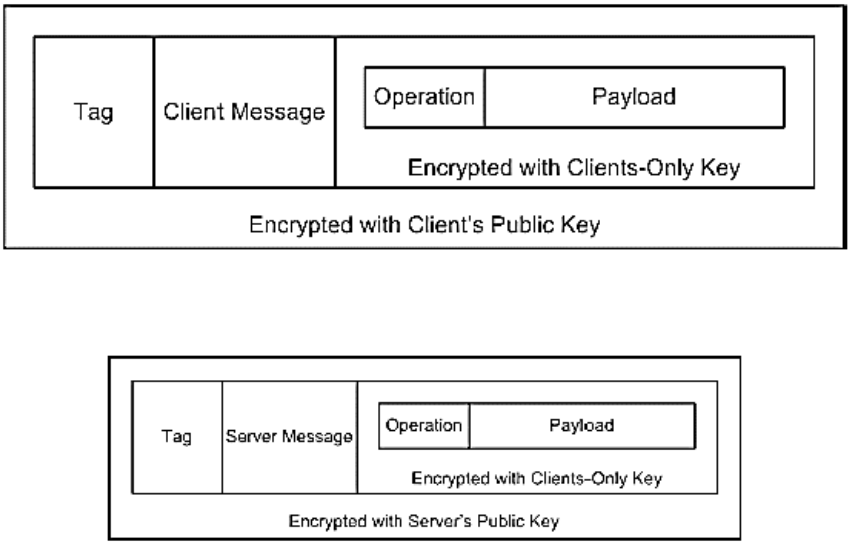


Figure 8: Encryption used by the SCS. *Top*: Encryption of server-to-client messages. *Bottom*: encryption of client-to-server messages. Portions of messages encrypted with the clients-only key, which are shaded in gray, cannot be read by the server.

8.1 Data Subject Confidentiality

The techniques on which this paper is based were developed to protect distrustful database owners from one another. They are not meant to do what traditional SDL (Doyle et al., 2001; Willenborg and de Waal, 2001) does – protect the confidentiality of individual data records, and no studies have been conducted regarding whether they do.

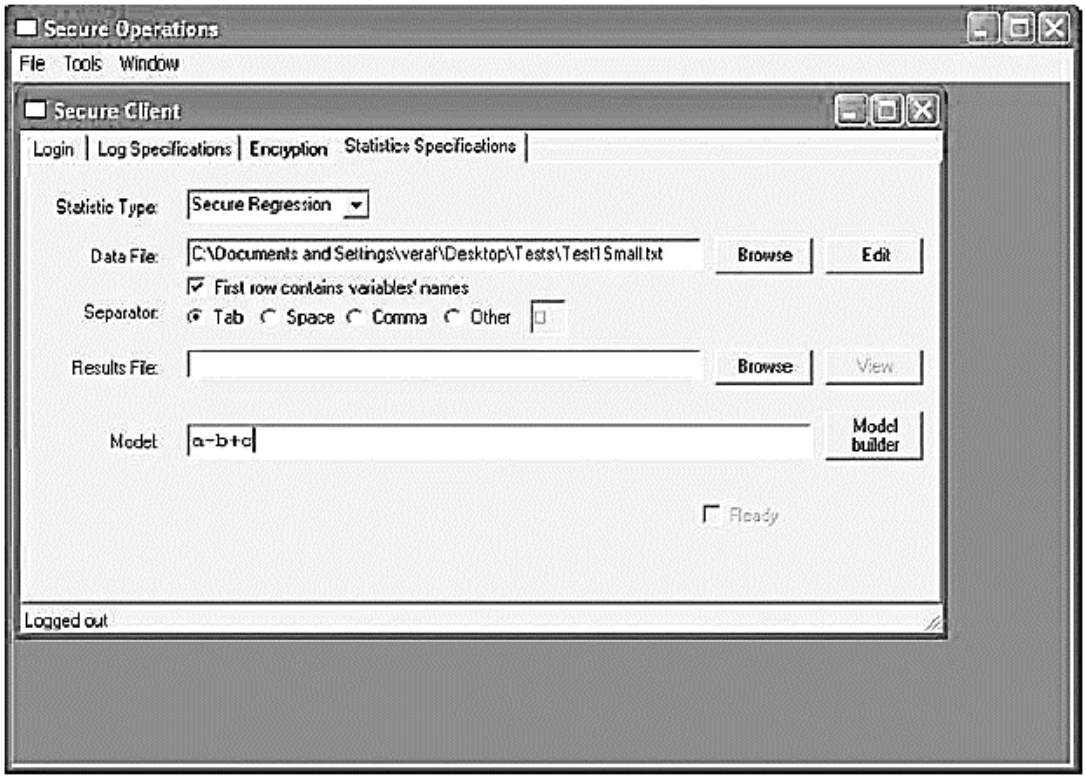
Some analyses are very threatening to confidentiality. For instance, in the case of regression for vertically partitioned data, the owner of the database containing the response may have promised protection to the data subjects, but all the other database owners will have (possibly, rather good) predictions of the response.[7]

The heterogeneity discussion in §9.3 shows that there can also be issues for horizontally partitioned data. In the context of Figure 11, suppose that "leftmost" agency has a few but only a few data points in the center region of the graph. Using only its own data, that agency cannot reliably predict the response for those few cases, but with combined, secure analysis, it would predict those values more accurately, and therefore better able (to its benefit) to detect measurement or reporting errors, which decreases subject-level confidentiality.

The secure summation framework appears, in principle, to be compatible with differential privacy (Dwork, 2006; Dwork et al., 2006), in the sense that prior to secure summation, each database owner could execute the underlying database queries in a differentially private manner. To provide some concreteness, if the object were to compute a global – over all databases – mean $\mu = \Sigma_j n_j \mu_j / \Sigma_j n_j$, then each database owner could replace its μ_j by μ^*_j, the latter calculated using a differentially private algorithm. In the simplest case, $\mu^*_j = \mu_j + Z_j$, where Z_j is a randomly

[7] There is a deeper issue about SDL here: must what an agency is able to infer from the data be accorded the same protection as respondent-provided data? In particular, what protection should be applied to edited and imputed values? Prevailing practice seems to be to make no distinction; to the extent that it is over-conservative, data quality suffers unnecessarily.

chosen “noise” whose scale reflects an exogenously chosen privacy bound conventionally written as ε_j. No details have been worked out, but under many noise structures, $\mu^* = \Sigma_j n_j \mu^*_j / \Sigma_j n_j$ would be less noisy than the μ^*_j.



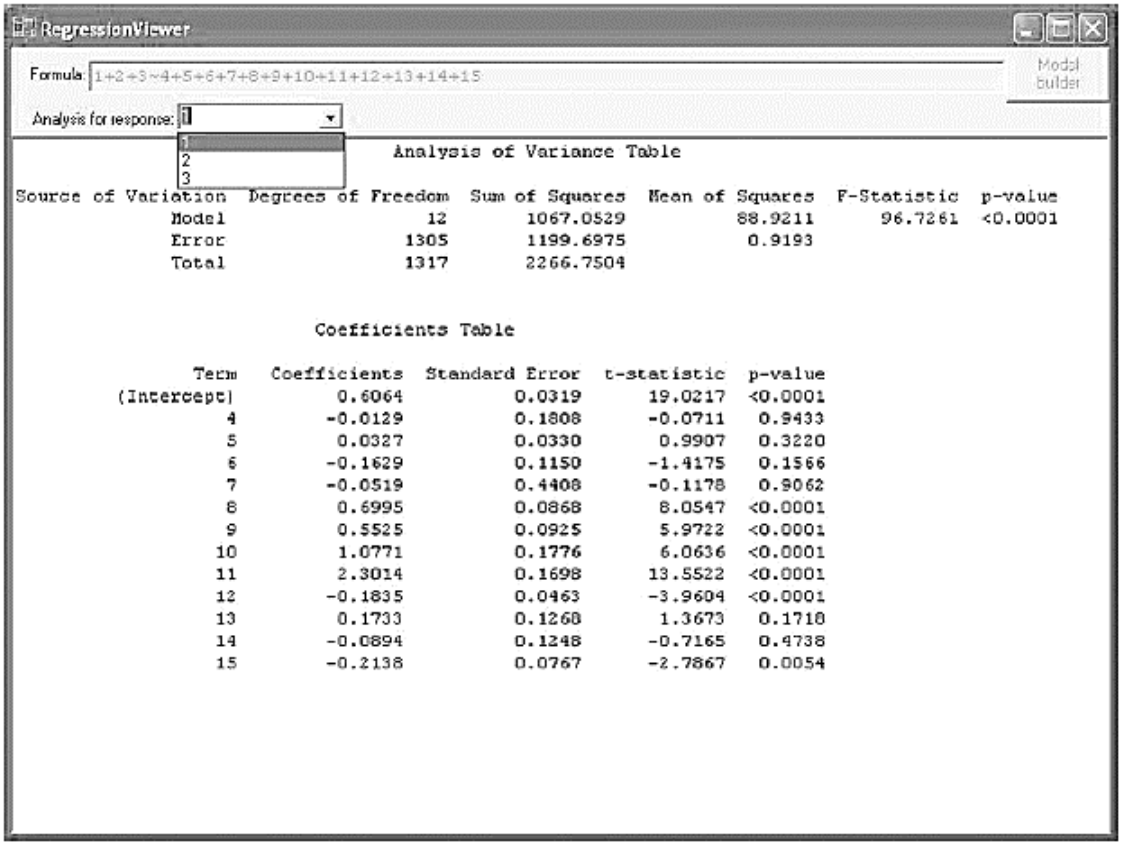


Figure 9: Selected components of the GUI for the client-side SCS software. Top: GUI to specify secure regression protocol, location and field separators for data file, location for file containing results, and model. Bottom: GUI showing output from secure regression.

8.2 Missing Data

Given current knowledge, little can be done regrading missing data values except to leave them as missing and to adjust analyses accordingly. Of course, it is possible to impute missing values separately in each database, and then to perform analyses as described above. Unless the same imputation methodology is employed for all databases, calculation of imputation of imputation-engendered uncertainties is problematic. Effectively nothing is known whether the models typically employed for imputation, which are underlain by Bayesian Markov chain Monte Carlo (MCMC) computations, can be performed in the secure analysis framework of this paper. This is a pity, because individual database owners would benefit in clear and quantifiable ways from such a protocol.

Initial approaches to addressing systematically missing values in the context of vertically partitioned data are outlined in Karr et al. (2007) and Reiter et al. (2004); these have not been implemented.

8.3 Database Heterogeneity

The notion that SMPC allows participants to learn only what is knowable from their input and the answer is most persuasive when agencies contribute approximately equally to the process. When they do not, the “information” surrendered can vary dramatically. The simplest instance of this is

when agencies have unequal database sizes.[8] Figure 3 provides some insight: the larger the database, the more closely the global regression resembles each company's regression.

A natural question in the context of the same example is that, if the companies knew the sizes of each others' databases, should companies 1, 2 and 4 allow company 3 to participate, since it has so little data? Figure 10 sheds some light on this. There, the *x*-axis again contains the coefficients for the four-company regression and the *y*-axis those for the regression for companies 1, 2 and 4, excluding company 3. Although close, the coefficients are not identical, so companies 1, 2 and 4 *do* gain from including company 3. Whether they gain enough to justify the information they surrender to company 3, which cannot even perform the regression on its own, is question not yet formulated to the point of being answerable.

A second issue is *data heterogeneity* across agencies. It is clear that there is something fundamentally different between the top panel in Figure 11, where the three agencies possess 2-dimensional data lying in the same region, and the bottom panel, where the *x*-variable ranges are very different. For the latter, a properly performed analysis would elucidate a global quadratic structure in the data that is invisible to each agency on its own, but the issue raised in §2 as "a massive shortcoming" could keep such a model from being considered.

There is a deep point here. In the "top panel of Figure 11" context, the principal benefit to the agencies is increased sample size, whereas in the "bottom panel of Figure 11" context, there is dramatic – yet hard-to-attainable – benefit, if only the right analysis were done.

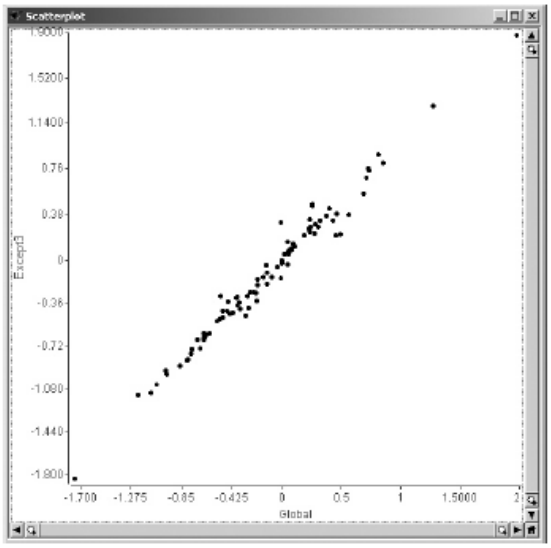


Figure 10: Scatterplot of regression coefficients for all four companies (*y*-axis) versus those for the regression for companies 1, 2 and 4, but excluding company 3 (*x*-axis)

Problems also arise when there is differential model fit across agencies. In the regression setting, if the global fit of the global model is good, but the fit of the global model to an agency's data is poor, then it has potentially learned more about the other agencies' data than they have about its data.

[8] To get a sense of relevance, populations of U.S. states vary by a factor of 100.

8.4 Data Quality

The most difficult form of heterogeneity may be differential data quality. Data quality itself is poorly understood from a statistical perspective (Karr et al., 2006b), and we do not know how to accommodate differences in data quality across databases in the setting of this paper. More concrete data quality problems may be present but in the material presented above would be ignored. For instance, in the case of vertically partitioned data, two agencies may have the same attribute but different values for it. This problem is detectable (for instance, using a secure dot product (Samet and Miri, 2011)), but fixable only using external domain knowledge. Similarly, the possibility that the agencies databases pertain to the same subjects but not at the same time (e.g., health data are from one year and economic data from another) can only be dealt with "off-line."

8.5 Weights

In the framing SLDS example, sample weights are not relevant. Weights may be present in other settings, however. Weighted analyses, such as Horvitz-Thompson estimation of finite population quantities (Horvitz and Thompson, 1952) and, given replicate weights, calculation of standard errors, appear to be straightforward generalizations of methods described in previous sections. For instance, weighted averages can be calculated by secure summation of weighted sums and sums of weights.

What is less clear is whether weighted analyses *per se* comprise threats, regardless of how the analyses are performed. We know at some level that this is true, and know one method to ameliorate the problem (Cox et al., 2011), but are not remotely close to a full understanding.

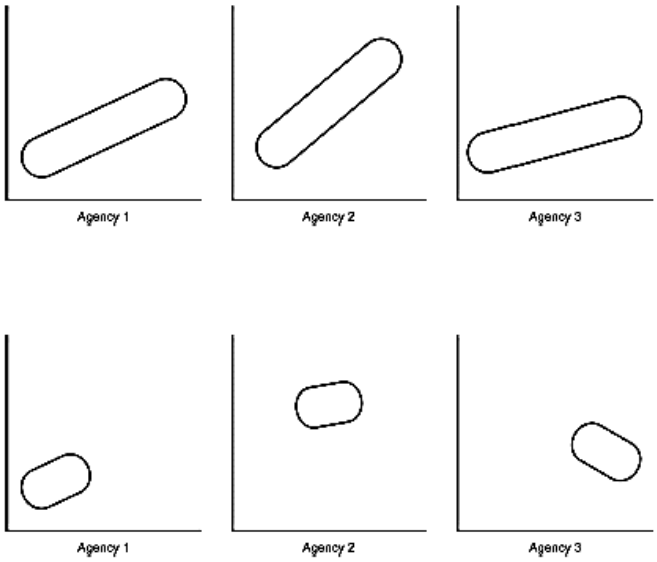


Figure 11: Homogeneous data (top) across databases contrasted with heterogeneous data (bottom) across databases.

REFERENCES

Beaton, A. E. (1964). The use of special matrix operations in statistical calculus. Research Bulletin RB-64-51, Educational Testing Service, Princeton, NJ.

Ben-Or, M., Goldwasser, S., and Wigderson, A. (1988). Completeness theorems for non-cryptographic fault tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pages 1–10, Chicago. ACM.

Benaloh, J. (1987). Secret sharing homomorphisms: Keeping shares of a secret secret. In Odlyzko, A. M., editor, CRYPTO86, pages 251–260. Springer–Verlag. Lecture Notes in Computer Science No. 263.

Bollen, K. A. (1989). Structural Equations with Latent Variables. Wiley, New York.

Christen, P. (2012). Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer–Verlag, New York.

Christen, P. (2013). Overview and taxonomy of techniques for privacy-preserving record linkage.

Clifton, C., Doan, A., Elmagarmid, A., Kantarcioglu, M., Schadow, G., Suciu, D., and Vaidya, J. (2003). Privacy-preserving data integration and sharing. Presented at 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Paris, June 13, 2004.

Cook, R. D. and Weisberg, S. (1982). Residuals and Influence in Regression. Chapman & Hall, London.

Cox, L. H., Karr, A. F., and Kinney, S. K. (2011). Risk-utility paradigms for statistical disclosure limitation: How to think, but not how to act (with discussion). Int. Statist. Rev., 79(2):160–199.

Dempster, A., Laird, N., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. J. Royal Statist. Soc., Series B, 39(1):1–38.

Doyle, P., Lane, J. I., Theeuwes, J. J. M., and Zayatz, L. V. (2001). Confidentiality, Disclosure and Data Access: Theory and Practical Application for Statistical Agencies. Elsevier, Amsterdam.

Du, W., Han, Y., and Chen, S. (2004). Privacy-preserving multivariate statistical analysis: Linear regression and classification. In Proceedings of the 4th SIAM International Conference on Data Mining, pages 222–233.

Dwork, C. (2006). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., and Wegener, I., editors, Automata, Languages and Programming, volume 4052 of Lecture Notes in Computer Science, pages 1–12. Springer–Verlag, Berlin.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In Proc. the 3rd Theory of Cryptography Conference, pages 265–284.

Fellegi, I. P. and Sunter, A. B. (1969). A theory for record linkage. J. Amer. Statist. Assoc., 64:1183–1210.

Ghosh, J., Reiter, J. P., and Karr, A. F. (2007). Secure computation with horizontally partitioned data using adaptive regression splines. Computational Statist. and Data Anal., 51:5813–5820.

Goldwasser, S. (1997). Multi-party computations: Past and present. In Proceedings of the 16th Annual ACM Symposium on Principles of Distributed Computing, pages 1–6, New York. ACM Press.

Hall, R. and Fienberg, S. E. (2013). Privacy-preserving record linkage. Available on-line at https://www.cs.cmu.edu/˜rjhall/linkage_survey_final.pdf.

Heckman, J. J. (2005). The scientific model of causality. Sociological Methodology, 35:1–97.

Hoerl, A. E. and Kennard, R. (1970). Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12:55–67.

Horvitz, D. G. and Thompson, D. J. (1952). A generalization of sampling without replacement from a finite universe. J. Amer. Statist. Assoc., 47:663–685.

Karr, A. F. (2010). Secure statistical analysis of distributed databases, emphasizing what we don't know. J. Privacy and Confidentiality, 1(2):197–211.

Karr, A. F., Feng, J., Lin, X., Reiter, J. P., Sanil, A. P., and Young, S. S. (2005a). Secure analysis of distributed chemical databases without data integration. J. Computer-Aided Molecular Design, November, 2005:1–9. Available online at www.niss.org/dgii/technicalreports.html.

Karr, A. F., Fulp, W. J., Lin, X., Reiter, J. P., Vera, F., and Young, S. S. (2007). Secure, privacy-preserving analysis of distributed databases. Technometrics, 49(3):335–345.

Karr, A. F. and Lin, X. (2010). Privacy-preserving maximum likelihood estimation for distributed data. J. Privacy and Confidentiality, 1(2):213–222.

Karr, A. F., Lin, X., Reiter, J. P., and Sanil, A. P. (2005b). Secure regression on distributed databases. J. Computational and Graphical Statist., 14(2):263–279.

Karr, A. F., Lin, X., Reiter, J. P., and Sanil, A. P. (2006a). Secure analysis of distributed databases. In Olwell, D., Wilson, A. G., and Wilson, G., editors, Statistical Methods in Counterterrorism: Game Theory, Modeling, Syndromic Surveillance, and Biometric Authentication, pages 237–261. Springer–Verlag, New York.

Karr, A. F., Sanil, A. P., and Banks, D. L. (2006b). Data quality: A statistical perspective. Statistical Methodology, 3(2):137–173.

Lindell, Y. and Pinkas, B. (2000). Privacy preserving data mining. In Advances in Cryptology—Crypto2000, Lecture Notes in Computer Science, Volume 1880, pages 20–24, New York. Springer–Verlag.

McCulloch, C. E., Searle, S. R., and Neuhaus, J. M. (2008). Generalized, Linear, and Mixed Models. Wiley, Hoboken, NJ, second edition.

McIntyre, J. (1992). Using the SAS® for remote access and distributed computing. Available on-line at http://www.lexjansen.com/nesug/nesug92/NESUG92057.pdf.

Moore, A. and Lee, M. (1998). Cached sufficient statistics for efficient machine learning with large datasets. *J. Artificial Intell. Res*., 8:67–91.

Naor, M. and Pinkas, B. (1999a). Oblivious polynomial evaluation. Available on-line at http://www.wisdom.weizmann.ac.il/ naor/PAPERS/ope.pdf.

Naor, M. and Pinkas, B. (1999b). Oblivious transfer with adaptive queries. In Advances in Cryptology: CRYPTO '99, volume 1666 of *Lecture Notes in Computer Science*, pages 573–590. Springer–Verlag, New York.

Pearl, J. (2009). Causality: *Models, Reasoning and Inference*. Cambridge University Press, Cambridge, UK, second edition.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1:81–106.

Reiter, J. P. (2003). Model diagnostics for remote access regression servers. *Statistics and Computing*, 13:371–380.

Reiter, J. P., Karr, A. F., Kohnen, C. N., Lin, X., and Sanil, A. P. (2004). Secure regression for vertically partitioned, partially overlapping data. *ASA Proc*.

Rivest, R., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126.

Samet, S. and Miri, A. (2008). Privacy-preserving protocols for perceptron learning algorithm in neural networks. In *Proceedings of The 4th International IEEE Conference on Intelligent Systems (IS '08)*, volume 2, pages 10–65.

Samet, S. and Miri, A. (2011). *Privacy-Preserving Data Mining*. DVM Verlag Dr. Müller & Co. KG, Saarbücken, Germany.

Sanil, A. P., Karr, A. F., Lin, X., and Reiter, J. P. (2004). Privacy preserving regression modelling via distributed computation. In *Proc. Tenth ACM SIGKDD Internat. Conf. on Knowledge Discovery and Data Mining*, pages 677–682.

Sanil, A. P., Karr, A. F., Lin, X., and Reiter, J. P. (2009). Privacy preserving analysis of vertically partitioned data using secure matrix products. J. *Official Statist*., 25(1):125–138.

SAS Institute, Inc. (2014). The mixed procedure: Mixed models theory. Available on-line at http://support.sas.com/documentation/cdl/en/statug/63033/HTML/default/viewer.htm#statug_mixed_sect022.htm.

Schadow, C., Grannis, S., and McDonald, C. (2002). Privacy-preserving distributed queries for a clinical case research network. In Clifton, C. and Estivill–Castro, V., editors, Privacy, Security and Data Mining, volume 14 of *Conferences in Research and Practice in Information Technology*, pages 55–65. Australian Computer Society, Sydney.

Schafer, J. L. (1997). *Analysis of Incomplete Multivariate Data*. Chapman & Hall, London.

Schneier, B. (1995). *Applied Cryptography*. Wiley, New York.

Vaidya, J. and Clifton, C. (2002). Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 639–644, New York. ACM Press.

Vaidya, J. and Clifton, C. (2003). Privacy preserving k-means clustering over vertically partitioned data. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 206–215, New York. ACM Press.

Vaidya, J., Clifton, C., and Zhu, M. (2006). *Privacy Preserving Data Mining*. Springer–Verlag, New York.

Willenborg, L. C. R. J. and de Waal, T. (1996). *Statistical Disclosure Control in Practice*. Springer–Verlag, New York.

Willenborg, L. C. R. J. and deWaal, T. (2001). *Elements of Statistical Disclosure Control*. Springer–Verlag, New York.

Xiao, M.-J., Huang, L.-S., Luo, Y.-L., and Shen, H. (2005). Privacy preserving id3 algorithm over horizontally partitioned data. In *Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pages 239–243.

Yao, A. C. (1982). Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 160–164, New York. ACM Press.