Privacy Preserving Regression

Junhao Hua

2014/10/30

1. Privacy preserving regression modelling via distributed computation

Sanil, Ashish P., Alan F. Karr, Xiaodong Lin, and Jerome P. Reiter.

In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 677-682. ACM, 2004.

The regression problem

Consider the case when we need to fit the standard linear regression model [14]

$$\mathbf{y} = X\beta + \epsilon, \tag{1}$$

where

$$X = [\mathbf{x}_1 \dots \mathbf{x}_p], \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \tag{2}$$

with

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{in} \end{bmatrix}, \tag{3}$$

and

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}, \qquad \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix}. \tag{4}$$

Under the condition that

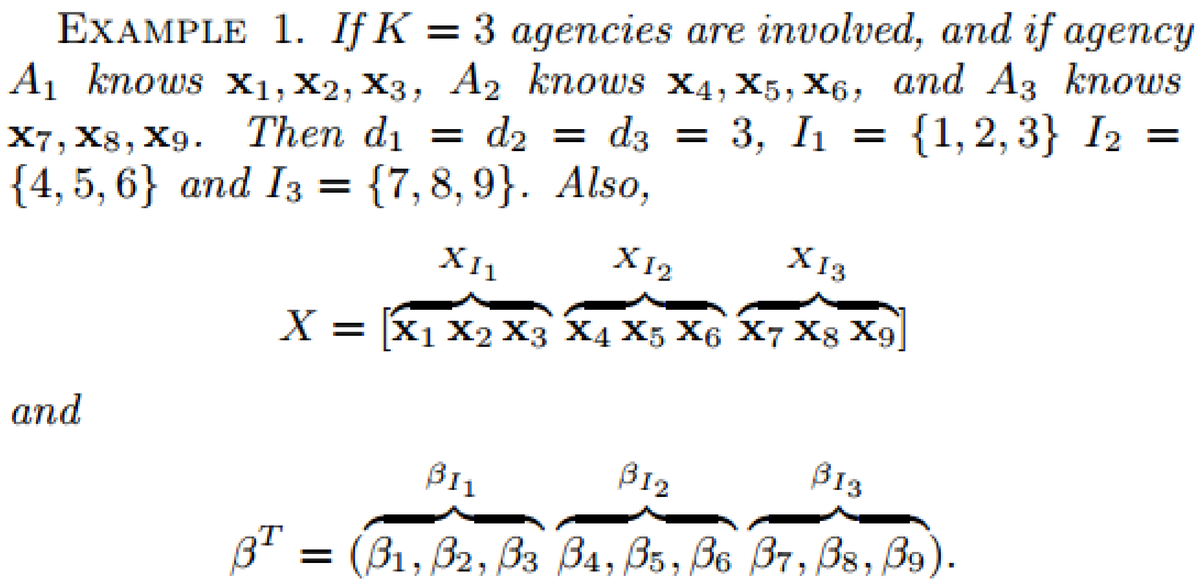
$$\varepsilon \sim \mathcal{N}(0, \sigma^2 I), \tag{5}$$

p: attributes
n: sample size
K>=3: agencies

Least-squares:

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

The regression problem (*con't.*)



EXAMPLE 1. *If $K = 3$ agencies are involved, and if agency A_1 knows $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$, A_2 knows $\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6$, and A_3 knows $\mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9$. Then $d_1 = d_2 = d_3 = 3$, $I_1 = \{1, 2, 3\}$ $I_2 = \{4, 5, 6\}$ and $I_3 = \{7, 8, 9\}$. Also,*

$$X = [\overbrace{\mathbf{x}_1\ \mathbf{x}_2\ \mathbf{x}_3}^{X_{I_1}}\ \overbrace{\mathbf{x}_4\ \mathbf{x}_5\ \mathbf{x}_6}^{X_{I_2}}\ \overbrace{\mathbf{x}_7\ \mathbf{x}_8\ \mathbf{x}_9}^{X_{I_3}}]$$

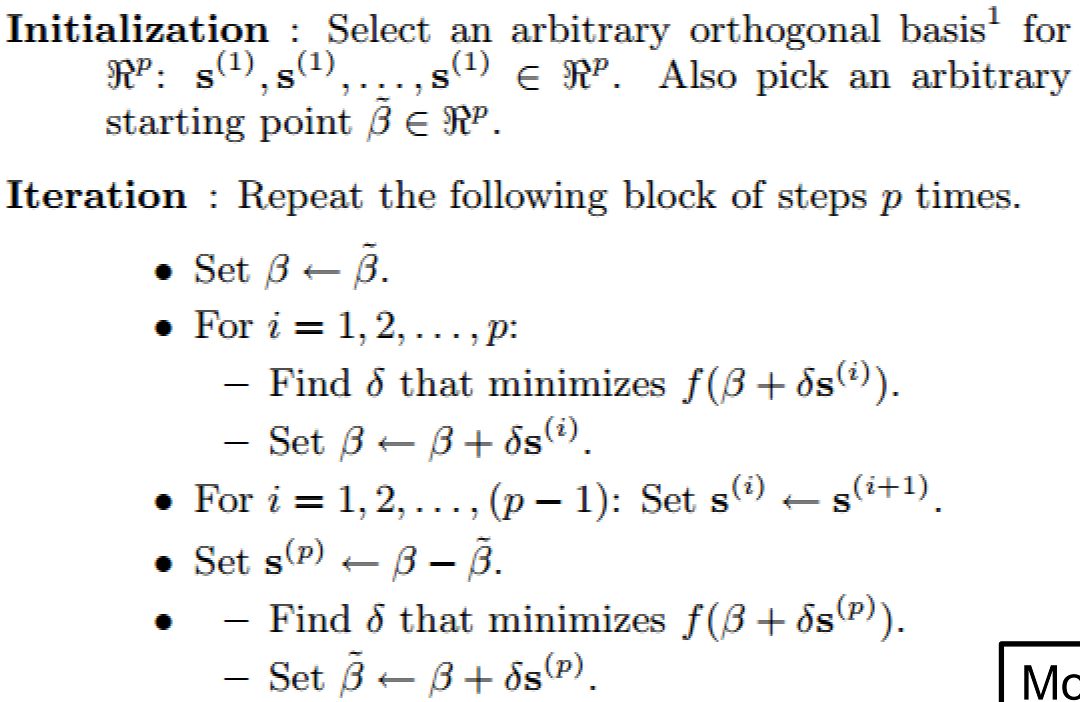
and

$$\beta^T = (\overbrace{\beta_1, \beta_2, \beta_3}^{\beta_{I_1}}\ \overbrace{\beta_4, \beta_5, \beta_6}^{\beta_{I_2}}\ \overbrace{\beta_7, \beta_8, \beta_9}^{\beta_{I_3}}).$$

- Vertically partitioned data
- Not share summary statistics
- Have a lead role
- Align the records
- The attribute sets do not overlap

Preliminaries: Powell’s methods

- Conjugate direction methods
- a derivative-free numerical minimization method that solves the multidimensional minimization problem by solving a series of 1-dimesional line minimization problems.



```
Initialization : Select an arbitrary orthogonal basis[1] for
    $\Re^p$: $\mathbf{s}^{(1)}, \mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(1)} \in \Re^p$. Also pick an arbitrary
    starting point $\tilde{\beta} \in \Re^p$.

Iteration : Repeat the following block of steps $p$ times.

    • Set $\beta \leftarrow \tilde{\beta}$.
    • For $i = 1, 2, \ldots, p$:
        – Find $\delta$ that minimizes $f(\beta + \delta \mathbf{s}^{(i)})$.
        – Set $\beta \leftarrow \beta + \delta \mathbf{s}^{(i)}$.
    • For $i = 1, 2, \ldots, (p-1)$: Set $\mathbf{s}^{(i)} \leftarrow \mathbf{s}^{(i+1)}$.
    • Set $\mathbf{s}^{(p)} \leftarrow \beta - \tilde{\beta}$.
    •   – Find $\delta$ that minimizes $f(\beta + \delta \mathbf{s}^{(p)})$.
        – Set $\tilde{\beta} \leftarrow \beta + \delta \mathbf{s}^{(p)}$.
```

More details…

Preliminaries: Secure Summation

- K>2
- calculate a summation: $v = \sum_{j=1}^{K} v_j$
- Choose *m* to be a very large number known to all the agencies such that *v* is known to lie in the range [0, *m*).
- Agency 1 generates a random number *R*, chosen uniformly from [0, *m*).

Agency j receives

$$s_{j-1} = (R + \sum_{s=1}^{j-1} v_s) \bmod m,$$

of $v_1, \ldots, v_{j-1}$. Agency j then computes and passes on to Agency $j+1$

$$s_j = (s_{j-1} + v_j) \bmod m = (R + \sum_{s=1}^{j} v_s) \bmod m.$$

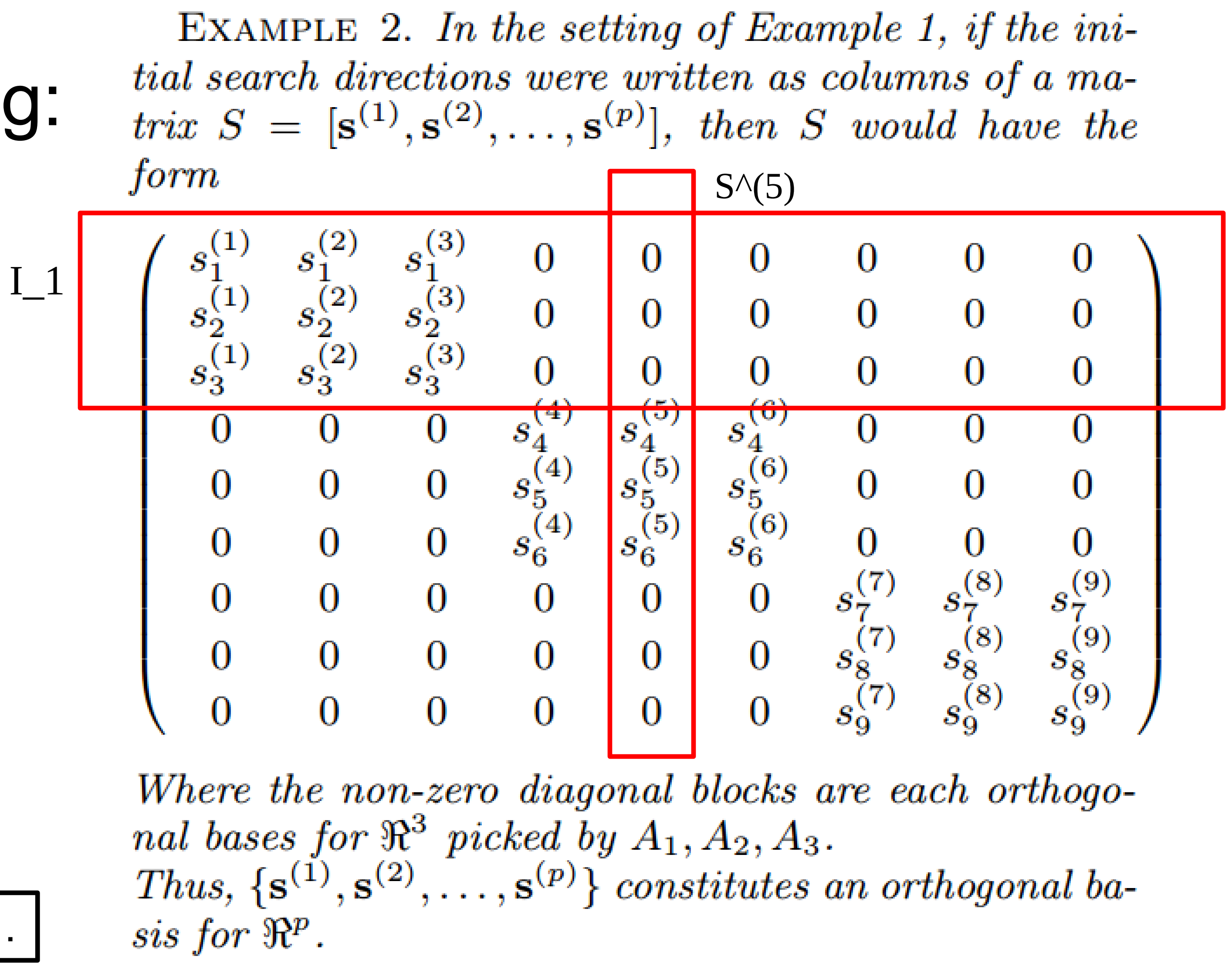
Finally, agency K adds v_K to $s_{K-1}(\text{mod } m)$, and sends the result s_K to agency 1. Agency 1, which knows R then calculates v by subtraction:

$$v = (s_K - R) \bmod m$$

and shares this value with the other agencies. This method for secure summation faces an obvious problem if, contrary to our assumption, some agencies collude. However, there exist collusion-resistant versions of secure summation that involve each agency partitioning their constribution to the sum and all agencies conducting multiple rounds of the summation process to obtain the required sum [4, 9] (the order of the agencies could also be permuted to add an extra layer of security)[2].

Algorithm

- Setting:



Example 2. *In the setting of Example 1, if the initial search directions were written as columns of a matrix* $S = [\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \dots, \mathbf{s}^{(p)}]$, *then* S *would have the form*

$$\begin{array}{c|ccccccccc}
 & & & & & \text{S\^{}(5)} & & & & \\
\text{I\_1} & s_1^{(1)} & s_1^{(2)} & s_1^{(3)} & 0 & 0 & 0 & 0 & 0 & 0 \\
 & s_2^{(1)} & s_2^{(2)} & s_2^{(3)} & 0 & 0 & 0 & 0 & 0 & 0 \\
 & s_3^{(1)} & s_3^{(2)} & s_3^{(3)} & 0 & 0 & 0 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & s_4^{(4)} & s_4^{(5)} & s_4^{(6)} & 0 & 0 & 0 \\
 & 0 & 0 & 0 & s_5^{(4)} & s_5^{(5)} & s_5^{(6)} & 0 & 0 & 0 \\
 & 0 & 0 & 0 & s_6^{(4)} & s_6^{(5)} & s_6^{(6)} & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 0 & 0 & s_7^{(7)} & s_7^{(8)} & s_7^{(9)} \\
 & 0 & 0 & 0 & 0 & 0 & 0 & s_8^{(7)} & s_8^{(8)} & s_8^{(9)} \\
 & 0 & 0 & 0 & 0 & 0 & 0 & s_9^{(7)} & s_9^{(8)} & s_9^{(9)}
\end{array}$$

Where the non-zero diagonal blocks are each orthogonal bases for $\Re^3$ *picked by* A_1, A_2, A_3.
Thus, $\{\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \dots, \mathbf{s}^{(p)}\}$ *constitutes an orthogonal basis for* $\Re^p$.

More details…

Algorithm (con't.)

In our regession case (equation (6)), the δ that minimizes $E(\beta + \delta \mathbf{s}^{(i)}) = (\mathbf{y} - X(\beta + \delta \mathbf{s}^{(i)}))^T (\mathbf{y} - X(\beta + \delta \mathbf{s}^{(i)}))$, is readily obtained as

$$\delta = \frac{(\mathbf{y} - X\beta)^T X \mathbf{s}^{(i)}}{(X\mathbf{s}^{(i)})^T X \mathbf{s}^{(i)}}. \qquad (8)$$

$\mathbf{z} = \mathbf{y} - X\beta = \mathbf{y} - \sum_{j=1}^{k} X_{I_j} \beta_{I_j}$ and $\mathbf{w} = X\mathbf{s}^{(r)} = \sum_{j=1}^{k} X_{I_j} \mathbf{s}_{I_j}^{(r)}$ are computed collectively by $A_1, A_2, \ldots, A_k$

- Less risk with masking
- Each agency contributes to the sum are functionally related from one iteration to the next. (secure summation)

Concluding Remarks

There are some situations that the agencies need to be aware of.

- Our method critically relies on semi-honestness. If an agency is malicious and participates only to sabotage the collective efforts of the others, it can be quite successful by secretly not following protocol.
- The method is susceptible to "unfortunate" data. For instance, it might turn out that $R^2 \approx 1$ and all $\beta_j \approx 0$ for $j \neq 3$; then $\mathbf{x}_3$ is at risk.
- The ownership of certain attributes itself might be a sensitive issue. For instance, a agency that provides investment advice might possess health-related data on their clients that they would like to include in the regression, but would not like to reveal that to other agencies.

2. Secure regression on distributed databases

Karr, Alan F., Xiaodong Lin, Ashish P. Sanil, and Jerome P. Reiter.

Journal of Computational and Graphical Statistics 14, no. 2 (2005): 263-279.

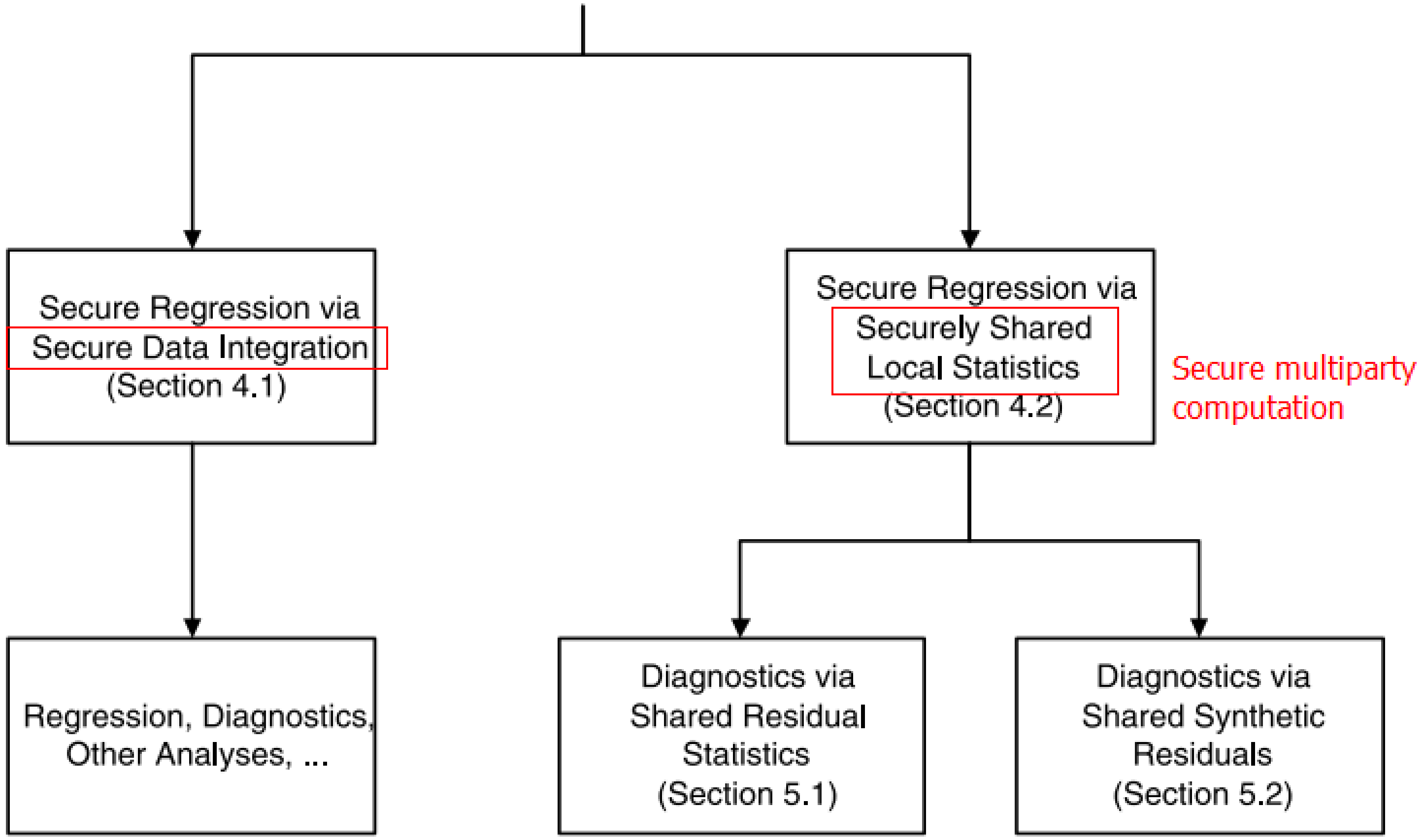
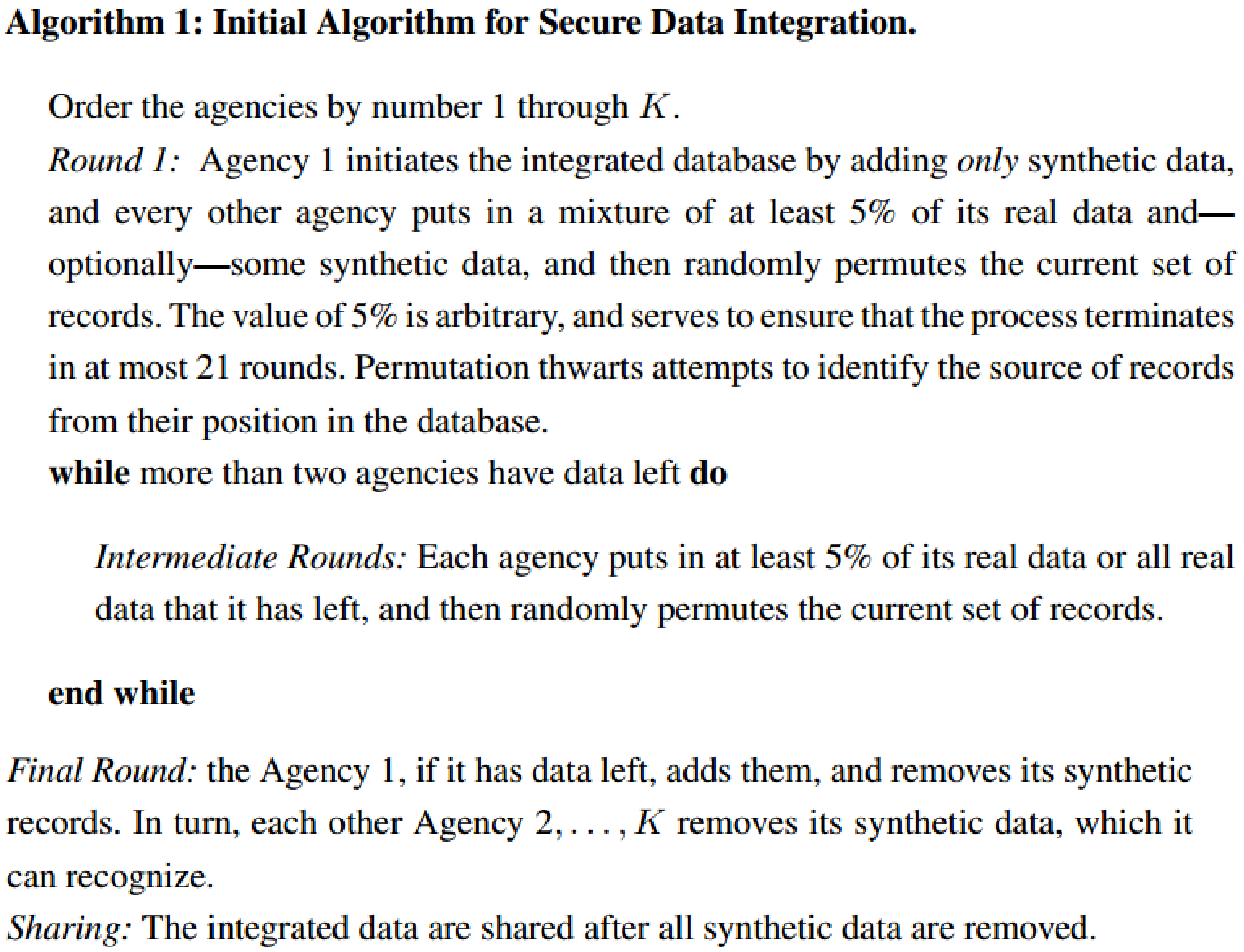

Secure Regression via Secure Data Integration (Section 4.1)
Secure Regression via Securely Shared Local Statistics (Section 4.2)
Secure multiparty computation
Regression, Diagnostics, Other Analyses, ...
Diagnostics via Shared Residual Statistics (Section 5.1)
Diagnostics via Shared Synthetic Residuals (Section 5.2)

Figure 1. Conceptual view of the secure regression problem for multiple, distributed databases. The left-hand branch—secure data integration—is described in Section 3 and Section 4.1. The right-hand branch, which is more secure because it shares only locally computed statistics, is described in Section 4.2, with associated issues of diagnostics discussed in Section 5.

Secure data integration

- *Secure data integration*, which provides the lowest level of protection, actually integrates the databases, but in a manner that no database owner can determine the origin of any records other than its own.

- K > 2;
- In a round-robin order;
- Insert both real and “synthetic” records;
- “synthetic” records may be drawing from predictive distributions fit to the data;
- Once all real data have been included, each agency recognizes and removes its synthetic data.



Algorithm 1: Initial Algorithm for Secure Data Integration.

```
Order the agencies by number 1 through $K$.
Round 1: Agency 1 initiates the integrated database by adding only synthetic data, and every other agency puts in a mixture of at least 5% of its real data and—optionally—some synthetic data, and then randomly permutes the current set of records. The value of 5% is arbitrary, and serves to ensure that the process terminates in at most 21 rounds. Permutation thwarts attempts to identify the source of records from their position in the database.
while more than two agencies have data left do
    Intermediate Rounds: Each agency puts in at least 5% of its real data or all real data that it has left, and then randomly permutes the current set of records.
end while
Final Round: the Agency 1, if it has data left, adds them, and removes its synthetic records. In turn, each other Agency $2, \ldots, K$ removes its synthetic data, which it can recognize.
Sharing: The integrated data are shared after all synthetic data are removed.
```

Problems:

- Identify real records
- Synthetic data are detectable

Solutions:

- Not to retained intermediate quantities (semi-honest)
- Reduce the fraction of the data (5%)
- By randomizing the order in which agencies add data

obviated. In addition to a growing integrated database, Algorithm 2 requires transmission of a binary vector $d = (d_1, \ldots, d_K)$, in which $d_j = 1$ indicates that agency j has not yet contributed all of its data and $d_j = 0$ indicates that it has.

Algorithm 2: Secure data integration with randomized ordering.

```
A randomly chosen agency is designated as the Stage 1 agency $a_1$.
Stage 1: (1) The Stage 1 agency $a_1$ initializes the integrated database with some—there is no option—synthetic data and at least one real data record, and permutes the order of the records. If $a_1$ has exhausted its data, it sets $d_{a_1} = 0$. Then, $a_1$ picks a Stage 2 agency $a_2$ randomly from the set of agencies $j$, other than itself, for which $d_j = 1$, and sends the integrated database and the vector $d$ to $a_2$.
while more than two agencies have data left
    Stages 2, ...: The Stage $\ell$ agency $a_\ell$ adds at least one real data record and, optionally, as many synthetic data records as it wishes to the integrated database, and then permutes the order of the records. If its own data are exhausted, it sets $d_{a_\ell} = 0$. It then selects a Stage $\ell + 1$ agency $a_{\ell+1}$ randomly from the set of agencies $j$, other than itself, for which $d_j = 1$ and sends the integrated database and the vector $d$ to $a_{\ell+1}$.
end while
Last round: Each agency removes its synthetic data.
Sharing: The integrated data are shared after all synthetic data are removed.
```

Discussion…

Secure multiparty computation

- Secure summation.
- Shared local statistics effects

- (old story)

Secure Linear Regression

We assume the usual linear regression model

$$\mathbf{y} = \mathbf{X}\beta + \epsilon, \tag{4.1}$$

where

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & \dots & x_{1p-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \dots & x_{np-1} \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \tag{4.2}$$

and

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_{p-1} \end{bmatrix}, \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix}. \tag{4.3}$$

Under the condition that

$$\text{cov}(\boldsymbol{\varepsilon}) = \sigma^2 I, \tag{4.4}$$

the least squares estimate for $\boldsymbol{\beta}$ is, of course,

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{4.5}$$

Secure Linear Regression (con’t)

- Horizontally partitioned

$$\mathbf{X}^j = \begin{bmatrix} x^j_{11} & \cdots & x^j_{1p} \\ \vdots & \ddots & \vdots \\ x^j_{n_j 1} & \cdots & x^j_{n_j p} \end{bmatrix}, \qquad \mathbf{y}^j = \begin{bmatrix} y^j_1 \\ \vdots \\ y^j_{n_j} \end{bmatrix}.$$

- Solutions:
  - Via secure data integration
    - every agency can perform linear regression using the integrated and shared database.
  - Via securely shared local statistics

Secure Linear Regression (con't)

- Via securely shared local statistics

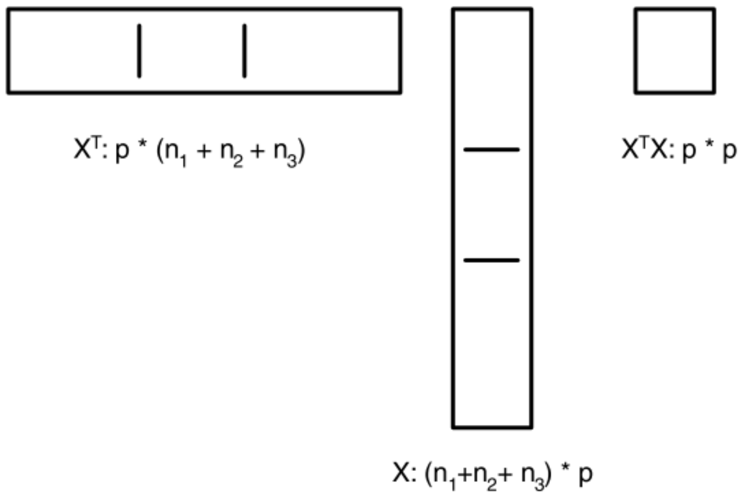
$$\mathbf{X} = \begin{bmatrix} X^1 \\ \vdots \\ X^K \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y^1 \\ \vdots \\ y^K \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_{p-1} \end{bmatrix} \quad \boldsymbol{\epsilon} = \begin{bmatrix} \varepsilon^1 \\ \vdots \\ \varepsilon^K \end{bmatrix}.$$

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

$$\mathbf{X}^T\mathbf{X} = \sum_{j=1}^{K} (\mathbf{X}^j)^T\mathbf{X}^j.$$

$$\mathbf{X}^T\mathbf{y} = \sum_{j=1}^{K} (\mathbf{X}^j)^T\mathbf{y}^j,$$

$\boxed{\text{Others are similar.}}$



Model Diagnostics

- the **coefficient of determination**: R^2

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2},$$

- Model misspecification: not close to zero.
- Via secure summation
  - Shared residual statistics
  - Shared synthetic residuals

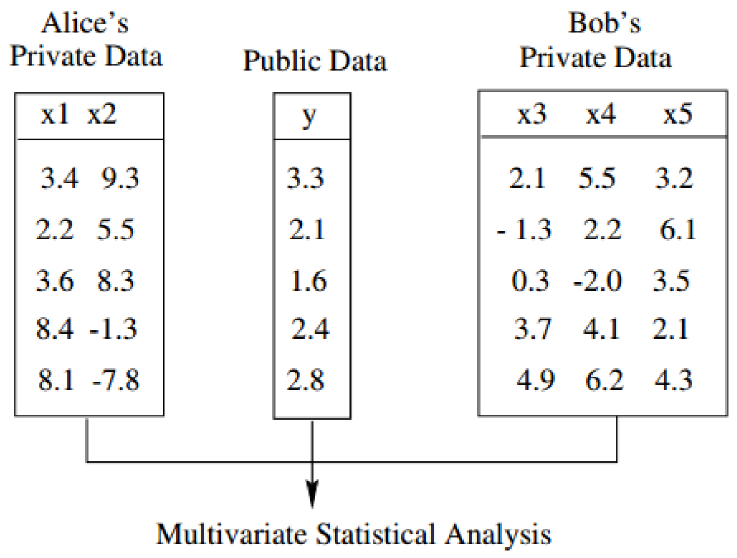
3. Privacy-Preserving Multivariate Statistical Analysis: Linear Regression and Classification

Du, Wenliang, Shigang Chen, and Yunghsiang S. Han.
In Proceedings of the 4th SIAM International Conference on Data Mining. 2004.

Abstract

Multivariate statistical analysis is an important data analysis technique that has found applications in various areas. In this paper, we study some multivariate statistical analysis methods in Secure 2-party Computation (S2C) framework illustrated by the following scenario: two parties, each having a secret data set, want to conduct the statistical analysis on their joint data, but neither party is willing to disclose its private data to the other party or any third party. The current statistical analysis techniques cannot be used directly to support this kind of computation because they require all parties to send the necessary data to a central place. In this paper, We define two Secure 2-party multivariate statistical analysis problems: Secure 2-party Multivariate Linear Regression problem and Secure 2-party Multivariate Classification problem. We have developed a practical security model, based on which we have developed a number of building blocks for solving these two problems.

Notation:



x1	x2
3.4	9.3
2.2	5.5
3.6	8.3
8.4	-1.3
8.1	-7.8

y
3.3
2.1
1.6
2.4
2.8

x3	x4	x5
2.1	5.5	3.2
- 1.3	2.2	6.1
0.3	-2.0	3.5
3.7	4.1	2.1
4.9	6.2	4.3

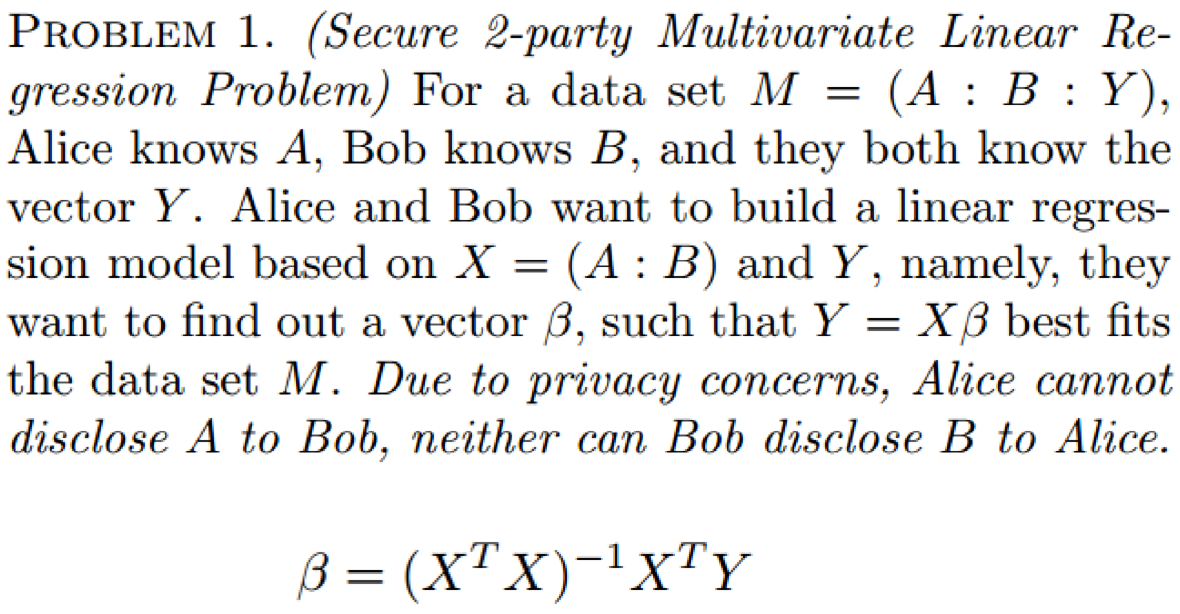
Figure 1: Secure 2-party Multivariate Statistical Analysis

Vertical partition

$$\mathbf{M} = \begin{pmatrix} x_{1,1} & \cdots & x_{1,n} & x_{1,n+1} & \cdots & x_{1,n+m} & y_1 \\ \vdots & & \vdots & \vdots & & \vdots & \vdots \\ x_{N,1} & \cdots & x_{N,n} & x_{N,n+1} & \cdots & x_{N,n+m} & y_N \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ \vdots & \vdots & & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,n} \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} x_{1,n+1} & x_{1,n+2} & \cdots & x_{1,n+m} \\ \vdots & \vdots & & \vdots \\ x_{N,n+1} & x_{N,n+2} & \cdots & x_{N,n+m} \end{pmatrix}$$

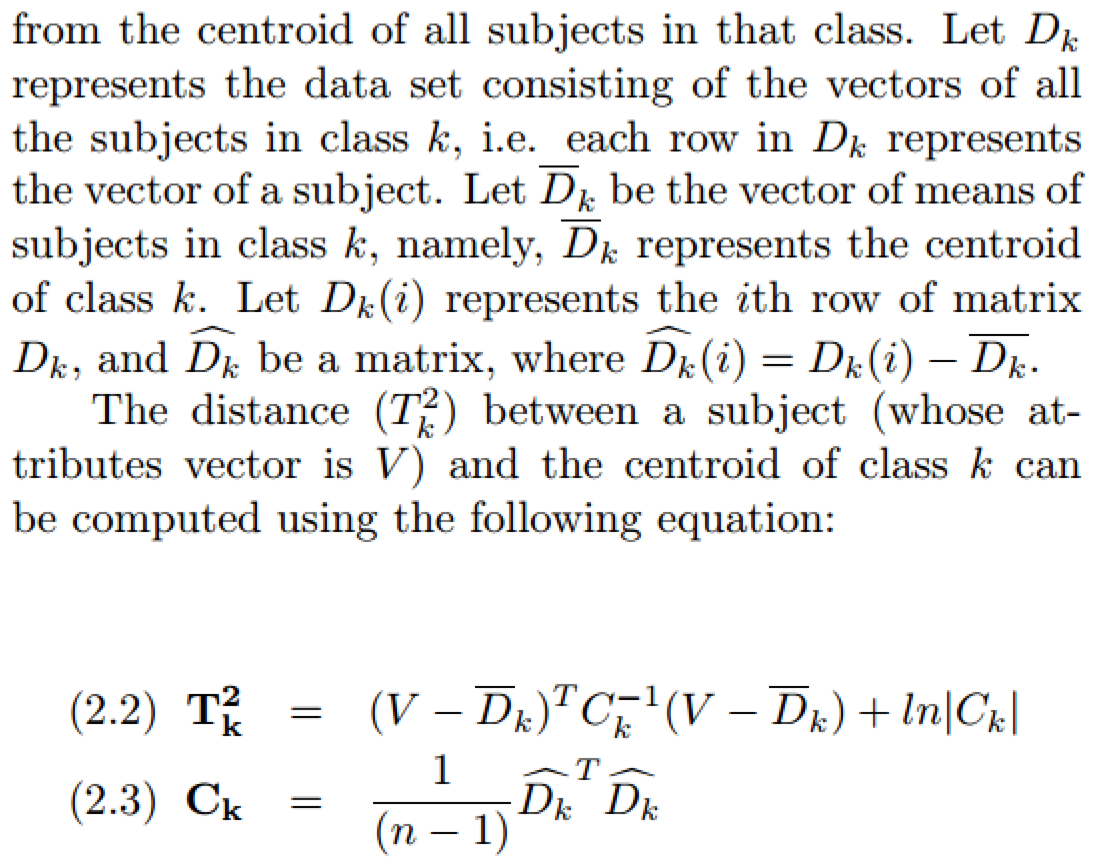
Problem 1: Multivariate Linear Regression



PROBLEM 1. *(Secure 2-party Multivariate Linear Regression Problem)* For a data set $M = (A : B : Y)$, Alice knows A, Bob knows B, and they both know the vector Y. Alice and Bob want to build a linear regression model based on $X = (A : B)$ and Y, namely, they want to find out a vector β, such that $Y = X\beta$ best fits the data set M. *Due to privacy concerns, Alice cannot disclose A to Bob, neither can Bob disclose B to Alice.*

$$\beta = (X^T X)^{-1} X^T Y$$

Problem 2 : classification



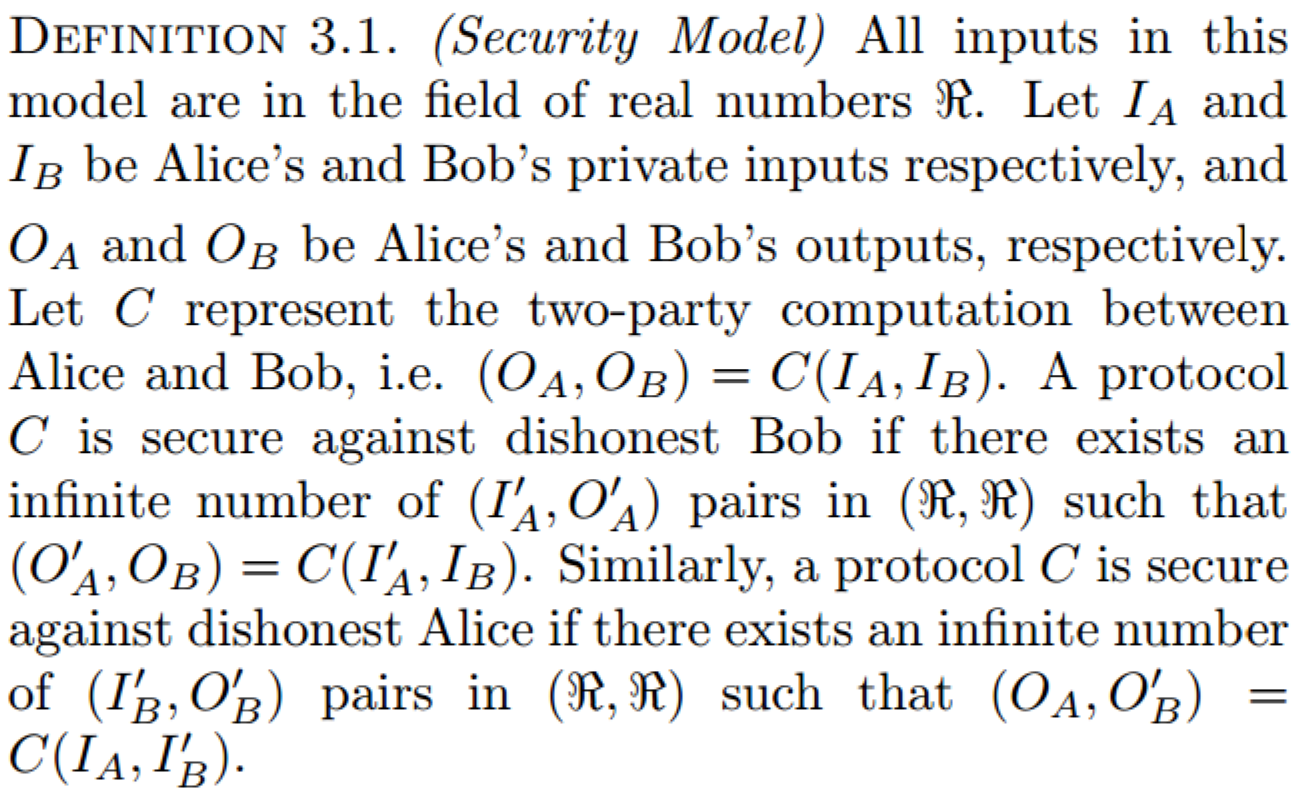
from the centroid of all subjects in that class. Let D_k represents the data set consisting of the vectors of all the subjects in class k, i.e. each row in D_k represents the vector of a subject. Let $\overline{D_k}$ be the vector of means of subjects in class k, namely, $\overline{D_k}$ represents the centroid of class k. Let $D_k(i)$ represents the ith row of matrix D_k, and $\widehat{D_k}$ be a matrix, where $\widehat{D_k}(i) = D_k(i) - \overline{D_k}$.

The distance (T_k^2) between a subject (whose attributes vector is V) and the centroid of class k can be computed using the following equation:

$$
\begin{aligned}
(2.2)\quad \mathbf{T_k^2} &= (V - \overline{D_k})^T C_k^{-1} (V - \overline{D_k}) + ln|C_k| \\
(2.3)\quad \mathbf{C_k} &= \frac{1}{(n-1)} \widehat{D_k}^T \widehat{D_k}
\end{aligned}
$$

A new security model

- balance between efficiency and security

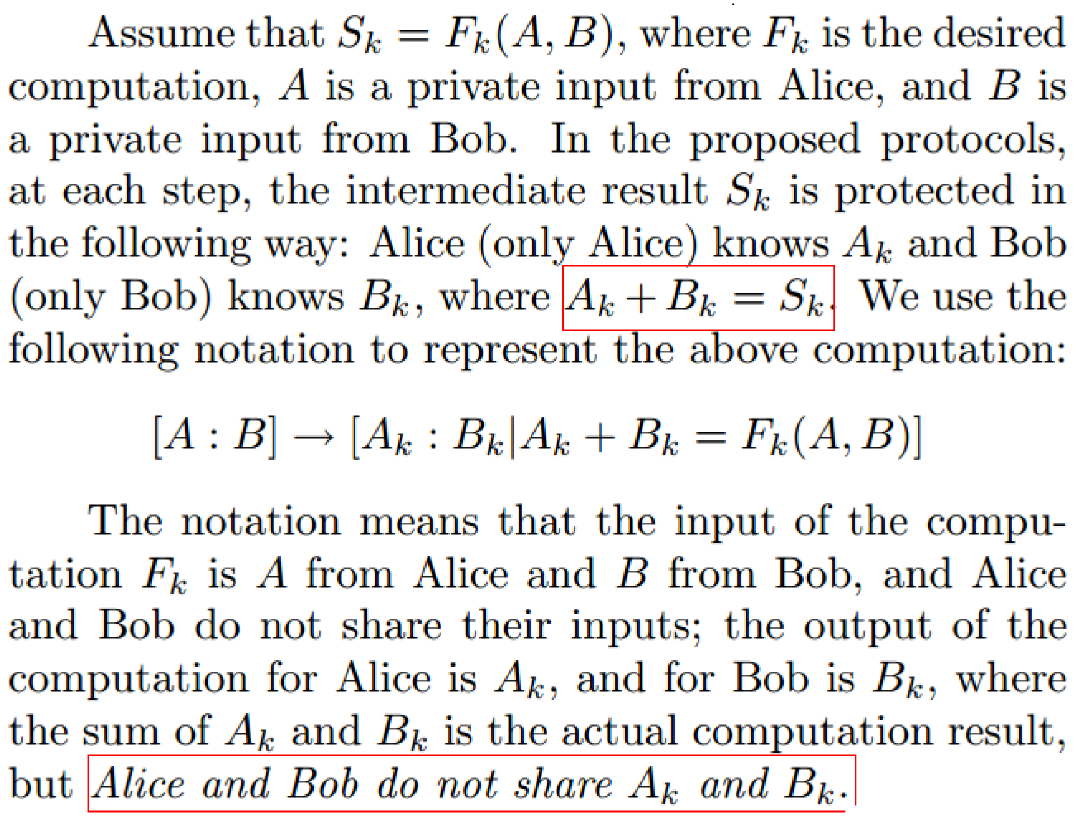


Definition 3.1. *(Security Model)* All inputs in this model are in the field of real numbers $\Re$. Let I_A and I_B be Alice's and Bob's private inputs respectively, and O_A and O_B be Alice's and Bob's outputs, respectively. Let C represent the two-party computation between Alice and Bob, i.e. $(O_A, O_B) = C(I_A, I_B)$. A protocol C is secure against dishonest Bob if there exists an infinite number of (I'_A, O'_A) pairs in $(\Re, \Re)$ such that $(O'_A, O_B) = C(I'_A, I_B)$. Similarly, a protocol C is secure against dishonest Alice if there exists an infinite number of (I'_B, O'_B) pairs in $(\Re, \Re)$ such that $(O_A, O'_B) = C(I_A, I'_B)$.

Computation model

- The two-party model
- The Commodity-Server (CS) model
  - participants accept help from a semi-trusted third party.
  - The third party learns nothing about the private data if it does not collude with any of the two participants.
  - leads to much more efficient solutions.

Data Disguising Methodology



Assume that $S_k = F_k(A, B)$, where F_k is the desired computation, A is a private input from Alice, and B is a private input from Bob. In the proposed protocols, at each step, the intermediate result S_k is protected in the following way: Alice (only Alice) knows A_k and Bob (only Bob) knows B_k, where $\boxed{A_k + B_k = S_k}$. We use the following notation to represent the above computation:

$$[A : B] \rightarrow [A_k : B_k | A_k + B_k = F_k(A, B)]$$

The notation means that the input of the computation F_k is A from Alice and B from Bob, and Alice and Bob do not share their inputs; the output of the computation for Alice is A_k, and for Bob is B_k, where the sum of A_k and B_k is the actual computation result, but *Alice and Bob do not share A_k and B_k.*

- We should not only protect the private inputs, but also protect the intermediate results.
- With one piece being randomly generated.

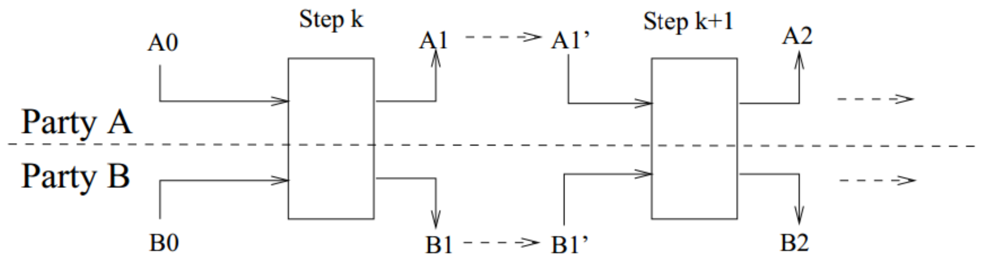


Figure 2: Data Disguising Strategy

Building Blocks

- Matrix Product I:

Alice has an $n \times N$ matrix A and Bob has an $N \times m$ matrix B.

$$[A : B] \rightarrow [V_a : V_b | V_a + V_b = A \cdot B]$$

- How?
  - Commodity-Server solution
  - The two-party solution

Commodity-Server solution

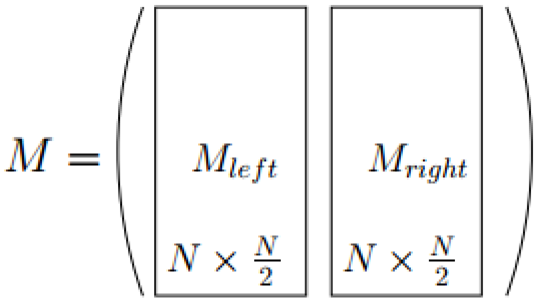
PROTOCOL 1. *($(A \cdot B)$ Protocol – Commodity Server)*

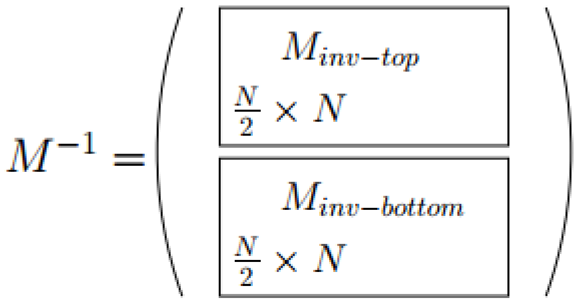
1. The Commodity Server generates a random $n \times N$ matrix R_a and another random $N \times m$ matrix R_b, and lets $r_a + r_b = R_a \cdot R_b$, where r_a (or r_b) is a randomly generated $n \times m$ matrix. Then the server sends (R_a, r_a) to Alice, and (R_b, r_b) to Bob.
2. Alice sends $\widehat{A} = A + R_a$ to Bob, and Bob sends $\widehat{B} = B + R_b$ to Alice.
3. Bob generates a random $n \times m$ matrix V_b, and computes $T = \widehat{A} \cdot B + (r_b - V_b)$, then sends the result T to Alice.
4. Alice computes $V_a = T + r_a - (R_a \cdot \widehat{B})$

It is easy to verify that

$$\begin{aligned}
& V_a + V_b \\
= \; & [(\widehat{A} \cdot B + (r_b - V_b)) + r_a - (R_a \cdot \widehat{B})] + V_b \\
= \; & [A \cdot B - V_b + (r_a + r_b - R_a \cdot R_b)] + V_b \\
= \; & A \cdot B
\end{aligned}$$

Two-Party solution





Protocol 2. *($(A \cdot B)$ Protocol – Two Party)*

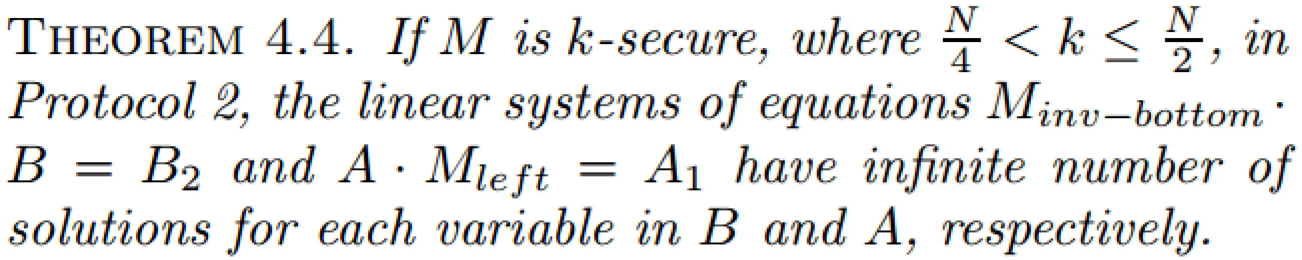
1. Alice and Bob jointly generate a random invertible $N \times N$ matrix M.
2. Alice computes $A_1 = A \cdot M_{left}$, $A_2 = A \cdot M_{right}$. and sends A_1 to Bob.
3. Bob computes $B_1 = M_{inv-top} \cdot B$, $B_2 = M_{inv-bottom} \cdot B$, and sends B_2 to Alice.
4. Alice computes $V_a = A_2 \cdot B_2$.
5. Bob computes $V_b = A_1 \cdot B_1$.

It is easy to see that the above protocol achieves the following:

$$\mathbf{A} \cdot \mathbf{B} = AM \cdot M^{-1}B = (A_1 \;\; A_2)\begin{pmatrix} B_1 \\ B_2 \end{pmatrix} = V_a + V_b$$

Analysis of the Two-Party solution

- How to choice the random matrix M?
  - Bob only knows N/2 equations;
  - Some properties of M:
    - K-secure
    - …



THEOREM 4.4. *If M is k-secure, where $\frac{N}{4} < k \leq \frac{N}{2}$, in Protocol 2, the linear systems of equations $M_{inv-bottom} \cdot B = B_2$ and $A \cdot M_{left} = A_1$ have infinite number of solutions for each variable in B and A, respectively.*

- Input reusing
- The actual range for certain x

Building Blocks (con't)

- Matrix Product II: $(A_1 + B_1)(A_2 + B_2)$

Alice and Bob need to compute $(A_1 + B_1)(A_2 + B_2)$, such that Alice gets V_a and Bob gets V_b, where $V_a + V_b = (A_1 + B_1)(A_2 + A_2)$. This computation can be achieved using the $(A \cdot B)$ Protocol twice because $(A_1 + B_1)(A_2 + B_2) = A_1A_2 + A_1B_2 + B_1A_2 + B_1B_2$. The protocol is represented by the following:

$$\begin{aligned} & [(A_1, A_2) : (B_1, B_2)] \\ \rightarrow \quad & [V_a : V_b | V_a + V_b = (A_1 + B_1) \cdot (A_2 + B_2)] \end{aligned}$$

Building Blocks (con’t)

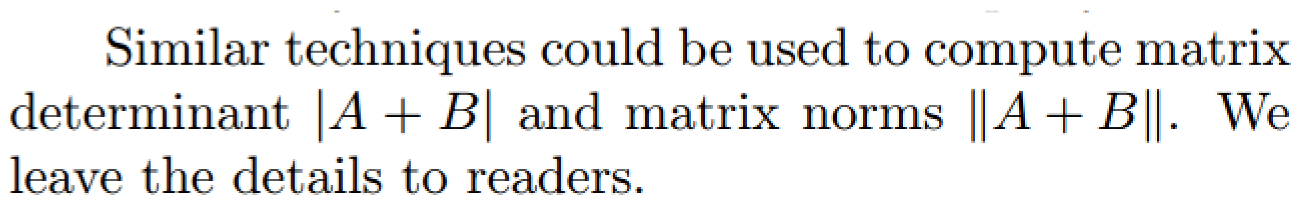
- Matrix Inverse:

The protocol is represented by the following notation:

$$[A : B] \rightarrow [V_a : V_b | V_a + V_b = (A + B)^{-1}]$$

Our solution consists of two major steps: first Alice and Bob jointly convert matrix $(A+B)$ to $P(A+B)Q$ using two random matrices P and Q that are only known to Bob. The results of $P(A+B)Q$ will be known only by Alice who can conduct the inverse computation and gets $Q^{-1}(A+B)^{-1}P^{-1}$. The purpose of P and Q is to prevent Alice from learning matrix B. In the second step, Alice and Bob jointly remove Q^{-1} and P^{-1} and gets $V_a + V_b = (A+B)^{-1}$. Both steps can be achieved using the $(A \cdot B)$ protocol, thus can be solved using both the commodity-server model and the two-party model.

Building Blocks (con’t)



Similar techniques could be used to compute matrix determinant $|A + B|$ and matrix norms $\|A + B\|$. We leave the details to readers.

- $|P(A+B)Q| = |P|*|A+B|*|Q|$
- $\|A+B\|_F^2 = \mathrm{Tr}((A+B)(A+B)^H)$

Privacy-Preserving Multivariate Statistical Analysis

- Multivariate Linear Regression

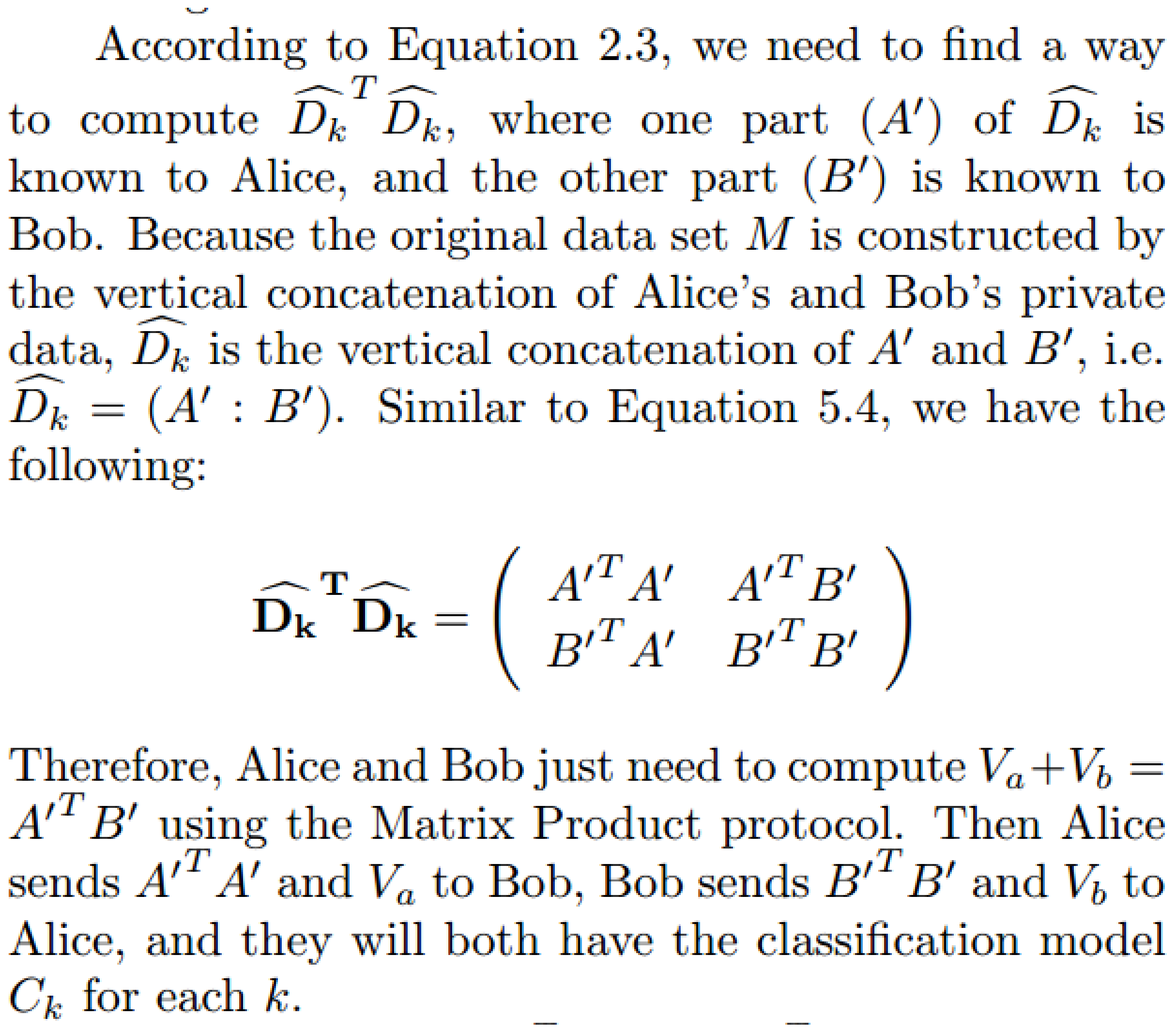
$$\beta = (X^TX)^{-1}(X^TY)$$

$$\mathbf{X^TX} = \begin{pmatrix} A^TA & A^TB \\ B^TA & B^TB \end{pmatrix}$$

1. $V_{a1} + V_{b1} = X^TX$
2. $V_{a2} + V_{b2} = (X^TX)^{-1} = (V_{a1} + V_{b1})^{-1}$
3. $V_{a3} + V_{b3} = X^TY$
4. $\beta = (V_{a2} + V_{b2})(V_{a3} + V_{b3})$.

Step 1 can be achieved using our $(A \cdot B)$ Protocol; step 2 can be achieved using our $(A+B)^{-1}$ Protocol; step 3 can be achieved simply by letting $V_{a3} = A^TY$ and $V_{b3} = B^TY$; finally step 4 can be achieved using our Matrix Product II protocol.

- Multivariate Classification



According to Equation 2.3, we need to find a way to compute $\widehat{D_k}^T \widehat{D_k}$, where one part ($A'$) of $\widehat{D_k}$ is known to Alice, and the other part (B') is known to Bob. Because the original data set M is constructed by the vertical concatenation of Alice's and Bob's private data, $\widehat{D_k}$ is the vertical concatenation of A' and B', i.e. $\widehat{D_k} = (A' : B')$. Similar to Equation 5.4, we have the following:

$$\widehat{\mathbf{D_k}}^{\mathbf{T}} \widehat{\mathbf{D_k}} = \begin{pmatrix} A'^T A' & A'^T B' \\ B'^T A' & B'^T B' \end{pmatrix}$$

Therefore, Alice and Bob just need to compute $V_a + V_b = A'^T B'$ using the Matrix Product protocol. Then Alice sends $A'^T A'$ and V_a to Bob, Bob sends $B'^T B'$ and V_b to Alice, and they will both have the classification model C_k for each k.

$\boxed{\text{Problem.}}$