

Privacy-Preserving Multivariate Statistical Analysis: Linear Regression and Classification *

Wenliang Du

Department of Electrical Engineering
& Computer Science
Syracuse University, 121 Link Hall
Syracuse, NY 13244
Email: wedu@ecs.syr.edu

Yunghsiang S. Han

Department of Computer Science
& Information Engineering
National Chi Nan University
Taiwan 545
Email: yshan@csie.ncnu.edu.tw

Shigang Chen

Department of Computer & Information
Science & Engineering
University of Florida
Gainesville, FL 31611
Email: sgchen@cise.ufl.edu

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.

Keywords: Privacy, security, multivariate statistical analysis, secure multi-party computation.

*This work was supported in part by Grant IIS-0219560 and IIS-0312366 from the United States National Science Foundation, by the SUPRIA program of the CASE Center at Syracuse University, and by the National Science Council of Taiwan, R.O.C., under grants NSC 92-2213-E-260-007.

1 Introduction

Multivariate statistical analysis is an important data analysis technique that has found applications in various areas, such as business, education, and defense. In this paper, we study the multivariate statistical analysis in a *Secure 2-party Computing (S2C)* environment, where data are observed by two parties who are not willing to fully share their private observations with others; however, both parties do want to take advantage of the collaboration, and they do want to benefit from a joint computation on their joint data. For example, they might want to derive a prediction model based on the joint data set, or they might want to know whether an attribute observed by Alice is related to an attribute observed by Bob, etc. We call this type of data analysis problem under S2C environment the *Secure 2-party Multivariate Statistical Analysis (S2-MSA)* problem. Figure 1 depicts the S2-MSA problem. In this figure, Alice observes attributes x_1 and x_2 , while Bob observes attributes x_3 , x_4 , and x_5 ; they both can observe the y attribute. The task of S2-MSA is to conduct multivariate statistical analysis, without requiring each party's data to be disclosed to the other.

A common strategy to solve the S2-MSA problem is to assume the trustworthiness of the participants, or to assume the existence of a *trusted* third party. In today's environment, making such assumptions can be difficult and infeasible. Moreover, in certain situations, even though we could trust that the other parties will

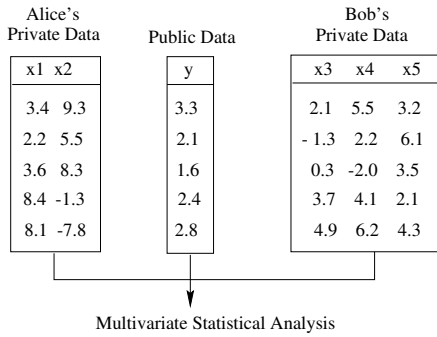


Figure 1: Secure 2-party Multivariate Statistical Analysis

not abuse our private information, we cannot guarantee that their computer systems and network are secure enough to prevent our information from being stolen. Alternatively, from the trusted party's point of view, in order to conduct such a cooperative computation, they have to carry the extra burden of securing other party's data. A security breach that compromises the data may result in serious ramifications. Therefore, it is desirable if nobody knows the other parties' secret information. Techniques that can support this type of joint computation while protecting the participants' privacy are of growing importance.

There are many multivariate data analysis techniques, such as regression, classification, factor analysis, T^2 test, etc. In this paper, we focus on two techniques: multivariate linear regression and classification. Multivariate linear regression concerns about determining a linear function that best fits a set of data observations. Multivariate classification concerns about building a classification model for predicting membership of objects from their measurements on one or more predictor variables.

This paper defines two S2-MSA problems: *Secure 2-party Multivariate Linear Regression (S2-MLR)* problem, and *Secure 2-party Multivariate Classification (S2-MC)* problem. Because MLR and MC problems are built upon matrix computations (multiplication, inverse, etc.), we have developed a set of basic protocols for secure 2-party matrix computations; then we develop our solutions to the S2-MLR and S2-MC problems based on these basic protocols. It should be noted that the building blocks and the methodologies proposed in this paper can be used to solve other privacy-preserving problems beyond the Multivariate Classification and the Linear Regression problems.

2 Problem Definition and Background

2.1 Notations Let M be a data set represented by the following $N \times (n + m) + 1$ matrix:

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

Each column of the data set represents an attribute, so there are $(n + m) + 1$ attributes in this data set, X_1, \dots, X_{n+m} and Y . Each row of M represents a subject's values of these attributes, and there are N subjects in this data set. We define X as an $N \times (n+m)$ matrix formed by attributes X_1, \dots, X_{n+m} ; we define Y as a vector formed by attribute Y .

In the problems studied in this paper, the data set X is vertically divided into two parts and are distributed to two parties. We call this partition the *vertical partition*, and these two parties Alice and Bob. Alice has the subset A and Bob has the subset B , where A and B are defined as the following:

$$A = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ \vdots & \vdots & & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,n} \end{pmatrix}$$

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

In other words, Alice is able to observe the values of attributes X_1, \dots, X_n for these N subjects, while Bob can observe the values of attributes X_{n+1}, \dots, X_{n+m} for the same set of the subjects. In addition, both Alice and Bob can observe the values of the Y attribute¹. We use the notation $(A : B)$ to represent the union of these two data sets, i.e., $X = (A : B)$; we also use $M = (A : B : Y)$ to represent the whole data set.

The goal of this paper is to find ways to conduct multivariate statistical analysis under the circumstance described above, without any party disclosing his/her private data (A or B) to the other.

There is another way for the data set M to be distributed: M can be horizontally divided into two parts, with Alice knowing all the attributes values of a subset of the subjects, and Bob knowing all the attributes values of the other subset of the subjects. We call this partition the *horizontal partition*. Conducting statistical analysis on the horizontally partitioned data is much easier than that on the vertically partitioned

¹Our solutions can be easily extended to the situations where Y is observed by only one party.

data because there is no need for Alice and Bob to disclose their parts of data to the other party: all they need to exchange is the aggregate information, not the raw data. We will skip the details for the horizontal-partition situation in this paper.

2.2 Multivariate Linear Regression The goal of linear regression is to determine the values of parameters for a linear function that cause the function to best fit a set of data observations. For the data set $M = (A : B : Y)$, to find out how the value of the dependent variable Y depends on the value of independent variables X_1, \dots, X_{n+m} , we need to find the parameters β , such that $Y = X\beta$ best fits the data set M .

Once the values of the parameters are determined, one can use the formula to predict the value of Y for a new subject whose values of X_1, \dots, X_{n+m} are provided. For example, after we build a linear regression model based on a known data set, we can predict the number of credit cards used by a customer based on his/her family size, family income, and number of automobiles owned.

If a perfect fit existed between the function and the actual data, the actual value of Y in the data set M would exactly equal to the predicted value. Typically, however, this is not the case, and the difference between the actual value of the dependent variable Y and its predicted value for a particular observation is the error of the estimate, which is known as the “residual”. The goal of regression analysis is to determine the values of the parameters that minimize the sum of the squared residual values for the set of observations. This is known as a “least squares” regression fit. Based on the least square method, β can be calculated using the following equation:

$$(2.1) \quad \beta = (X^T X)^{-1} X^T Y$$

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.*

2.3 Classification Unlike in the linear regression analysis situation where the value of Y for each subject is a continuous value, in many other situations, the value of Y is a discrete value, representing a category that a subject belongs to. If we want to build a model and use it to predict what category a new subject should belong to, we cannot use the linear regression analysis, instead,

we use the classification method.

The purpose of classification is to build a model which can be used for predicting the class label (the category) for a subject based on its values of a set of attributes. Classification can be conducted using various methods, such as statistical methods, decision trees, and neural networks [17]. In this paper we focus on a statistical analysis method and we only give a brief overview of this method. The details of the method can be found in [24].

One can say that a subject belongs to a class k , if the attributes vector of that subject is closest to the centroid of class k (i.e. the vector of means of subjects in class k). The distance (it will be defined later) for a subject in a class indicates how far the subject is 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 i th 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:

$$(2.2) \quad \mathbf{T}_k^2 = (V - \overline{D}_k)^T C_k^{-1} (V - \overline{D}_k) + \ln|C_k|$$

$$(2.3) \quad \mathbf{C}_k = \frac{1}{(n-1)} \widehat{D}_k^T \widehat{D}_k$$

A large distance indicates that an observation is an outlier for the predictor. Assume there are c different categories (class labels) in Y . After computing T_1^2, \dots, T_c^2 , the smallest value is selected. For instance, if T_j^2 is the smallest value, one can decide that the new subject belongs to category j . Therefore, in order to make predictions, the goal of classification is to compute C_k (the covariance matrix) and \overline{D}_k .

PROBLEM 2. (*Secure 2-party Classification 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 wants to build a classification model based on $X = (A : B)$ and Y such that, given a new subject whose Y attribute (category) is unknown, the model can predict which category the subject belongs to based on the subject’s X attributes. At the end of the computation, both Alice and Bob should know the classification model, i.e. C_k and \overline{D}_k for $k = 1, \dots, c$. However, *due to privacy concerns, Alice cannot disclose A to Bob, neither can Bob disclose B to Alice.*

3 Security Model and Methodologies

To compute equation (2.1), (2.2) and (2.3) under the secure two-party framework, we need to know how to conduct basic matrix computations on the private data from two different parties. We present our solutions to various basic matrix computations including matrix product, matrix inverse, and matrix determinant. These solutions serve as the building blocks for solving our secure two-party multivariate statistical analysis problems. Before we discuss the building blocks, we first introduce our security model, computation model, and methodology.

3.1 Security Model Studies in Secure Multi-party Computation (SMC) have formalized privacy in the Secure Two-party Computation [15]. According to such formalization, whatever computed by a party participating in the protocol can only be computed based on the party's input and output. Since the parties have the access rights to their input and output, by the problem definition, no additional information is learned.

The security property of the above SMC security model is very desirable because it does not disclose extra information; however, this desirable property is difficult to achieve efficiently. As the above literature has shown, the results of SMC are limited only to a small set of computation problems. Very few work has been done to extend the SMC concepts to more complicated computations, such as data mining, statistical analysis, and scientific computations, all of which require a large amount of complicated computations. Although some work has extended SMC to more complicated computation problems, such as data mining problem in [18, 25], most of them are still not efficient enough for practical use.

In order to achieve a balance between efficiency and security, we propose a new security model. The proposed security model lowers the requirements on security in exchange for better performance. Our tradeoff is that a dishonest party might be able to learn some information about the other party's private data, but it is still impossible for the dishonest party to derive the raw data of the other party.

Since the computations we are studying, such as statistical analysis, are employed in the field of real numbers, the proposed security model is defined on the field of real numbers other than on a finite field (all the known secure multi-party computation protocols are defined in a finite field).

DEFINITION 3.1. (Security Model) All inputs in this model are in the field of real numbers \mathfrak{R} . 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 $(\mathfrak{R}, \mathfrak{R})$ 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 $(\mathfrak{R}, \mathfrak{R})$ such that $(O_A, O'_B) = C(I_A, I'_B)$.

Intuitively speaking, a protocol is secure if, for any input/output pair (I, O) from one party, there exists an infinite number of possible inputs in \mathfrak{R} from the other party such that the result of the protocol is still O from the first party's point of view when given its own input I . Therefore, from its own observed output, a dishonest party cannot determine the inputs from the other party.

Comparing to the SMC security model, our security model is weaker in security. Theoretically, a protocol that satisfies this security model might still disclose significant information. This happens when all the possible values I'_A of input I_A are close to I_A , thus although the exact information about I_A is not disclosed, enough information about it is compromised. However, how likely can this situation occur and under what situation can it occur are yet to be determined. Before the theories are developed for this model, we only consider the model as a heuristic model, and we believe it is of great importance to study it because the model can lead to solutions that are much more efficient than the solutions based on the secure multi-party computation model. Theoretical analysis of this model is still our ongoing investigation. This paper focuses on how practical solutions for various multivariate statistical analysis problems can be developed.

3.2 Computation Model In this paper, we discuss the Secure 2-party Multivariate Statistical Analysis problems under two computation models: the *two-party* model and the *Commodity-Server* (CS) model. The two-party model involves just two parties, and it does not need any help from a third party. In a 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. Theoretically, the two-party model is better than the CS model in security; however, in practice, the CS model could also be appealing because it usually leads to much more efficient solutions.

The commodity server is called a semi-trusted third party because of the following reasons: (1) it cannot derive the private information of the data from Alice or Bob; it should not know the computation result either. (2) It should not collude with either party. (3) It follows

the protocol correctly. In real world, finding such a semi-trusted third party is much easier than finding a trusted third party.

As we will see from our solutions, the commodity server does not participate in the actual computation between Alice and Bob; it only supplies commodities that are independent of the private data, and these commodities can help Alice and Bob to achieve computation security. Therefore, the server can generate independent data off-line, and sell them as commodities to Alice and Bob (hence the name “commodity server”).

The commodity server model was first proposed by Beaver [4, 5], and has been used for solving Private Information Retrieval problems [4, 5, 8] and various secure 2-party computation problems [9].

3.3 Data Disguising Methodology For many non-trivial computations, achieving security usually takes more than one step. We inevitably face one problem: who should keep the intermediate results, the results produced after each step? In many applications, nobody should know the intermediate results. For example, if the intermediate result is $a \times b$ (a and b are two numbers), where a belongs to Alice and b belongs to Bob; whoever knows $a \times b$ can find out the other party’s private input. Therefore, we should not only protect the private inputs, but also protect the intermediate results.

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 $A_k + B_k = S_k$. 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 .*

When the computation contains multiple steps, we use the scheme depicted in Figure 2 to conduct each step. As long as we can cut each intermediate result into two pieces, with one piece being randomly generated, nobody can find out the intermediate results.

4 Building Blocks

4.1 Matrix Product I: $A \cdot B$ We now describe several building blocks that are used in our solutions. We start from the matrix product protocol. In this protocol, Alice has an $n \times N$ matrix A and Bob has

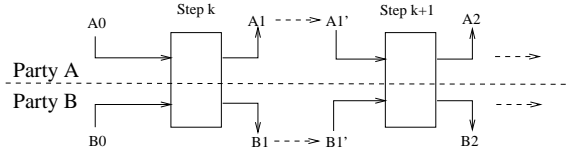


Figure 2: Data Disguising Strategy

an $N \times m$ matrix B . They want to conduct the product computation, such that Alice gets V_a and Bob gets V_b , where $V_a + V_b = A \cdot B$, namely, the product of A and B is divided into two secret pieces, with one piece going to Alice and the other piece going to Bob. We use the following notation to represent this computation.

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

It should be noted that if $A \cdot B$ is the intermediate result, to maximize information disguise, Alice (resp. Bob) should not disclose V_a (resp. V_b) to the other party. However, in some problems (e.g. multivariate classification), $A \cdot B$ is the final result and needs to be known by both parties. In these situations, regardless of what solutions are used, it is impossible to achieve security if both A and B are invertible matrices, because knowing $A \cdot B$ and one of the inputs (A or B) allows one to derive the other input. However, in our Secure 2-party multivariate classification problem, $N \gg n$ and $N \gg m$; therefore $A \cdot B$ (a $n \times m$ matrix) is much smaller than A and B , so knowing $A \cdot B$ and one of the inputs does not allow one to derive the significant information about the other input.

This matrix product protocol will be used as the building block for our other protocols; therefore we provide two solutions: one uses the commodity server model, and the other uses the two-party model.

4.1.1 Commodity-Server Solution To achieve security, we let Alice send $\hat{A} = A + R_a$ to Bob, and let Bob send $\hat{B} = B + R_b$ to Alice, where R_a and R_b are random matrices. Therefore Bob can compute $\hat{A} \cdot B = A \cdot B + R_a \cdot B$, and Alice can compute $R_a \cdot \hat{B} = R_a \cdot B + R_a \cdot R_b$. Combining $\hat{A} \cdot B$ and $R_a \cdot \hat{B}$ can give us $A \cdot B - R_a \cdot R_b$. The challenge is how to get rid of $R_a \cdot R_b$; the Commodity Server can help us achieve that.

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

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

THEOREM 4.1. *Protocol 1 does not allow Alice to learn B , it does not allow Bob to learn A either.*

Proof. See Appendix A.

It should also be noted that our protocol does not deal with the situation where one party lies about its input. For example, instead of sending $B + R_b$, Bob sends $B' + R_b$, where B' is an arbitrary matrix. In that case, neither of them can get correct results, but still, neither of them can gain information about the other party's private input.

If $N \gg n$, which is the situation we face in solving S2-MSA problems (note n is the number of attributes, and N is the number of subjects), the dominating communication cost of this protocol is caused by sending R_a , R_b , \widehat{A} , and \widehat{B} , which is $4nN$ (we assume that $m = n$ for the simplicity of the analysis). This cost is 4 times the *optimal* cost of a two-party matrix product (the optimal cost of a matrix product is defined as the cost of conducting the product of A and B without the privacy constraints, namely one party just sends its data in plaintext to the other party). The cost can be further improved to $2nN$ because matrices R_a and R_b are randomly generated by the commodity server, and only the seeds (numbers of constant size) for a common random generator are needed by Alice and Bob to compute them.

4.2 Two-Party Solution The matrix product can also be achieved using a two-party protocol without using any third party. The idea is to transform A (resp. B) to another matrix A' (resp. B') such that disclosing part of A' to Bob does not allow Bob to derive the raw data of A , and disclosing part of B' to Alice does not allow Alice to derive the raw data of B . A linear transformation would achieve this goal. We select an invertible $N \times N$ matrix M , and let $A' = AM$; disclosing half of the data of A' does not allow anyone to derive

all raw data of A . Based on this observation, we derive our protocol (for the sake of simplicity, we assume N is even; this can be achieved by padding an extra row or column in the original matrices when N is odd):

We vertically divide the $N \times N$ matrix M to two equal-sized sub-matrices M_{left} and M_{right} with size $N \times \frac{N}{2}$; we horizontally divide M^{-1} to two equal-sized sub-matrices $M_{inv-top}$ and $M_{inv-bottom}$ with size $\frac{N}{2} \times N$. The notations are depicted in Figure 3.

$$M = \left(\begin{array}{c|c} M_{left} & M_{right} \\ \hline N \times \frac{N}{2} & N \times \frac{N}{2} \end{array} \right) \quad M^{-1} = \left(\begin{array}{c} M_{inv-top} \\ \hline M_{inv-bottom} \\ \hline \frac{N}{2} \times N \\ \hline \frac{N}{2} \times N \end{array} \right)$$

Figure 3: M and M^{-1}

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:

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

4.3 Analysis of the Two-Party Solution To analyze how secure Protocol 2 is, we need to find out how much Alice and Bob know about each other's information. Since the protocol is symmetric, we just discuss how much information Alice disclosed to Bob. Let vector X represent a row of matrix A . We now discuss how much information about X is disclosed to Bob. In this protocol, Bob knows A_1 , which contains $\frac{N}{2}$ data items of the vector XM . Let us consider $X = (x_1, \dots, x_N)$ as N unknown variables, and XM as a linear system of equations on these N unknown variables. If Bob knows all of the N equations, Bob can easily solve this linear system, and recover all the values in X . However, in this protocol, Bob only knows $\frac{N}{2}$ equations. Theoretically, if

x_i 's are in \mathfrak{R} , based on these $\frac{N}{2}$ equations, there are infinite number of solutions to these N unknown variables. Therefore, although Bob learns $\frac{N}{2}$ linear combinations of the N data items, it is impossible for Bob to learn the actual values for all items in vector X .

However, the above discussion does not exclude the situation when Bob can learn the actual values for some data items in vector X . For instance, if we choose $M = I$, Bob now learns half of the values in vector $XM = X$. This is a significant information disclosure. Therefore, a good selection of M is very important. In the following, we analyze the properties of matrix M (whose size is $N \times N$).

Let M_k be a sub-matrix formed by removing any k rows from M_{left} , and let M_{inv-k} be a sub-matrix formed by removing any k columns from $M_{inv-bottom}$. In the following discussion we assume that the ranks of M_{left} and $M_{inv-bottom}$ are $\frac{N}{2}$ otherwise M is not invertible. For the simplicity of presentation, we only analyze the properties of $M_{inv-bottom}$, and the properties of M_{left} are similar. We let $\hat{M} = M_{inv-bottom}$, i.e., \hat{M} is an $\frac{N}{2} \times N$ matrix. We introduce the following definition.

DEFINITION 4.1. (*k-secure*) Let \hat{M}_k be a sub-matrix formed by removing any k columns from the matrix \hat{M} . \hat{M} is *k-secure* if the rank of \hat{M}_k is $\frac{N}{2}$ for any \hat{M}_k .

THEOREM 4.2. If \hat{M} is *k-secure*, any nonzero linear combination of the the row vectors of \hat{M} generates a row vector with at least $k + 1$ non-zero entries.

Proof. Let row vector v be a nonzero linear combination of the row vectors of \hat{M} , i.e., $v = p\hat{M}$, where p is row vector of size $\frac{N}{2}$. Assume that v has at most k non-zero entries. We use e_1, \dots, e_k to represent the position of these entries.

Remove the columns e_1, \dots, e_k from \hat{M} and we get \hat{M}_k . Therefore, $p\hat{M}_k = 0$, i.e., the same linear combination p on the new sub-matrix generates a zero vector. This means, after removing k columns from \hat{M} , the rank of resultant matrix \hat{M}_k is less than $\frac{N}{2}$. This contradicts to the fact that \hat{M} is *k-secure*.

The above theorem guarantees that, regardless of how the equations $\hat{M}x = b$ are linearly combined (except the trivial zero combination), it is impossible to generate an equation that contains less than $k + 1$ variables. This means that each single equation disclosed by Alice or Bob involves at least $k + 1$ variables. In other words, each unknown data item is disguised by at least k other unknown data items.

Based on our above definition, the identity matrix I is a bad choice for M because I is not even 1-secure:

removing any row from I_{left} results in a matrix whose rank is less than $\frac{N}{2}$.

Only requiring that each equation contains $k + 1$ variables is still not sufficient to prevent information disclosure, because these $k + 1$ variables might still be solvable. For example, if there exists $k + 1$ linearly independent equations that involves the same $k + 1$ variables, the linear system of equations formed by these $k + 1$ equations has a unique solution and can be solved. We need to guarantee that the situation like this cannot happen.

THEOREM 4.3. Let Λ be a $k \times N$ matrix, where each row of Λ is a nonzero linear combination of row vectors in \hat{M} . If \hat{M} is *k-secure*, the linear system of equations $\Lambda x = b$ involves at least $2k$ variables.

Proof. Using a proper Gaussian elimination on the linear system of equations $\Lambda x = b$, we can get a new linear system of equations $(I : \Lambda')x = b'$, where I is the $k \times k$ identity matrix, Λ' is an $k \times (N - k)$ matrix, and $(I : \Lambda')$ means the vertical concatenation of I and Λ' .

According to Theorem 4.2, each row of $(I : \Lambda')$ contains at least $k + 1$ non-zero entries, i.e., each row of Λ' contains at least k non-zero entries. Therefore the linear system of equations $(I : \Lambda')x = b'$ involves at least $k + k = 2k$ variables, with k variables being contributed by I , and at least k other variables being contributed by Λ' .

In summary, *k-secure* guarantees that any equation contains at least $k + 1$ variables and any k combined equations contain at least $2k$ variables; however, with these properties, *k-secure* is still not secure enough because it does not rule out the situation when $2k$ equations can contain just the same $2k$ variables (thus can be solvable). To make it impossible to find $2k$ equations, we should let $\frac{N}{4} < k \leq \frac{N}{2}$ because there are only $\frac{N}{2}$ equations in \hat{M} .

We say that an $N \times N$ matrix M is *k-secure* if both M_{left} and $M_{inv-bottom}$ are *k-secure*. Based on the above discussion, we can derive the following theorem:

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.

To achieve the best security, we should choose a matrix that is $\frac{N}{2}$ -secure. We have developed an efficient algorithm to construct such a matrix M that is $\frac{N}{2}$ -secure by using a class of *maximum distance separable codes*, and we have proved that such M satisfies $\frac{N}{2}$ -secure property. The details are described in Appendix B.

In addition to the selection of M , we should also be careful about the input reusing. For each row vector X in A , A_1 in Protocol 2 discloses $\frac{N}{2}$ equations about X . If A is reused in another protocol between Alice and Bob, another $\frac{N}{2}$ equations about X could be disclosed to Bob. If these $N = \frac{N}{2} + \frac{N}{2}$ equations are linear independent, Bob can solve X . Therefore, when A is reused in another protocol, make sure to use the same M and disclose the same part of the data.

It should be noted that the k -secure property of \hat{M} only guarantees that there exists infinite number of solutions in the real domain for each variable for the linear system of equation $\hat{M}x = b$, where \hat{M} is a matrix of size $\frac{N}{2} \times N$. However, since the domain of each variable x is only a subset of the real domain, based on $\hat{M}x = b$, an adversary might be able to further derive the actual range for certain x . The smaller such a range is, the less privacy is preserved, even though the adversary cannot find the exact value of x . Therefore, to fully understand the privacy-preserving property of our scheme, it is important to analyze the actual range of each variable. We plan to conduct such an analysis in our future work.

The communication cost of Protocol 2 is nN (assuming $n \approx m$), which is the optimal cost of the two-party (non-secure) matrix product computation. However, if N is large and $N \gg n$, as it is usually the case, the multiplication computation cost ($O(\frac{N^2}{2})$) is expensive due to the computation of finding M (M^{-1} can be constructed by the method give in Appendix B which only involves addition). However, since M does not depend on the inputs from Alice or Bob, M can be pre-computed off-line and can also be reused.

4.4 Matrix Product II: $(A_1 + B_1)(A_2 + B_2)$ In this protocol, Alice has matrices A_1 and A_2 , Bob has matrices B_1 and B_2 ; A_1 and B_1 are $n_1 \times t$ matrices while A_2 and B_2 are $t \times n_2$ matrices.

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 $V_a + V_b = (A_1 + B_1)(A_2 + B_2)$. 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:

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

4.5 Matrix Inverse: $(A + B)^{-1}$ In this protocol, Alice has A , Bob has B ; both A and B are $n \times n$ matrices, and $A+B$ is invertible. Alice and Bob need to compute $(A+B)^{-1}$, such that Alice (only Alice) gets V_a and Bob (only Bob) gets V_b , where $V_a + V_b = (A+B)^{-1}$.

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.

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

5 Privacy-Preserving Multivariate Statistical Analysis

5.1 Multivariate Linear Regression With the building blocks described above, we can now solve S2-MLR and S2-MC problems. In S2-MLR problem, Alice has a data set A for N subjects, Bob has another data set B for the same subjects, and $X = (A : B)$ is the concatenation of A and B . Alice and Bob both know the multivariate relationship (denoted by Y) between their data sets, but Alice does not know B and Bob does not know A . The goal is to find β from Equation 2.1:

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

where $X^T X$ can be represented by the following form:

$$(5.4) \quad \mathbf{X}^T \mathbf{X} = \begin{pmatrix} A^T A & A^T B \\ B^T A & B^T B \end{pmatrix}$$

Let V_{ai} represent the data known only to Alice, and let V_{bi} represent the data known only to Bob. To compute β , we want to achieve the following:

1. $V_{a1} + V_{b1} = X^T X$
2. $V_{a2} + V_{b2} = (X^T X)^{-1} = (V_{a1} + V_{b1})^{-1}$
3. $V_{a3} + V_{b3} = X^T Y$
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 $V_{a3} = A^T Y$ and $V_{b3} = B^T Y$; finally step 4 can be achieved using our Matrix Product II protocol.

In each of the step (except the last one), we have used a random number to disguise the intermediate results, such that nobody knows the intermediate results. Even if a party is dishonest, he/she still cannot get useful information because of the randomized disguise.

5.2 Multivariate Classification The goal of building a classification model using Equation 2.2 and 2.3 is to compute C_k and \overline{D}_K . \overline{D}_K can be directly obtained by letting Alice and Bob exchange the corresponding mean values. We describe how to compute C_k without disclosing the raw data.

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:

$$\widehat{D}_k^T \widehat{D}_k = \begin{pmatrix} A'^T A' & A'^T B' \\ B'^T A' & B'^T B' \end{pmatrix}$$

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 .

After Bob knows $A'^T B'$ and $A'^T A'$, he can learn some information about A' . However, since both $A'^T B'$ and $A'^T A'$ are $n \times n$ matrices (assuming both A' and B' are $N \times n$ matrices), and $N \gg n$ (remember that N is the number of subjects, and n is the number of attributes), the amount of information disclosed to Bob is negligible compared to the size of A' . In another words, if Bob treats A' as $N * n$ unknown variables, then Bob only has $2n^2$ equations, which is not enough to solve all these variables.

If Bob selects some specific matrix B' , he can learn up to n^2 values from A' . For example, Bob lets the first rows of B' to be $I_{n \times n}$, then $A'^T B'$ discloses the first n rows of A' to Bob. However, this is not the problem caused by our solution; it is associated with the Secure 2-party Multivariate Classification Problem itself regardless of what the solutions are used. One way to solve this problem is to disallow any party to learn the actual classification model, but this limits the usage of the model because, to make a prediction, one needs to communicate with both Alice and Bob. Another way is to add some random noise to perturb each element of A' before using it in the protocol. This way, even if Bob can learn the values of some elements in the perturbed

A' , he still has trouble determining the original values of A' because of the random noise. The downside of this approach is that the results will be approximate. We are currently investigating how accurate such an approximation approach can achieve.

6 Related Work

The Secure 2-party Multivariate Statistical Analysis problems we described in this paper are special cases of a more general problem, the Secure Multi-party Computation (SMC) problem. The history of the SMC problem is extensive since it was introduced by Yao [26] and extended by Goldreich, Micali, and Wigderson [16], and by many others.

Goldreich states in [15] that the general secure multi-party computation problem is solvable in theory. However, he also points out that using the solutions derived from these general results for special cases of multi-party computation, can be impractical; special solutions should be developed for special cases for efficiency reasons. This is one of the major motivations underlying this work.

Some statistical functions are discussed in the secure multi-party computation framework in [6], which introduces *Selective Private Function Evaluation (SPFE)*. In this problem, a client interacts with one or more servers holding copies of a database $x = x_1, \dots, x_n$ in order to compute $f(x_{i_1}, \dots, x_{i_m})$, for some function f , where indices $i = i_1, \dots, i_m$ are chosen by the client. Ideally, the client must learn nothing more about the database than $f(x_{i_1}, \dots, x_{i_m})$, and the servers should learn nothing. Various approaches for constructing sublinear-communication SPFE protocols were presented in [6], both for the general problem and for special cases of interest, such as the statistical functions.

Secure 2-party *univariate* statistical analysis problems were studied in [9, 11]. Some basic univariate statistical analysis were studied in the paper, including mean value of a data set, standard deviation, correlation coefficient, and linear regression line. Our paper not only extends this work to deal with *multivariate* statistical analysis techniques, but also provides more efficient solutions.

Another body of literature related to this work is the privacy preserving data mining. In general, there are two different approaches: one is to use data distortion approach, in which data are disguised by the added noises [1, 2, 13, 14, 22, 23]. Another general approach is the secure multi-party computation approach [7, 12, 18, 21, 25]. The advantage of the first approach is its performance, but it achieves this at the cost of accuracy. On the other hand, the second approach ensures the results are 100% the same as the results ob-

tained from the original algorithms (without the privacy concerns), but is general much more expensive than the first approach. This work takes the second approach, with the goal of improving the performance.

Both [12] and [25] use dot product as their building block. Secure 2-party computation on dot product of two vectors was originally studied by Du and Atallah [3, 10], and further studied by Vaidya and Clifton [25]. The solution proposed in [25] is more similar to the solution we proposed here than other solutions, because they both use a disguise matrix (in different ways) to mix data together for the privacy purpose. The results in this paper distinguish themselves from [25] in the following aspects: First the work in [25] does not study how the disguise matrix should be selected. As we discussed in this paper, the selection of the disguise matrix is important, and, if selected poorly, can disclose a significant amount of information. Our work has identified and proved that the disguise matrix needs to be $\frac{N}{2}$ -secure. We have also developed an efficient algorithm to come up with such kind of disguise matrix. Second, our work addresses a more general class of computations, the computation on matrices; the dot product is just a special case. Besides the matrix product, we also proposed ways to compute matrix inverse, matrix determinant and norms. Third, our work proposes a methodology to allow two parties to evaluate a more complicated math expression than just the dot product or matrix product.

7 Conclusion and Future Work

In this paper, we have described a new set of problems, the Secure 2-party Multivariate Statistical Analysis (S2-MSA) problems. S2-MSA allows two parties to conduct collaborative statistical analysis on their joint data sets without disclosing each party's private data to the other party. We have developed a practical security model and a number of building blocks to solve two specific S2-MSA problems, the Secure 2-party Multivariate Linear Regression problem and the Secure 2-party Multivariate Classification problem.

In our future work, we will also study more multivariate statistical analysis techniques under the secure 2-party computation framework, such as factor analysis, variance analysis, and cluster analysis. Our goal is to develop a set of useful building blocks that can be used to provide efficient solutions to the secure 2-party multivariate statistical analysis problems.

References

[1] D. Agrawal and C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms.

In *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Santa Barbara, California, USA, May 21-23 2001.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD on Management of Data*, pages 439–450, Dallas, TX USA, May 15 - 18 2000.

[3] M. J. Atallah and W. Du. Secure multi-party computational geometry. In *WADS2001: 7th International Workshop on Algorithms and Data Structures*, pages 165–179, Providence, Rhode Island, USA, August 8-10 2001.

[4] D. Beaver. Commodity-based cryptography (extended abstract). In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, El Paso, TX USA, May 4-6 1997.

[5] D. Beaver. Server-assisted cryptography. In *Proceedings of the 1998 New Security Paradigms Workshop*, Charlottesville, VA USA, September 22-26 1998.

[6] R. Canetti, Y. Ishai, R. Kumar, M. K. Reiter, R. Rubinfeld, and R. N. Wright. Selective private function evaluation with applications to private statistics (extended abstract). In *Proceedings of Twentieth ACM Symposium on Principles of Distributed Computing (PODC)*, 2001.

[7] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools for privacy preserving data mining. *SIGKDD Explorations*, 4(2), December 2002.

[8] G. Di-Crescenzo, Y. Ishai, and R. Ostrovsky. Universal service-providers for database private information retrieval. In *Proceedings of the 17th Annual ACM Symposium on Principles of Distributed Computing*, September 21 1998.

[9] W. Du. *A Study of Several Specific Secure Two-party Computation Problems*. PhD thesis, Purdue University, West Lafayette, Indiana, 2001.

[10] W. Du and M. J. Atallah. Privacy-preserving cooperative scientific computations. In *14th IEEE Computer Security Foundations Workshop*, pages 273–282, Nova Scotia, Canada, June 11-13 2001.

[11] W. Du and M. J. Atallah. Privacy-preserving statistical analysis. In *Proceedings of the 17th Annual Computer Security Applications Conference*, pages 102–110, New Orleans, Louisiana, USA, December 10-14 2001.

[12] W. Du and Z. Zhan. Building decision tree classifier on private data. In *Workshop on Privacy, Security, and Data Mining at The 2002 IEEE International Conference on Data Mining (ICDM'02)*, Maebashi City, Japan, December 9 2002.

[13] W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 24-27 2003.

[14] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proceedings of 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Ed-

monton, Alberta, Canada, July 2002.

- [15] O. Goldreich. Secure multi-party computation (working draft). Available from <http://www.wisdom.wieizmann.ac.il/home/oded/public.html/foc.html>, 1998.
- [16] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, 1987.
- [17] J. Han and M. Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.
- [18] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology - Crypto2000, Lecture Notes in Computer Science*, volume 1880, 2000.
- [19] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. New York, NY: Elsevier Science Publishing Company, Inc., 1977.
- [20] V. S. S. Nair and J. A. Abraham. Real-number codes for fault-tolerant matrix operations on processor arrays. *IEEE Transactions on Computers*, 39:426–435, 1990.
- [21] B. Pinkas. Cryptographic techniques for privacy-preserving data mining. *SIGKDD Explorations*, 4(2), December 2002.
- [22] H. Polat and W. Du. Privacy-preserving collaborative filtering. In *Proceedings of The Third IEEE International Conference on Data Mining (ICDM)*, November 2003.
- [23] S. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.
- [24] J. Stevens. *Applied Multivariate Statistics for the social sciences*. Lawrence Erlbaum Associates, 1986.
- [25] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 23-26 2002.
- [26] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.

Appendix

A Proof of Theorem 4.1

From the protocol, $\hat{A} = A + R_a$ is all what Bob gets that are related to Alice’s private data A . Because of the randomness and the secrecy of R_a , Bob cannot find out A .

We now prove that the protocol is secure if Alice is dishonest. According to the protocol, Alice gets (1) $\hat{B} = B + R_b$, (2) $T = \hat{A} \cdot B + (r_b - V_b)$, and (3) r_a, R_a , where $r_a + r_b = R_a \cdot R_b$. We will show that for any arbitrary B' , there exists r'_b, R'_b and V'_b that satisfies the above equations.

Assume B' is an arbitrary matrix. Let $R'_b = \hat{B} - B'$,

$r'_b = R_a \cdot R'_b - r_a$, and $V'_b = \hat{A} \cdot B' + r'_b - T$. Therefore Alice has (1) $\hat{B} = B' + R'_b$, (2) $T = \hat{A} \cdot B' + (r'_b - V'_b)$, and (3) r_a, R_a , where $r_a + r'_b = R_a \cdot R'_b$. Therefore from what Alice learns, there exists infinite possible values for B .

B Constructing the $\frac{N}{2}$ -secure Matrix M

Let $M = (A \ B)$ be an $N \times N$ invertible matrix and its inverse be $M^{-1} = \begin{pmatrix} C \\ D \end{pmatrix}$, where A, B are $N \times \frac{N}{2}$ matrices and C, D are $\frac{N}{2} \times N$ matrices. In order to satisfy the $\frac{N}{2}$ -secure property, the matrix A (D) must have rank of $\frac{N}{2}$ and the deletion of any $\frac{N}{2}$ rows (columns) in A (D) must not reduce the rank of the resultant $\frac{N}{2} \times \frac{N}{2}$ matrix. Equivalently, any $\frac{N}{2}$ rows (columns) of the matrix A (D) must be linearly independent over \mathfrak{R} , the field of real numbers. Next we will show that the matrix A (D) can be obtained by a generator matrix of an (n, k) maximum distance separable code (MDS code) over finite field $GF(p)$ [19], where p is a prime².

Let G be a generator matrix of an (n, k) linear code over $GF(p)$. Here n denotes the length of the codewords and k the dimension of the code. That is, G is a $k \times n$ matrix with each entry in $GF(p)$. The number of codewords is p^k and each codeword is a linear combination of the rows of G . If $u \in GF(p)^k$ represents a vector of k information symbols, it can be encoded into $c = uG$. An MDS code is defined as the code whose minimum distance of the code is $n + k + 1$, where the minimum distance of a linear code is the smallest number of nonzero entries of all nonzero codewords. It has been proved that every k columns in a generator matrix G of any MDS code are linear independent over $GF(p)$. A famous class of MDS codes is the Reed-Solomon codes which have been applied to many real-world applications. The length of the codewords n of all Reed-Solomon codes over $GF(p)$ is of $p - 1$ and the dimensions of the codes exist for $1 \leq k < n$. For a more detail introduction to MDS codes see [19].

Assume that $A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}, C = (C_1 \ C_2)$, and $D = (D_1 \ D_2)$, where A_i, B_i, C_i, D_i are all square matrices for $i = 1, 2$. That is,

$$M = \begin{pmatrix} A_1 & B_1 \\ A_2 & B_2 \end{pmatrix} \text{ and } M^{-1} = \begin{pmatrix} C_1 & C_2 \\ D_1 & D_2 \end{pmatrix}.$$

Since $MM^{-1} = I$ we have

$$\begin{aligned} A_1 C_1 + B_1 D_1 &= I, & A_1 C_2 + B_1 D_2 &= 0, \\ A_2 C_1 + B_2 D_1 &= 0, & A_2 C_2 + B_2 D_2 &= I. \end{aligned}$$

²Even though MDS codes are defined over $GF(q)$, where q is a prime or the power of a prime, in this paper we only consider those codes over a ground finite field, since the addition and multiplication in this field can be taken as real number addition and multiplication modula p .

Note that A_i and D_i must be all invertible matrices for $i = 1, 2$. Let B_1 be the identity matrix. Then a necessary condition for B_2 which satisfies the above four matrix equations is $D_2^{-1}D_1 + A_2A_1^{-1} = 0$. Under this condition, we have

$$\begin{aligned} B_1 &= I, \quad B_2 = D_2^{-1} + A_2A_1^{-1}, \\ C_1 &= A_1^{-1} - A_1^{-1}D_1, \quad C_2 = -A_1^{-1}D_2. \end{aligned}$$

Let $A_1 = I$. If we choose $D_2 = -I$, then we have $D_1 = A_2$. Consequently, we have

$$\begin{aligned} B_1 &= I, \quad B_2 = -I + A_2, \\ C_1 &= I - A_2, \quad C_2 = I. \end{aligned}$$

The remaining part we need to show is that every $\frac{N}{2}$ rows of $A = \begin{pmatrix} I \\ A_2 \end{pmatrix}$ are linearly independent over \mathfrak{R} and every $\frac{N}{2}$ columns of $D = (A_2 \ -I)$ are linearly independent over \mathfrak{R} .

Before our construction of A and D we first quote three needed theorems from the original materials without given any proof.

THEOREM B.1. [19] *An (n, k) code over $GF(p)$ with a generator matrix G is MDS iff every k columns of G are linearly independent. Furthermore, an (n, k) code with generator matrix $G = (I \ E)$, where E is a $k \times (n - k)$ matrix, is MDS iff every square submatrix (formed from any i rows and any i columns, for any $i = 1, 2, \dots, \min\{k, n - k\}$) of E is nonsingular.*

THEOREM B.2. [20] *Vectors which are linearly independent over $GF(p)$ are also linearly independent over \mathfrak{R} .*

The constructions of A and D are as follows:

Step 1: Find a prime number p which is larger than N .

In practice, one may want to find a prime which is as close to N as possible.

Step 2: Construct the generator matrix G of the $(p - 1, \frac{N}{2})$ Reed-Solomon code over $GF(p)$. The method of this construction can be found in [19].

Step 3: Make the first $\frac{N}{2}$ columns of G form the $\frac{N}{2} \times \frac{N}{2}$ identity matrix by row operations. Assume that $G = (I \ E)$.

Step 4: Delete the last $p - 1 - N$ columns in G such that the resultant matrix $G' = (I \ E')$ becomes an $\frac{N}{2} \times N$ matrix. It is clear that this deletion will not effect the desired linearly independent property that the resultant matrix G' inherits from G before deletion.

Step 5: Take $A = G'^T = \begin{pmatrix} I \\ E'^T \end{pmatrix}$ and $D = (E' \ -I)$. Treat each entry in A to be a real number.

First we prove that every $\frac{N}{2}$ rows (columns) in A (D) are linearly independent over $GF(p)$ and, by Theorem B.2, they are also linearly independent over \mathfrak{R} . According to Theorem B.1, every $\frac{N}{2}$ columns in G of Step 2 are linearly independent over $GF(p)$. In Step 3, the row operations taken transfers the original MDS code into a systematic MDS code where the first $\frac{N}{2}$ symbols of each codeword represent the information symbols. By Theorem B.1 again, after Step 3, every $\frac{N}{2}$ columns of $G = (I \ E)$ are linearly independent. By Theorem B.1, every square submatrix of E is nonsingular. Consequently, every square submatrix of E' is nonsingular after the deletion of last $p - 1 - N$ columns. Therefore, by Theorem B.1 again, both $(E' \ I)$ and $(I \ E'^T)$ are generator matrices for two MDS codes. Therefore, the desired properties of linearly independent are followed immediately for A and D since the minus sign in D does not affect the linearity of D .