Thesis No: CSER-M-18-05

A Study on Secure Outsourcing of Large Scale Computations to Cloud

By

Jannatul Ferdush

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh

November, 2018

A Study on Secure Outsourcing of Large Scale Computations to Cloud

by

Jannatul Ferdush

A thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science & Engineering

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna-9203, Bangladesh

November 2018

Declaration

This is to certify that the thesis work entitled "A Study on Secure Outsourcing of Large Scale Computations to Cloud" has been carried out by Jannatul Ferdush in the Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna, Bangladesh. The above thesis work or any part of this work has not been submitted anywhere for the award of any degree or diploma.

[illegible] 14/11/2018

Signature of Supervisor

Jannatul Ferdush 19.11.18

Signature of Candidate

Approval

This is to certify that the thesis work submitted by Jannatul Ferdush entitled **"A Study on Secure Outsourcing of Large Scale Computations to Cloud"** has been approved by the board of examiners for the partial fulfillment of the requirements for the degree of Master of Science in Engineering in the Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna, Bangladesh in November, 2018.

BOARD OF EXAMINERS

1. [illegible] 14/11/2018 — Supervisor
Prof. Dr. M.M.A Hashem
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology,
Khulna, Bangladesh

2. [illegible] 14/11/18 — Member
Head
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology,
Khulna, Bangladesh

3. [illegible] 14.11.18 — Member
Prof. Dr. K. M. Azharul Hasan
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh

4. [illegible] 14.11.18 — Member
Prof. Dr. Kazi Md. Rokibul Alam
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology,
Khulna, Bangladesh

5. [illegible] 14-11-2018 — Member (External)
Prof. Dr. Md. Rafiqul Islam
CSE Discipline ,
Khulna University,
Khulna, Bangladesh

Acknowledgment

At first, I would like to thank almighty for giving all his blessings on me to complete my task. It is my immense pleasure to express my deepest gratitude to my supervisor Dr. M. M. A. Hashem, Professor, Department of Computer Science and Engineering (CSE), Khulna University of Engineering & Technology (KUET) for his continuous encouragement, constant guidance and keen supervision throughout of this study. I will remember his inspiring guidelines in my future. At last, I am grateful to my parents and friends for their patience, support and encouragement during this period.

Abstract

At present, cloud computing has become the most prominent field for the clients which make them relax to outsource their expensive computations. Based on a pay-per-use model, a client without enough computational power can easily outsource large scale computational tasks to a cloud. But this relaxation also brings some anxious like as data confidentiality, storage overhead and trustworthiness of cloud and so on. To relief from these threats, a perfect cloud outsourcing protocol is needed. A cloud outsourcing system is perfect if it can minimize the client's overhead as well as it has trust. So, the problem is how a client can send his overload task to cloud by privacy preserving and verifiable way. Since mathematical computations always have contribution on various scientific and engineering tasks, we are motivated to design a secure, efficient and verifiable cloud outsourcing protocol for some large-scale mathematical computations such as lyapunov equation, linear regression. Lyapunov equation needs for stability analysis. It is applied to the power system analysis also. To solve it by cloud, we use affine transformation that actually transfers the problem linearly. These transformation computes in a such way that from transferred result original result can be found. Linear regression is studied for with constrained and without constrained. Unconstrained linear regression that is studied here in two ways. First of all, it is studied by hiding its dimension and then transferred the problem by random permutation. Because of increasing dimension, efficiency of this method is decreasing but increased security. It is the trade off between efficiency and security. Second, for transforming it, we use the idea of chaotic map and frobenius matrix. Chaotic map is one of the most used random number generation algorithms and it has some advantages using frobenius matrix over diagonal matrix. At last, constrained linear regression outsourcing is done by affine mapping like as lyapunov equation. Actually, We proposed protocols for outsourcing these problems and studied these protocols in a privacy preserving, efficient and verifiable way. Real cloud is also invoked and a comparison between real and simulation cloud is also showed. Theoretical and experimental results confirm the effectiveness and efficiency of our protocols.

Contents

List of Tables

List of Figures

Nomenclature

LSC	**L**arge **S**cale **C**omputation
MM	**M**atrix **M**ultiplication
LE	**L**yapunov **E**quation
CO	**C**loud **O**utsourcing
LR	**L**inear **R**egression
ULR	**U**nstrainted **L**inear **R**egression
CLR	**C**onstrainted **L**inear **R**egression

Chapter 1
Introduction

1.1 Background

Because of rapid growing communication, data is increasing day by day and also drawing serious attention too. It is approximated that the amount of usable data created will be over 15 zettabytes by 2020, compared to 0.9 zettabytes in 2013 [1]. Because of large data, these computations are very time consuming and need advance technology to reduce the burden for a resource constraint client. Data owners and analysts have to fight to find an efficient way to solve this problem. So, a possible solution is to host supercomputer. Many governments or large companies are derived to host supercomputer for solving this problem. But, in general people can't do it because of high cost and maintenance. So, the above problems are the motivation to develop the cloud outsourcing model. Because in cloud, scalable and flexible IT functionalities are delivered as a service to external customers using Internet technologies.

The reasons why cloud computational outsourcing becomes important from the client's perspective are low computational power, lacking special software, high cost, low infrastructure. If there is no computational outsourcing, it may take huge time, memory etc and also high maintenance cost. Therefore, outsourcing the computation to a third party, e.g. a cloud, is a better choice. These reasons also serve as the motivation for cloud computing [2],[3].

1.2 Motivation

Data comes from various source with large variety and volume. The meaning of big data is also changing. Additionally, it adds also two important features, shown in fig 1.1. Veracity is one of the most prominent new features of big data. It is difficult to maintain big data in the time of cloud adoption [4]. So, it needs to put some concentration to smooth this way and finds the problems with this adoption or outsourcing and corrects these problems. For this reason, we are motivated to design some protocols of cloud outsourcing system for mathematical problems.

Cloud outsourcing is actually used for large scale computation. In various real life problems actually we need mathematical various computations to solve them. So, we are motivated to design some practical cloud outsourcing model of mathematical computations that have huge applications on real life like as lyapunov equation(LE), linear regression(LR).

Lyapunov equation(LE) is actually used for bio-mathematics, mechanics, robotics and control theory. It is also needed to signal processing, filtering, reduction model, to restore image, for ordinary differential equation implementation of implicit numerical methods for solution and block-diagonalization of matrices [6],[7].

Linear Regression is basic task of statistics and data analysis. From the previous data, we can predict future data by linear regression. Prediction is very important for future direction. It can be used in business, image processing(face recognition, facial image quality estimation and so on), software cost

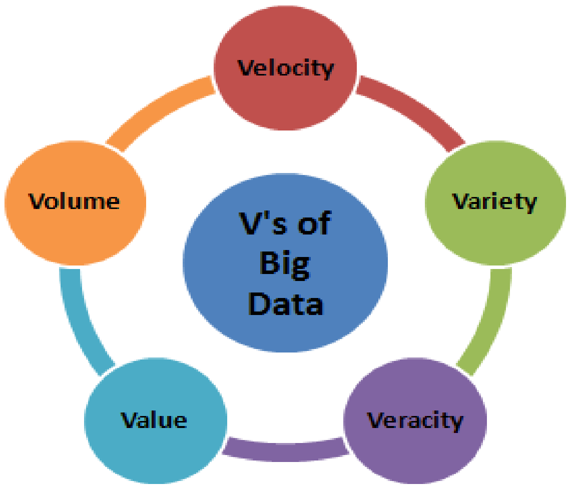

Figure 1.1: Big Data Characteristics [5]

prediction, software effort prediction, software quality assurance, predicting the crime rate of a states based on drug usage, number of gangs, human trafficking and Killings [8].
So from the above discussion, we are motivated to design two protocols that enable clients to securely, verifiably and efficiently outsource lyapunov equation and linear regression to cloud.

1.3 Problem Statement

Outsourcing computational also brings in new security concerns and challenges. First challenge is the privacy of client's input/output. Because data is very important and can contain sensitive information. To hide the data from the cloud, the client needs to encrypt their data before outsourcing and decrypt the returned result from the cloud after outsourcing. The second challenge is the verification of the result. Because cloud is third party and can't be trusted fully. Third challenge is efficiency. For to protect data, we use various encryption and decryption method. But, it needs to design effectively so that the cost of encryption and decryption are low than problem solving. For these reasons, our goal is to design two cloud outsourcing protocols for LE and LR problem that fulfill the goal of correctness, security, verifiability with high efficiency.

1.4 Specific Objective

Among the various mathematical problems, lyapunov equation and linear regression are very popular method. The aim of the study is to design secure, efficient verifiable based cloud outsourcing model for LE and LR. Also, recently developed protocols are investigated in order to compare with proposed method. To reach the goal this study will be carried out with the following specific objectives:

- Study of LE outsourcing problem and a way to solve it.
- Investigate LR with constraints and without constraints.
- Study of ULR with existing methods and find a better way in terms of efficiency or security.
- Investigate CLR outsourcing problem and a way to solve it.
- Compare performance between existing methods.

1.5 Methodology

Figure 1.2 illustrates the cloud outsourcing model for large-scale computation. As it can be observed, there are two parties involved. On one side, there is the client who has a large scale mathematical computational problem, either LE or LR, but has computationally weak devices. Thus, the client intends to outsource the problem to the cloud. On the other side, there is the cloud with powerful resources, huge storage facilities. But the cloud, not being fully secured, cannot be trusted with the original information. So, the client encrypts the original problem into an encrypted problem in order to protect the privacy of input data. Then the encrypted problem is outsourced from the client side to the cloud side. The cloud performs necessary calculations to solve the encrypted problem. Then the cloud provides the solution to the client which is in an encrypted form. In this way, the output privacy is maintained. Along with this, the cloud also sends a proof of verification of the solution. On the client side, the client verifies the obtained result. After result verification, if the client gets to understand that the result is correct, he accepts it. Otherwise if the returned result is perceived to be wrong, then the client rejects it and asks the cloud to recompute it. Ultimately on getting the correct result, the client decrypts it and thus can obtain the solution to the original problem.

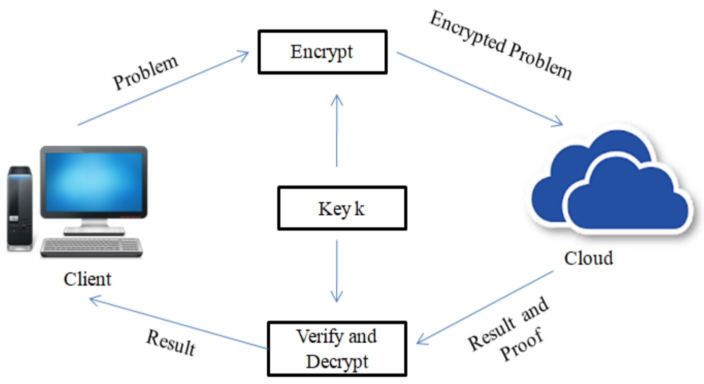

Figure 1.2: Methodology for Large Scale Computational Outsourcing

1.6 Scope of the Thesis

Cloud Computing is a large field and the most growing field of research. From this scope, we choose cloud outsourcing field that is becoming popular day by day because of its effectiveness behavior from the client's view. There are huge applications that depend on cloud outsourcing. From these huge applications, we select some mathematical computational outsourcing as our thesis. Because, in industry or financial companies there are huge data and there very large scale mathematical computations are needed. So, our thesis scope is defined as some mathematical computational outsourcing like as: lyapnuov equation, linear regression problem to cloud.

1.7 Contribution

Our contributions are summarized below:

1. We propose a new encryption schema for lyapunov equation outsourcing model that is secure, efficient and verifiable.
2. We propose two methods for unconstrained linear regression outsourcing of privacy preserving and misbehaviour detection that cloud server can't find out any important information from encrypted inputs.
3. In key generation time the idea of chaotic map and frobenius matrix are studied.
4. We present constrained linear regression outsourcing model that is also secure, verifiable and efficient.

1.8 Organization of the Thesis

The rest of this thesis is organized in five chapters, which are as follows:

Chapter 2 provides literature review related to outsourcing large scale computation to cloud, discussed in this thesis. This chapter provides the previous works over which the improvement is made.

Chapter 3 specifies problem formulation that are essential to model the proposed system.

Chapter 4 introduces the methodology of proposed method. It contains all algorithms that are used to implement proposed protocol.

Chapter 5 contains performance analysis of proposed methods from different angle.

Chapter 6 describes the results with discussion and conclude the thesis work with recommendation of future work.

Chapter 2

Literature Review

2.1 Introduction

Cloud outsourcing is a perfect way to reduce the burden of computation. To keep the advantage of computing ability on the server side, cloud server has to have the ability to compute correctly on encrypted problem and return the correct result to the client. In the age of smartphone, IoT, sensors, it is necessary to outsource and also needs to know how outsourcing protocols are designed previously. This chapter describes the most popular works on cloud outsourcing field.

2.2 Outsourcing of Matrix Operation

Matrix algebra operations are mostly used in scientific computing algorithms and has huge applications. These operations are actually combination of several vector operations. There have been already many studies about secure outsourcing of matrix operations such as: matrix multiplication, inversion, determinant and factorization. Matrix multiplication is used in graph processing, signal processing. At first matrix multiplication with check ability has been studied by Benjamin and Atallah [9]. This paper applied homomorphic encryption(HE) [A.4] system. Here, client computing gain decreases monotonically with the increase in the correctness verification parameter. It has more computational overload. Then Atallah and Frikken [10] proposed a protocol that eliminates the possibility of server collision. This technique used secret sharing idea. To enhance security some fake shares are also introduced as extension. But because of using HE, there downs the performance issue. After, Lei et al. [11] proposed another secure, robust cheating resistance and efficient matrix multiplication outsourcing. It used the idea of monte carlo verification method. But, it suffers that the number of zero elements are the same in both original and encrypted matrix. At 2015, Jia et al. presented another method that is efficient in terms of computation but it suffers from communication and storage overhead [12]. That method has no any verification method by client can check about the computational correctness of cloud. Zhang [13] solved the problem using blind divisors with bilinear pairing. Here, an unique ID is given in every matrix. But, here storage overhead depends on the number of matrix. After that, Shaojing [14] designed random matrix carefully in key generation phase. It also removes the leakage of the number of zero item problem that is the limitation of previous. Another mechanism of MM outsourcing is proposed by Malay. His proposed algorithm is capable of multiplying any valid dimension. It linearly transforms the problem. At 2018, an another method is proposed that used the idea of random permutation [15]. This algorithm used fisher yates suffle method that is already optimized. But, it doesn't consider communication latency.

Following the matrix multiplication [16], Lei et al. [17] proposed a protocol for outsourcing matrix inversion operations that used a similar transformation technique. It used random permuted diagonal matrices. But, the protocol fails over the infinite field R. For the field R, the computational error should be considered seriously. Then, Mohassel [18] also proposed an algorithm for outsourcing matrix

inversion that directly used matrix multiplication idea. So, it can be said that matrix inversion operations borrow the idea of matrix multiplication.
Matrix factorization has huge applications on singular value decomposition, eigen value decomposition and so on. At first, Duan et al. [19] proposed an outsourcing schema that used multiplication of permuted matrices. It uses the iterative idea for result verification. Then, Zhou et al. [20] designed another protocol that used the idea of permutation based data masking technique. It also showed an application that used the idea of eigen and singular value decomposition. But the protocol suffers from the zero item leakage. Another, QR factorization of matrix is proposed by Luo [21]. This algorithm requires lower computational complexity and small communication overhead. Permutation and matrix 1-norm technique based idea is used in non negative matrix factorization outsourcing [22]. After that, Pan et al. [23] proposed another non negative matrix factorization that used lightweight encryption technique.
Determinant computation (DC) is known as another fundamental computation task widely used in scientific and engineering applications, specially in statistics. Lei et al. [24] first proposed DC cloud computation to cloud by dividing the matrix lower and upper part. It verification allows monte carlo simulation. But it also takes iterative rounds to complete its operation.

2.3 Outsourcing of Equation

Wang et al. [25] first proposed an outsourcing solution for a large scale linear equation. But it is an iterative approach. So, it takes many rounds to complete the task. Following this first work, an improved version is proposed by F. Chen et al. [26]. Here, the limitation of previous technique is stated. Using a similar method, X. Chen et al. [27] also another protocol has been proposed that is used sparse matrices to protect the original dense coefficient matrix. Most of the strategies for securely solving linear equation in the literature are transformation-based. The techniques are similar to those used for matrix operations.

2.4 Outsourcing of Optimization

Wang et al. [29] was the first to provide practical mechanism for secure outsourcing of large-scale linear programming computation. It applies affine transform of the problem. Using a similar setup, Nie et al. [28] provided a new secure outsourcing algorithm to lower the client's work. Compared to the solution given in reference [29], this work replaces random dense matrices to sparse matrices. The design follows a similar method to previous. The client outsources two transformed tasks to the cloud and holds two separate keys associated with the transformations. Chen et al. [26] reformulate the LP problem and claims that some transformations used in reference [29] are not necessary. For quadratic programming a secure outsourcing protocol has been proposed by Zhou and Li [30] designed a protocol. This protocol is very close to the protocol of reference [29]. But it also describes the feasible region solution of quadratic programming.

2.5 Outsourcing of Regression

Chen et al. designed two protocols of linear regression outsourcing [31]. One is suitable for efficiency and other is suitable for security. First protocol transforms based on orthogonal matrix and next invertible matrix. But the cost of building orthogonal matrix is huge. Second protocol describes on diagonal matrix. But, it breaks the security of protocol. Then Zhou presented a new method for LR and also described the limitations of previous method [32] at section 3. Here, data is encrypted based on dense matrix. Lots of calculations and performance is not satisfaction level. At last, Malay proposed an regression outsourcing model that utilizes the idea of inverse matrix and random permutation [33]. That is both secure and efficient.

Chapter 3
Problem Formulation

3.1 Introduction

For to the journey of computational outsourcing to cloud, there is huge fear of privacy of data, trustworthiness of cloud. There are the treat of outsourcing computation and to defend these threat, system model is designed. This chapter describes the total problem formulation of cloud outsourcing model.

3.2 Threat Model

Cloud outsourcing has two parts: decreases the computation but increases the concern of data to client. But new challenges are introduced because data is not now on local storage. In case of cloud outsourcing, security threat can be defined by concerning of data security and computational integrity. The security threat in outsourcing system model originated from the suspicious behavior of the cloud server. The previous work for the secure outsourcing computation defines three threat model that are “Trusted Model”, “Semi-Trusted Model” and “Untrusted Model” [17], [24].

3.2.1 Trusted Model

The cloud server follows the all instructions of clients correctly. It has no intention about the access of client’s information. So, no need of security of data and verifiability of result.

3.2.2 Semitrusted Model

In this model, the cloud acts like as “honest but curious” or “lazy but honest” or even both. It follows all algorithm instructions correctly and gives the correct result. But, it has intention about access client’s information and tries to find out important information that hides in data. However, the proposed algorithm has been design to handle these things.The lazy but honest model behaves honestly. It has no intention about data but it is honest. It doesn’t correctly computes on data. It sends random result to client without computation. So, the algorithm has been designed so that its misbehaviour can be detected.

3.2.3 Untrusted Model

The third is "Untrusted Model". Here, the server could be “lazy”, “curious”, and “dishonest”. The cloud server doesn’t follow instructions correctly. Also, it may return incorrect result. So, here both privacy of data and verifiability of result both are equally important.

3.3 System Model

The previous model uses a system model that describes the whole task in cloud outsourcing. Figure 3.1 is actually reflected from their model [29], [17], [30]. It describes the system model for securely outsourcing large scale computation (LSC) problems. We can denote the problem is σ. We already know that there are two parties involved here. Client tasks are: key generation, encryption, verification and decryption. So, at first, he generates key, K and uses this key k to transfer original problem, σ to encrypted problem, σ_k. Then, he sends σ_k to cloud. Cloud solves the σ_k problem and generates encrypted result, R_K and proof, V_P. Client checks V_P correct or not. If correct then he decrypts R_K to find original result R.

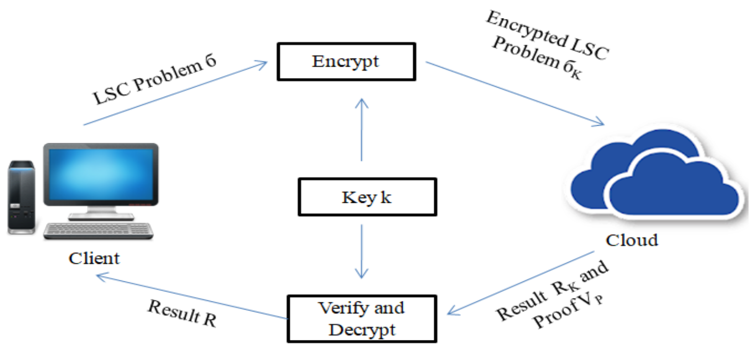

Figure 3.1: System Model of LSC to Cloud

3.4 Framework

- Secure key generation protocol(K): This algorithm generates key, K. This key is used to encrypt the original problem σ.
- Problem encryption protocol($\sigma, K \rightarrow \sigma_k$): The input tuple for LSC is σ. The problem is to encrypt this tuple and find σ_k.
- Problem solving protocol($\sigma_k \rightarrow R_K, V_P$): Solves encrypted LSC problem by any solving method in cloud. This algorithm solves encrypted problem σ_k and sends encrypted result $R_k = Y$ and V_P to client.
- Result Verification Protocol($V_P \rightarrow B$): This algorithm, checks V_P is successfully verified or not. If V_P is verified then it returns $B = 1$ or otherwise 0.
- Result Decryption Protocol($R_K \rightarrow R$): This algorithm decrypts R_K and gets original result R.

3.5 Design Goals

To enable secure and verifiable outsourcing protocol of LSC model, our design protocol must satisfy the following design goals:

- Correctness: Cloud server must correctly solve the LSC. Cloud server must produce an output which can be successfully verified and decrypted by customer.
- Input/Output security: Client data is very sensitive and always important. So, cloud may want to derive the data by the cloud server during performing LSC computation.
- Verifiability: Client doesn't trust cloud. So, he must verify the returned result from cloud.

- Efficiency: Client's computational burden must be less than cloud computation time. Also the computation burden of the cloud server must be comparable with the time complexity of solving LSC problems.

3.6 Problems

Already, many problems are outsourced to cloud. Most of them are mathematical problems that have huge applications on engineering and scientific task. Inspiring from them, following problems are to be outsourced as per system model.

3.6.1 Lyapnuov Equation

Lyapunov Equation is actually a matrix equation. The format of this equation is :

$$AX + XA^T + Q = 0 \tag{3.1}$$

The more general format of this equation is :

$$AXE^T + EXA^T + Q = 0 \tag{3.2}$$

A, E, Q matrices are given, the problem is to find matrices X which satisfy the equation 3.2. We assume that A, Q and E are all square matrices of size m. Q matrices is also symmetric. So the problem can be defined by σ=(A,E,Q) as input and X as output .

3.6.2 Unconstrained Linear Regression(ULR)

Unconstrained Linear Regression is a linear method to find out the linear relationship between variables. It can be written as:

$$Y = X\beta \tag{3.3}$$

Where Y is dependent variable of size $m \times 1$ and X is independent variable of size $m \times n$. β is coefficient of size $n \times 1$. Problem is to find β which satisfy minimum error to fit data. The solution for can be written as:

$$\beta = (X^T X)^{-1} X^T Y \tag{3.4}$$

So the LR problem can be marked $\sigma = (X,Y)$ as input and β as output.

3.6.3 Constrained Linear Regression(CLR)

Constrained linear regression expresses the linear relationship with variables with some constrained.

$$\begin{aligned} &min\ \frac{1}{2}(Cx-D)^2 \\ &such\ that \begin{cases} A.x \le b \\ Aeq.x = beq \end{cases} \quad . \end{aligned} \tag{3.5}$$

Where Y is dependent variable of size $m \times 1$ and X is independent variable of size $n \times n$. β is coefficient of size. Problem is to find β which satisfy minimum error and constraints to fit data .

So the CLR problem can be marked $\sigma = (A, b, Aeq, beq)$ as input and x as output.

Chapter 4

Proposed Method

4.1 Introduction

This chapter describes the whole procedure of proposed method based on the reference of chapter 3. We focused on two popular mathematical functions lyapunov equation and linear regression. Lyapunov equation actually originates from stability analysis of control theory and linear regression comes from to fit data in linear model. In this chapter, a detail discussion about methodology of lyapunov equation and linear regression outsourcing model are described.

4.2 Proposed Method for Lyapunov Equation(LE)

Lyapunov equation is one of the matrix equations. When it has large problem size, it is difficult to solve with low computational resource. Then, it is wise decision outsourcing it to cloud. In this section, general format of lyapunov equation and further how the cloud outsourcing framework can be implemented for it are described.

4.2.1 Framework for lyapunov Equation

Actually refer to chapter 3, laypnuov equation(LE) cloud outsourcing framework can be divided by five protocols: "Key Generation Protocol", "Problem Encryption Protocol", "Problem Solving Protocol", "Result Verification Protocol" and "Result Decryption Protocol". "Key Generation Protocol" describes how client generates secret key, k for LE. It depends on the parameters of LE such as A, E and Q. Next, "Problem Encryption Protocol" represents how the parameters of LE are encrypted so that after encryption it will be another LE problem. Then by using any algorithm of LE solver, cloud solves the encrypted problem. It is discussed on "Problem Solving Protocol". Then cloud sends LE result with proof. By this proof client checks that cloud solves LE correctly or intentionally gives wrong result. After successful verification that cloud returns correct result of LE, client decrypts it. These two methods are discussed on "Result Verification Protocol" and "Result Decryption Protocol". Suppose, the original inputs for $m = 2$ are

$$A = \begin{bmatrix} 8 & 9 \\ 2 & 0 \end{bmatrix} \quad E = \begin{bmatrix} 3 & 1 \\ 0 & 10 \end{bmatrix} \quad Q = \begin{bmatrix} 18 & 16 \\ 16 & 0 \end{bmatrix}$$

- **secure key generation protocol(k):** For to encrypt LE, client generates four random matrices N, M, κ and r as a part of key $K = (N, M, K, r)$. Here, all matrices are square matrices of size m. Here, at first a vector V of size m are generated from a specific range that is given by client. Then, four zero matrices of size m are generated. These matrices are called N_1, M_1, κ_1 and r_1. The specific range are described by $[P_1, P_2]$ and must be $P_1 < P_2$. After that, the diagonal elements of M, N, κ, r are selected from V. Then by four random row permutations, these four parameters N_1, M_1, κ_1 and r_1 are randomized to get new N_1, M_1, κ_1 and r_1 respectively. The detailed algorithm of key generation are shown in Algorithm 1.

```
Algorithm 1: Key Generation for LE
  Input: Size $m$
  Output: Four random square matrices $A$, $E$, $\kappa$ and $r$ of size $m$
1 Client selects randomly two value $P_1$ and $P_2$.
2 Then client generates four square zero matrices $N_1$, $M_1$, $\kappa_1$ and $r_1$ of $m$ size.
3 Randomly generated a vector $V = \{V_1, V_2, ..., V_m\}$ where $P_1 \leq V_i \leq P_2$.
4 for $i \leftarrow 0$ to $m-1$ do
5     $N_1(i,i) = V_i$
      $M_1(i,i) = V_i$
      $K_1(i,i) = V_i$
      $r_1(i,i) = V_i$
6 Four random permutation function $\pi_1$ to $\pi_4$ are generated to permute $N_1$, $M_1$, $K_1$ and $r_1$
    respectively.
7 for $i \leftarrow 0$ to $m$ do
8     $N(i,i) = permute(N_1, \pi_1)$
      $M(i,i) = permute(M_1, \pi_2)$
      $K(i,i) = permute(K_1, \pi_3)$
      $r(i,i) = permute(r_1, \pi_3)$
```

After key generation algorithm, following keys are generated:

$$N = \begin{bmatrix} 0 & 17 \\ 16 & 0 \end{bmatrix} \quad M = \begin{bmatrix} 0 & 12 \\ 13 & 0 \end{bmatrix} \quad \kappa = \begin{bmatrix} 0 & 10 \\ 12 & 0 \end{bmatrix} \quad r = \begin{bmatrix} 0 & 16 \\ 17 & 0 \end{bmatrix}$$

- **Problem Encryption Protocol** LE encryption consists of three parts. Such as: "Hiding Square Matrices", "Hiding Symmetric Matrix" and "Enhancing Security by Affine Mapping". In LE, there are three matrices A, E and Q. All of these matrices are square matrix. Q has also a special property that it is symmetric matrix that means it equals to it's transpose matrix. In 'Hiding Square Matrices' shows how A and E are encrypted and then 'Hiding Symmetric Matrix' describes how Q are encrypted. After that, how affine mapping can be enhanced security are explained in 'Enhancing Security by Affine Mapping'. Also, how M, N, κ and r are generated that are completely known by client and also it is different for different clients because the range from where the values of matrices come that is given by client.
  - **Hiding Square Matrices:** The term A and E from tuple are encrypted by $A' = NMA$ and $E' = NME$. So, without knowing M and N, it is impossible to know the value of A and E.
  - **Hiding Symmetric Matrix:** By the previous generated M and N, the Q matrix can be encrypted via: $Q' = NMQ(NM)^T$ Where Q' is also a symmetric matrix.
  - **Enhancing Security by Affine Mapping:** To enhance the security, output matrix X is hidden

by mapping $X = \kappa Y \kappa^T + rr^T$ and Y is the solution of encrypted problem. Thus the affine mapping of our protocol is defined by transforms X into $Y = \kappa^{-1}(X - rr^T)(\kappa^T)^{-1}$ Then we get,

$$\begin{aligned}
NMA(\kappa Y \kappa^T + rr^T)E^T M^T N^T + \\
NME(\kappa Y \kappa^T + rr^T)A^T M^T N^T + NMQMN = 0 \\
NMA\kappa Y \kappa^T E^T M^T N^T + NMArr^T E^T M^T N^T + \\
NME\kappa Y \kappa^T A^T M^T N^T + \\
NMErr^T A^T M^T N^T + NMQMN = 0 \\
NMA\kappa Y \kappa^T E^T M^T N^T + NME\kappa Y \kappa^T A^T M^T N^T + \\
(NMErr^T A^T M^T N^T + NMQMN + \\
NMArr^T E^T M^T N^T) = 0
\end{aligned} \tag{4.1}$$

So we can write as follows: $A' = NMA\kappa$, $E' = NME\kappa$,

$$Q' = NMErr^T A^T M^T N^T + NMQMN + NMArr^T E^T M^T N^T \tag{4.2}$$

So rewrite Q as: $Q' = NMEr(NMAr)^T + NMAr(NMEr)^T + NMQMN$.
It shows that $NMEr(NMAr)^T$ and $NMAr(NMEr)^T$ are transpose each other. So the summation of these two term is a symmetric matrix. $NMQMN$ represents also a symmetric matrix. So the encrypted Q' matrix is symmetric matrix like Q. Then the new encrypted problem can be denoted via $\sigma_k = (A', E', Q')$. Then the problem is defined by to find Y to satisfy the equation $A'YE'^T + E'YA'^T + Q' = 0$. Detail procedure is shown in Algorithm 2.

Algorithm 2: Problem Encryption for LE

```
  Input: Four keys $N$, $M$, $K$, $r$ and inputs $A$, $E$, $Q$
  Output: Encrypted inputs $(A', E', Q')$
1 Perform $A' = MNA$ $E' = NME$
   $Q' = NMEr(NMAr)^T + NMAr(NMEr)^T + NMQMN$
```

After problem encryption, following encrypted inputs are generated:

$$A' = \begin{bmatrix} 1768 & 1989 \\ 384 & 0 \end{bmatrix} \quad E' = \begin{bmatrix} 663 & 221 \\ 0 & 1920 \end{bmatrix} \quad Q' = 10^9 \begin{bmatrix} -0.8551 & 01.1695 \\ 1.1695 & 0 \end{bmatrix}$$

- **lyapunov Equation solving protocol** This algorithm solves encrypted σ_k problem by cloud and returned $R_k = Y$ and proof parameter P_P to client. Here, there is no extra P_P. Y represents both encrypted result and also proof parameter. The detail algorithm of it is shown in Algorithm 3.

Algorithm 3: Problem Solving for LE

```
    Input: $A'$, $E'$ and $Q'$
    Output: Encrypted result $Y$
1   Cloud invokes any existing lyapunov equation solver can be called lyapsolver to solve it.
      $Y = lyapsolver(A', E', Q')$
2   Cloud then sends $Y$ back to client.
```

After solving the problem, encrypted result is generated by the following algorithm:

$$Y = \begin{bmatrix} -256.3162 & 0 \\ 0 & -289.1567 \end{bmatrix}$$

- **Result Verification Protocol** This algorithm checks the Verifiability of return result Y by proof parameter P_P. Previously stated that in case of LE the encrypted result is itself a proof parameter. By this proof parameter, clients calculates $P_r = A'YE'^T + E'YA'^T$. It will be must equal to Q' without considering sign. When it equals then client generates verify parameter, $V_p = 1$ otherwise 0. The detail algorithm is shown in Algorithm 4

Algorithm 4: Verification for LE

```
    Input: $Y$
    Output: $V_p$
1   Client calculates $P_r = A'YE'^T + E'YA'^T$
2   if $P_r$==$Q'$ then
3       $V_p = 1$
4   else
5       $V_p = 0$
```

Then calculated P_r is

$$P_r = 10^9 \begin{bmatrix} 0.8551 & 1.1595 \\ 1.1695 & 0 \end{bmatrix}$$

- **Result Decryption Protocol** If $V_p = 1$ then client decrypts the original result by $X = KYK' + rr'$. Because, it the mapping between input and output that is previously described. If verification proof, $V_p == 1$ then result is decrypted otherwise problem is re-transmitted. The detail algorithm is shown in Algorithm 5.

Then calculated X is

$$X = \begin{bmatrix} -0.3162 & 0 \\ 0 & -0.1567 \end{bmatrix}$$

4.3 Proposed Method for Linear Regression

Linear Regression(LR) is one of the basic tasks of data prediction. Normally, It can be solved by least square method. It is two types, such as: Unconstrained Linear Regression(ULR) and Constrained

Algorithm 5: Decryption

```
Input: key $K$,$r$, encrypted result $Y$ and $V_p$
Output: $X$
1 if $V_p == 1$ then
2     $X = KYK^T + rr^T$
3 else
4     retransmits the problem.
```

Linear Regression(CLR). It is quite difficult to solve when it works on huge value. Then, it is better to outsource with cloud. In this section, how ULR and CLR utsourcing framework work that are shown. Here, two frameworks for ULR and one framework for CLR are implemented reference to chapter 3. For ULR, one of the framework is based on "Dimension Hiding" and other is "Chaotic Map and Frobenius Matrix [A.1,A.2]". This section shows complete process of ULR outsourcing by hiding the dimension of original problem. Dimension refers the size of its parameter. The parameter for the problem is X and Y. Dimension hiding means to hide the size of this two parameters.

4.3.1 Framework for ULR By Dimension Hiding

This section shows complete process of ULR outsourcing by hiding the dimension of original problem. Dimension refers the size of its parameter. The parameter for the problem is X and Y. Dimension hiding means to hide the size of this two parameters. ULR outsourcing framework can be divided by also Secure Key Generation Protocol, Problem Encryption Protocol, Problem Solving Protocol, Result Verification Protocol and Result Decryption Protocol refer to chapter 3. "Key Generation Protocol" describes how client generates secret key, k for ULR. It depends on the parameters of ULR such as X, Y and a value that is generated by client G. Next, "Problem Encryption Protocol" represents how the parameters of ULR are encrypted so that after encryption it will be another ULR problem. Here, the main part of Problem encryption actually dimension hiding. From my known, It is completely new idea for constrained linear regression outsourcing. Then by using any algorithm of ULR solver, cloud solves the encrypted problem. It is discussed on "Problem Solving Protocol". Then cloud sends ULR result with proof. By this proof client checks that cloud solves LE correctly or intentionally gives wrong result. After successful verification that cloud returns correct result of LE then client decrypts it. These two methods are discussed on "Result Verification Protocol" and "Result Decryption Protocol". Suppose the inputs of ULR are:

$$X_1 = \begin{bmatrix} 11 & 18 & 9 \\ 9 & 6 & 5 \\ 3 & 5 & 19 \\ 10 & 12 & 2 \\ 18 & 13 & 3 \end{bmatrix} \quad Y_1 = \begin{bmatrix} 121 \\ 65 \\ 149 \\ 58 \\ 83 \end{bmatrix}$$

.

- **Secure Key Generation Protocol:** : For linear regression outsourcing, the key $K = (D,k)$ where $D = (R_1,R_2,O,R_3)$ and $k = (T_1,T_2)$. How this $K = (D,K)$. Detail algorithm is shown in Algorithm 6. Here, R_1 is $(G-m) \times (m-n)$, R_2 is $(G-m) \times n$, O is $m \times (m-n)$ and R_3 is a vector $(G-m) \times 1$. Here, O is completely a zero matrix. T_1 is square matrix of size G and T_2 is square matrix of size m. After key generation algorithm, following keys are generated:

$$R_1 = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix} \quad R_2 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \quad R_3 = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad O = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\mathrm{T}_1 = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 9 \end{bmatrix} \quad T_2 = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix}$$

- **Problem Encryption Protocol** : The inputs tuple for CLR is $\sigma = (X,Y)$. The problem is to encrypt this tuple and find $\sigma_k = (X',Y')$. It has three parts: "Hiding the dimension", "Encrypted by Diagonal Matrix" and "Enhancing Security by Random Permutation". After encryption, X and Y is sent to cloud.

  1. **Hiding the Dimension of (X,Y):** For to hide X and Y, we use the idea from [34]. From previously generated random diagonal keys, $D = (R_1,R_2,R_3,0)$ are used for hiding the dimension of X and Y. At first X_1 and X_2 are transferred by the following method:

     $$X_1 = \begin{bmatrix} R_1 & R_2 \\ O & X \end{bmatrix} \quad Y_1 = \begin{bmatrix} R_3 \\ Y \end{bmatrix}$$

     Then the dimension of X_1 is now Gxm and Y_1 is $Gx1$. Then the problem $\sigma(X,Y)$ can be now called $\sigma_d(X_1,Y_1)$.
  2. **Encrypting by Diagonal Matrix:** Two random diagonal matrices are generated T_1 and T_2 are generated of G size. Then, X_1 is encrypted by $X_2 = T_1X_1T_2$ and $Y_2 = T_1Y_1$. This idea is taken from [31]. The dimension of X_2 and Y_2 are same as X_1 and Y_1 respectively.
  3. **Enhancing Security by Random Permutation:** One random permutation function π_1 are used to randomly permuted row of X_2 and corresponding row of Y_2. Another random permutation function π_2 are used to randomly permuted column of X_2 after row permuted to find X_3. After this stage Y_2 is changed to Y_3. Total procedure is shown in Algorithm 7. After encryption, the inputs are changed to:

Algorithm 6: Key Generation for ULR by Dimension Hiding

```
Input: Size $m$ and $n$
Output: Six random matrices $R_1$, $R_2$, $R_3$, $O$, $T_1$ and $T_2$
1  Select randomly a value $G$ where $m < G$.
2  Client generates six zero matrices such as: $R'_1$ of size $(G-m) \times (m-n)$, $R'_2$ is of size
     $(G-m) \times n$, $R'_3$ is of size $(G-m) \times 1$, $O$ is size $m \times (m-n)$, $T_1$ is of size $G \times G$ and $T_2$ is of
     size $m \times m$ Client generates random vector $V_1$. Here, $V_1$ is $G \times 1$ size.
3  Calculate $t_1 = (G-m) * (m-n)$
4  for $i \leftarrow 1$ to $(G-m)$ do
5      randomly select $t$ where 1≤m≤t$_1$
         calculate $e_1 = t \div (m-n)$
6      calculate $e_2 = t \div (G-m)$
7      $R'_1(e_1, e_2) = V_1(i)$
         $R'_2(e_1, e_2) = V_1(i)$
         $R'_3(e_1, e_2) = V_1(i)$
         $T_2(i,i) = V_1(i)$
8  for $i \leftarrow 1$ to $G$ do
9      $T_1(i,i) = V_1(i)$
10 Three random permutation functions $\pi_1$ to $\pi_3$ are generated to permute $R'_1$, $R'_2$ and $R'_3$.
11 for $i \leftarrow 1$ to 3 do
12     $R_1 = permute(R'_1, \pi_1)$
         $R_2 = permute(R'_2, \pi_2)$
         $R_3 = permute(R'_3, \pi_3)$
```

$$X_3 = \begin{bmatrix} 24 & 0 & 144 & 0 & 0 \\ 0 & 243 & 0 & 0 & 6 \\ 0 & 0 & 176 & 360 & 36 \\ 0 & 0 & 180 & 150 & 25 \\ 0 & 0 & 84 & 175 & 133 \\ 0 & 0 & 40 & 60 & 2 \\ 0 & 0 & 648 & 585 & 27 \end{bmatrix} \qquad Y_3 = \begin{bmatrix} 4 \\ 0 \\ 484 \\ 325 \\ 1043 \\ 58 \\ 747 \end{bmatrix}$$

- **Problem Solving Protocol:** Cloud Solves encrypted linear regression problem by any solving method in cloud. This algorithm solves encrypted problem σ_k and sends encrypted result $R_k = \beta_k$ with proof parameter. Here, result β_k is itself proof parameter P_p. Cloud sends encrypted result to client. Detail algorithm is shown in Algorithm 8.

  After solving the problem, encrypted result is generated by the following algorithm:

$$\beta_k = \begin{bmatrix} -2.8333 \\ -0.1728 \\ 0.5000 \\ 0.4000 \\ 7.0000 \end{bmatrix}$$

Algorithm 7: Problem Encryption for LR by Dimension Hiding

```
Input: Six keys $R_1$, $R_2$, $R_2$, $R_3$, $T_1$ and $T_2$ and inputs $X$, $Y$
Output: Encrypted inputs $(X','Y')$
1 Perform $X' = [R_1 \ \ R_2; O \ \ X]$
    $Y' = [R_3 \ \ Y]$
2 Perform $X_2 = T_1 X_1 T_2$
    $Y_2 = T_1 Y_1$
3 Generate random permutation $\pi_1$ and $\pi_2$ to permute $X_2$ and $Y_2$. $\pi_1$ is used for row permutation
    and $\pi_2$ is used for column permutation of both $X_2$ and $Y_2$. Now, the permuted outputs are
    named $X_3$ and $Y_3$.
4 Client sends $X_3$ and $Y_3$ to cloud.
```

Algorithm 8: Problem Solving for ULR by Dimension Hiding

```
Input: $X_3$, $Y_3$
Output: Encrypted result $\beta_k$
1 Cloud computes $\beta_k = (X_3^T X_3)^{-1} X_3^T Y_3$
2 Cloud then Sends $\beta_k$ back to client.
```

- **Result Verification Protocol:** From proof parameter, P_p client calculates verification proof V_P. If cloud honestly calculates β_k then $Y_3 = \beta_k X_3$ relation exists. Then $V_p = 1$ and otherwise 0. Detail algorithm is shown in Algorithm 9.

Algorithm 9: Verification for ULR by Dimension Hiding

```
Input: $\beta_k$,$X_3$,$Y_3$
Output: $V_p$
1 Client calculates $P_r = X_3 \beta_k$
2 if $P_r$==$Y_3$ then
3   $V_p = 1$
4 else
5   $V_p = 0$
```

After solving the problem, for result verification P_r equals to Y_3.

- **Result Decryption Protocol:** After successfully verified client decrypts result by this protocol and here decrypted result $R = \beta$ and lower n portion is result. If $V_p = 1$ then client decrypts the result by $\beta_f = T_2^{-1} \beta_k$. The lower n part is our result β. Detail algorithm is shown in Algorithm 10. Then result is decrypted:

$$\beta = \begin{bmatrix} -2.8333 \\ -0.1728 \\ 0.5000 \\ 0.4000 \\ 7.0000 \end{bmatrix}$$

The lower n part is our result.

```
Algorithm 10: Decryption for ULR by Dimension Hiding
    Input: key $\beta_k$,$T_1$ and $V_p$
    Output: $\beta$
1   if $V_p == 1$ then
2       $\beta_f = T_2^{-1}\beta_k$
3       Perform inverse permutation of $\pi_1$ on $\beta_f$.
          $\beta_n = inversepermute(\beta_f, \pi_1)$ The lower part of $\beta_n$ is desired result and it is called $\beta$.
4   else
5       retransmits the problem.
```

4.3.2 Framework for ULR By Chaotic Map and Frobenius Matrix:

This section shows complete process of ULR outsourcing by chaotic map and frobenius matrix. Chaotic is well known function of randomize and frobenius a popular random matrix. They are used here for to generate secret key, k. Here complete framework is shown for ULR cloud outsourcing by chaotic map and frobenius matrix. ULR outsourcing framework by chaotic map and frobenius matrix can be divided by also "Secure Key Generation Protocol", "Problem Encryption Protocol", "Problem Solving Protocol", "Result Verification Protocol", "Result Decryption Protocol". It is different from the previous method by key generation and problem encryption protocol.

- **Secure Key Generation Protocol:** For linear regression outsourcing, the key $K = (A,B)$ where A and B are two diagonal frobenius matrices of size m and n respectively. Here, the entries of frobenius matrices come from chaotic sequence. Two different chaotic map with different initial values are generated to increase the randomness of key generation and after it permutes too. The detail algorithm is shown in Algorithm 11.

  The following keys are generated:

$$A = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 7 & 0 & 0 \\ 0 & 5 & 0 & 0 & 2 \\ 0 & 0 & 0 & 3 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 3 & 0 & 2 \\ 0 & 0 & 0 \\ 0 & 5 & 0 \end{bmatrix}$$

- **Problem Encryption Protocol:** The inputs tuple for linear regression $\sigma = (X,Y)$. The problem is to encrypt this tuple and find $\sigma_k = (X',Y',X'',Y'')$. Here, $X' = AXB$, $Y' = AY$ and then permute X' and Y' to get X'' and Y''. and $X''' = AX''B$, $Y''' = AY''$. Detail algorithm is shown in Algorithm 12.

  After encryption following encrypted parameters are generated:

$$X' = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix} \quad Y' = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix} \quad X'' = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix} \quad Y'' = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix}$$

- **Problem Solving Protocol:** Cloud solves encrypted linear regression problem by any solving method. This algorithm solves encrypted problem $\sigma_k = (\sigma_{k1}, \sigma_{k2})$ and sends encrypted result $R_k = (\beta_1, \beta_2)$

Algorithm 11: Key Generation of ULR by Chaotic Map and Frobineus Matrix

```
  Input: Size $m$ and $n$
  Output: Two diagonal frobenius matrices permuted version $A$ and $B$, The size of $A$ is $m \times$m and
          other is $n \times n$
1 Select a random value $r$ where $1 \leq r \leq n$
2 Two chaotic sequence $\{s_1, s_2, .........s_{2m-r}\}$ and $\{t_1, t_2, .........t_{2n-r}\}$ are generated from two
    chaotic map. $G_{n+1} = r_1 G_n(1 - G_n)$ and $H_{n+1} = r_2 G_n(1 - G_n)$ Where, $3.599 \leq r_1, r_2 \leq 4$ for
    random behaviour and $G_0 \neq H_0$
3 for $i \leftarrow 0$ to $m$ do
4     $A_z(i,i) = s_i$
5 for $i \leftarrow 0$ to $n$ do
6     $B_z(i,i) = t_i$
7 $g = m + 1$
8 for $i \leftarrow r + 1$ to $m$ do
9     $A_z(i,r) = s(g)$
      $g = g + 1$
10 $g = m + 1$ for $i \leftarrow r + 1$ to $n$ do
11    $B_z(i,r) = t(g)$
      $g = g + 1$
12 Clients generates two random permutation function $\pi_1$ and $\pi_2$ to permute $A_z$ and $B_z$ .
13 $A = permute(\pi_1, A_z)$
14 $B = permute(\pi_2, B_z)$
```

to client. β_3 is sent to client as a proof parameter P_p. Total procedure is shown in Algorithm 13. After problem solving, encrypted result is:

$$\beta' = \begin{bmatrix} 0.2141 \\ 0.9349 \\ 0 \end{bmatrix} \quad \beta'' = \begin{bmatrix} 0.2141 \\ 0.9349 \\ 0 \end{bmatrix}$$

- **Result Verication Protocol:** The proof parameter P_p is now β''.Result is successfully verified if $\beta' = \beta''$. Then $V_p = 1$ otherwise it will be zero. Detail algorithm is shown in Algorithm 14.

Algorithm 14: Verification

```
  Input: $\beta$ and $\beta''$
  Output: $V_p$
1 if $\beta'$=$\beta''$ then
2     $V_p = 1$
3 else
4     $V_p = 0$ retransmit the problem
```

Result verification returns 1 because β' and β'' are equal.

```
Algorithm 12: Problem Encryption for ULR by Chaotic Map and Frobenius Matrix
    Input: Two key $A$, $B$ and inputs $X$, $Y$
    Output: Encrypted pairs $(X', Y')$ and $(X'', Y'')$
1   $X' = AXB$
      $Y' = AY$
2   Permute $X'$ and $Y'$ to get $X''$ and $Y''$.
3   Client sends $\sigma_{k1}(X', Y')$ and $\sigma_{k2}(X'', y'')$ to cloud.
```

```
Algorithm 13: Problem Solving for LR by Chaotic Map and Frobenius Matrix
    Input: Two pair $(X', Y')$ and $(X'', Y'')$
    Output: Output $\beta' and \beta''$
1   $\beta' = (X'^T X')^{-1} X'^T Y'$
      $\beta'' = (X''^T X'')^{-1} X''^T Y''$

2   Cloud sends $\beta'$ and $\beta''$ to client.
```

- **Result Decryption Protocol:** After successfully veried client decrypts result by this protocol and here decrypted result $R = X$ and here $X = \beta$ and β is calculated by $B^{-1}\beta_1$. Detail algorithm is generated by Algorithm 15.

```
Algorithm 15: Decryption of ULR by Chaotic Map and Frobenius Matrix
    Input: $\beta'$
    Output: $\beta$
1   Client calculates $\beta = B^{-1}\beta'$
```

After decryption, original result is:

$$\beta = \begin{bmatrix} 2 \\ 3 \\ 10 \end{bmatrix}$$

4.3.3 Framework for Constrained Linear Regression(CLR)

Refer to chapter 3 the detail framework for CLR are described below. Here are mainly five parts. Such as: "Key Generation Protocol", "Problem Encryption Protocol", "Problem Solving Protocol", "Result Verification Protocol" and "Result Decryption Protocol". CLR problem is very close to linear programming [29]. But the objective functions of these two problem are different.
Suppose, the parameters of CLR are:

$$C = \begin{bmatrix} 6 & 1 & 6 & 4 \\ 2 & 5 & 9 & 5 \\ 2 & 1 & 1 & 2 \\ 2 & 5 & 3 & 5 \\ 6 & 2 & 3 & 7 \end{bmatrix} \quad D = \begin{bmatrix} 6 \\ 4 \\ 7 \\ 7 \\ 2 \end{bmatrix} \quad A = \begin{bmatrix} 5 & 6 & 7 & 7 \\ 10 & 8 & 3 & 3 \\ 6 & 3 & 6 & 3 \end{bmatrix} \quad b = \begin{bmatrix} 10 \\ 5 \\ 6 \end{bmatrix}$$

$$\text{Aeq=}\begin{bmatrix}7 & 5 & 7 & 4\\ 7 & 2 & 10 & 7\end{bmatrix} \quad \text{beq=}\begin{bmatrix}9\\ 5\end{bmatrix}$$

- **Secure Key Generation Protocol:** : For constrained least square problem outsourcing, the key $k = (\gamma,\beta,m,r)$ Here, γ is a square matrix of size k, β is square matrix of size o, m is square matrix of size n and r is a vector $n \times 1$. For this three random vectors are created to initialize the diagonal value of matrices and one random vector is created to initialize r. Then permutation functions are used to permute these values. The detail algorithm is shown in Algorithm 16.

Algorithm 16: Key Generation of CLR

```
   Input: Size $k$,$o$,$n$
   Output: Four random matrices $\gamma$, $\beta$, $m$ and $r$
1  Randomly created three vectors $V_1$ to $V_3$. $V_1$ is size $k$, $V_2$ is size $o$, $V_3$ is size $n$.
2  Generates three zero matrices $\gamma$, $\beta$, $m$ of size $k \times k$, $o \times o$ and $n \times n$ respectively. A zero vector $r$
     of size $n \times 1$ is also generated.
3  for $i \leftarrow 1$ to $k$ do
4      $\gamma'(i,i) = V_1(i)$
5  for $i \leftarrow 1$ to $o$ do
6      $\beta'(i,i) = V_2(i)$
7  for $i \leftarrow 1$ to $n$ do
8      $m'(i,i) = V_3(i)$
         $r'(i) = V_3(i)$
9  Generates three random permutation $\pi_1$ to $\pi_3$.$\pi_1$ to randomly permute row of $\gamma$, $\beta$ and $m$
     respectively. Here $\pi_3$ is also used to permute $r$.
10 for $i \leftarrow 1$ to 4 do
11     $\gamma = permute(\gamma', \pi_1)$
         $\beta = permute(\beta', \pi_2)$
         $m = permute(m', \pi_3)$
         $r = permute(r', \pi_4)$
```

After key generation, following keys are generated:

$$\gamma = \begin{bmatrix}7 & 0\\ 0 & 2\end{bmatrix} \quad \beta = \begin{bmatrix}2 & 0 & 0\\ 0 & 0 & 7\\ 0 & 4 & 0\end{bmatrix} \quad mu = \begin{bmatrix}0 & 0 & 0 & 7\\ 6 & 0 & 0 & 0\\ 0 & 0 & 10 & 0\\ 0 & 0 & 0 & 9\end{bmatrix} \quad r = \begin{bmatrix}2\\ 1\\ 8\\ 6\end{bmatrix}$$

- **Problem Encryption Protocol:** Problem encryption transfer σ to σ_k. The inputs tuple for CLR problem is $\sigma = (X,Y,A,b,Aequ,bequ)$ .The problem is to encrypt this tuple and find $\sigma_k = (X',Y',A',b',A'equ,b'equ)$. It has four parts: "Hiding Equality Constraints", "Hiding Inequality Constraints", "Hiding Objective Function" and "Enhancing Security by Affine Mapping [A.3]". The detail algorithm is shown in Algorithm 17.

- **Hiding Equality Constraints(Aequ,Bequ):** For the first time, a randomly generated k×k matrix γ is a part of the secret key K. Now, the customer can do following transform,

$$Aeq.x = beq \Rightarrow Aeq'.x = beq'$$

where $Aeq' = \gamma Aeq$ and $beq' = \gamma beq$.

- **Hiding Inequality Constraints(A,b):** For the inequality constraints, a randomly generated o×o matrix β is used as a part of the secret key K. So the inequality constraints can be expressed as follow,

$$A.x \leq b \Rightarrow A'.x \leq b'$$

Where $b' = \beta b, A' = \beta A$.

- **Hiding Objective Function:** We have to encrypt the objective function. So the encryption process is done in this segment as follow,

$$min\ \frac{1}{2}(Cx - D)^2 \Rightarrow min\ \frac{1}{2}(C'y - D')^2 \tag{4.3}$$

The in above Eq.(2), Where $C' = Cm$ and $D' = D - Cr$ for the equation $x = my + r$. Beside it is not possible to drive the original objective function $min\ \frac{1}{2}(Cx - D)^2$ without knowing the term C, r, y.

- **Enhance Security by Affine Mapping:** To enhance the security strength of CLLS outsourcing, we have to change the original CLLS and at the same time, it is neccesary to change the output vector x. In this paper, we proposed the encryption scheme Φ to Φ_k. In our proposed scheme, m is a randomly generated n×n non-singular matrix and r be an n×1 vector. The affine mapping transforms x into y = $m^{-1}(x - r)$. So the CLLS problem Φ can be represented as follow, For the objective function

$$min\ \frac{1}{2}(Cmy - (D - Cr))^2 \tag{4.4}$$

Such that $Amy \leq b - Ar$ and $Aeq.my = beq - Aeq.r$. Next by using the basic techniques to pick a random γ for equality constraints, then β for inequality constraints. Both are random square matrix of k size. Now the problem can be written as follow,

$$\begin{aligned} & min\ \frac{1}{2}(Cmy - (D - cr))^2 \\ & such\ that \begin{cases} \beta Amy \leq \beta b - \beta Ar \\ \gamma Aeq.my = \gamma beq - \gamma Aeq.r \end{cases} . \end{aligned} \tag{4.5}$$

The above Eq.(4) is the encrypted form of constraints linear least problem (CLLS).
So, it can denote the constraints of above CLLS as maintaining the rule $A' = \beta A.m$, $b' = \beta(b - A.r)$, $Aeq' = \gamma Aeq.m$, $beq' = \gamma(beq - Aeq.r)$, $C' = C.m$, $D' = D - Cr$.
So from the above discussion, the CLLS problem σ_k = (C',D',A',b',Aeq',beq') which is equivalent to σ . Total encryption process is shown in Algorithm 17.

Algorithm 17: Problem Encryption of CLR

```
Input: Four keys $\gamma$, $\beta$, $m$, $r$ and inputs $C$, $D$, $A$, $b$, $Aeq$ and $beq$
Output: Encrypted inputs $(C',D',A',b',Aeq',beq')$
1 Client calculates
    $A' = \beta A.m$
    $b' = \beta(b - A.r)$
    $Aeq' = \gamma Aeq.m$
    $beq' = \gamma(beq - Aeq.r)$
    $C' = C.m$
    $D' = D - Cr.$
2 Client sends $\sigma_k$ = $(C',D',A',b',Aeq',beq')$ to cloud.
```

Then encrypted inputs are:

$$C' = \begin{bmatrix} 6 & 0 & 60 & 78 \\ 30 & 0 & 90 & 59 \\ 6 & 0 & 10 & 32 \\ 30 & 0 & 30 & 59 \\ 12 & 0 & 30 & 105 \end{bmatrix} \quad D' = \begin{bmatrix} -79 \\ -101 \\ -18 \\ -56 \\ -78 \end{bmatrix} \quad A' = \begin{bmatrix} 10 & 12 & 14 & 14 \\ 42 & 21 & 42 & 21 \\ 40 & 32 & 12 & 16 \end{bmatrix} \quad b' = \begin{bmatrix} 20 \\ 42 \\ 20 \end{bmatrix}$$

$$\text{Aeq'=} \begin{bmatrix} 49 & 35 & 49 & 28 \\ 14 & 4 & 20 & 14 \end{bmatrix} \quad \text{beq'=} \begin{bmatrix} 63 \\ 10 \end{bmatrix}$$

- **Problem Solving Protocol:** Solves encrypted linear regression problem by any solving method in cloud. This algorithm solves encrypted problem σ_k and sends encrypted result $R_k = Y$ to client. It is itself proof parameter, P_p. The algorithm is shown in Algorithm 18.

Algorithm 18: Problem Solving of CLR

```
Input: Encrypted inputs $C'$, $D'$, $A'$, $b'$, $Aeq'$, $beq'$
Output: Output $Y$
1 Y=clrsolver(C', D',A', b', Aeq', beq')
2 Cloud then Sends $Y$ back to client.
```

Then encrypted result is:

$$Y = \begin{bmatrix} -1.2407 \\ 1.8580 \\ 2.0987 \\ -1.5740 \end{bmatrix}$$

- **Result Verification Protocol:** Here verification proof V_P checks if cloud honestly calculates β_k then $Aeq'Y = beq'$ relation exists. If exists then $V_p = 1$ otherwise 0. Detail algorithm is shown in Algorithm 19.

Algorithm 19: Verification of CLR

```
  Input: $Y$, $Aeq'$,$beq'$
  Output: $V_p$
1 Calculate $F$= $Aeq' * Y$ if $F$=$beq$ then
2     $V_p$=1
3 else if then
4     $V_p = 0$
      retransmit the problem
```

Then calculated F is

$$F = \begin{bmatrix} 11 & 18 \\ 9 & 6 \end{bmatrix}$$

So, here $V_P = 1$.

- **Result Decryption Protocol:** After successfully veried client decrypts result by this protocol and here decrypted result $R = X$ and here X is calculated by $X = my + r$. This procedure is shown by Algorithm 20.

Algorithm 20: Decryption of CLR

```
  Input: $Y$, $m$, $r$
  Output: $X$
1 Calculate $X = mY + r$
```

Then calculated X is

$$X = \begin{bmatrix} -0.8939 \\ 1.7424 \\ 1.6364 \\ -1.2273 \end{bmatrix}$$

Chapter 5

Performance Evaluation

5.1 Introduction

The chapter describes the detail study of results and discussions for Lyapnuov Equation(LE) and Linear Regression(LR). From refer to chapter 4, we know that LR is classified by unconstrained linear regression(ULR) and constrained linear regression(CLR). In this chapter, theoretical analysis of all problems are shown.

5.1.1 Theoretical Analysis for Lyapnuov Equation

Theoretical analysis of LE consists of security analysis, verifiability analysis and efficiency analysis. Security analysis describes 'Input Privacy Analysis' and 'Output Privacy Analysis'. 'Input Privacy Analysis' describes the privacy of (A, E, Q) and output privacy analysis describes the privacy of Y.

- **Input Privacy Analysis:** At first, the relationship between the original Lyapnuov equation σ and σ_k are considered. So we consider the process how σ is converted to σ_k. It is also issue for security concern. The data which is available for cloud is $\sigma_k = (A', E', Q')$. The semi honest cloud always tries to discover the original input value from encrypted input value. The A is multiplied by a random square matrices N, M, K of dimension m. Here firstly N and M matrices are multiplied and then result matrix is multiplied by original A and at last NMA matrix is multiplied by K. For to know A, cloud must know N, M, K but it is so difficult. So the structure pattern of A can not be exposed any more and this analysis as same for E. So input privacy is preserved.
- **Output Privacy Analysis:** For to check output privacy needs to know that we let $X = KYK^T + rr^T$. So for knowing X, cloud needs to know K and r. Both are square matrices of size m. If the K and r domain size is denoted by η then cloud needs to guess $m!\eta^{2m}$ times to recover X from Y. So output privacy is also preserved.
- **Verifiability Analysis:** The encrypted problem is to find Y such that $A'YE'^T + E'YA'^T + Q' = 0$. So if Y is the exact solution of $A'YE'^T + E'YA'^T + Q' = 0$ then Y must satisfy it. Our protocol checks it and verifies the result of LE returned from cloud.
- **Efficiency Analysis:** For to key generation, encryption problem and decryption needs $\mathcal{O}(n)$, $\mathcal{O}(n^2)$ and $\mathcal{O}(n)$ respectively. So but in cloud, for solving LE problem needs almost $\mathcal{O}(n^3)$. So customer efficiency or speedup is increased with size of problem.

5.1.2 Theoretical Analysis for ULR by Dimension Hiding

Theoretical analysis describes the privacy of input and output privacy, verifiability and efficiency analysis of of ULR by dimension hiding. In the dimension hiding model, it has three stage of encryption. So, it is difficult to break.

- **Input Privacy Analysis** If we want to analyze the inputs security of LR problem then how Y is changed to Y_k need to be considered. Here, inputs of LR X and Y are gone by three stage to nd the nal encrypted form X_3 and Y_3. This X_3 and Y_3 are available to cloud for computation. At rst stage the dimension of X and Y are hidden by total two random matrix, a zero matrix and a random vector. So, cloud is now blind about an important pattern of data. To nd out the original dimension and data these matrix and vector must be discovered. It is difcult to nd out in reasonable time. Then encryption is done by diagonal matrix of size G and m. But after that, some security concerns come. So, two random permutations are used. For this, the data and its structure are totally unknown by cloud or others except client. So, the inputs privacy is preserved.

- **Output Privacy Analysis** From the decryption equation it is clear that cloud operates only on nal encrypted data and so after calculation it nds encrypted result. For this decryption, it has followed two stages. T_2 should be known which is only known by client. Moreover, it is one time key. That means one key for one problem. So, known plaintext or known ciphertext attack are not possible. So, output privacy is preserved also.
- **Verifiability Analysis:** The encrypted problem is to find β_k such that $Y_3 = \beta_k X_3$. So if β_k is the exact solution of the encrypted problem then β_k must satisfy it. Our protocol checks it and verifies the result of ULR returned from cloud.
- **Efficiency Analysis:** ULR actually takes $\mathcal{O}(n^3)$ to solve. For to key generation, encryption problem and decryption needs $\mathcal{O}(n)$, $\mathcal{O}(n^2) + \mathcal{O}n$ and $\mathcal{O}(n)$ respectively. Here, dimension is increased by encryption. So customer efficiency or speedup is decreased here.

5.1.3 Theoretical Analysis for ULR by Chaotic Map and Frobenius Matrix

It describes the privacy of both input parameter and output result, verifiability and efficiency analysis. Here data of keys come from chaotic and from the idea of using random diagonal matrix, it is deviated to use the idea of frobenius matrix.

- **Input Privacy Analysis:** In this method, X can be transmitted by AXB and Y is transmitted to AY where A and B are frobenius matrix and their elements come from chaotic sequence. So, they are invertible matrix and can't be predicted. Further, they are permuted by permuted function and for every new problem a new initial value is selected like as one time pad. So, the inputs are secured from outside world..
- **Output Privacy Analysis** For to check output privacy needs to know that we let $\beta = B^{-1}\beta'$ So, also we need to know B that is frobenius chaotic matrix in permuted form. So, it is difficult to find out.
- **Verifiability Analysis:** Here, cloud solves two encrypted version of same problem and their results are β and β''. So, they must be equal.
- **Efficiency Analysis:** Here, for solving ULR problem needs almost $\mathcal{O}(n^3)$. For to key generation by chaotic and frobenius,encrypting problem and decryption needs $\mathcal{O}(n)$, $\mathcal{O}(n^2)$ and $\mathcal{O}(n)$ respectively. So customer efficiency or speedup is increased with size of problem.

5.1.4 Theoretical Analysis for Constrained Linear Regression

The theoretical analysis of CLS is covered both input and output privacy, verifiability and efficiency analysis. The structure of secret key is known by client only and output is mapped to original output by affine mapping. So, both input, output privacy are preserved.

- **Input Privacy Analysis:** Every time a new key is sent to cloud for a new problem like one time pad. So, this scheme is free from chosen cipertext attack. The matrix multiplications are the most time consuming operations that is happened in customer side. To insure input privacy, we implement a algorithm. This algorithm encrypts the input tuple Φ into Φ_k with the secret key K. The encrypted problem Φ_k has the same form as Φ. Input privacy is preserved because we used γ for hiding equality constraint, β for hiding inequality constraint and m and r for objective function. its not possible for cloud to know the original data without knowing the γ, β, m and r. Also, these key generation algorithm is run on client side. So, the structure is difficult to find out. So from the above analysis it can be said, its maintain input privacy.
- **Output Privacy Analysis:** Cloud server need to solve the encrypted CLR problem Φ_k and generating the result proof. For preserving output privacy, cloud needs to know that $x = my + r$ and for finding x and also cloud needs to know m and r that is a randomly generated n$\times$n matrix and n$\times$1 vector from client side.. So it takes too many times to recover x from y. So here output privacy is preserved.
- **Verifiability Analysis:** The encrypted problem is to find Y such that $Aeq'Y = Beq'$. So if Y is the exact solution then Y must satisfy it. Our protocol checks it and verifies the result of CLR returned from cloud.
- **Efficiency Analysis:** CLR is highly efcient for key denition, data encryption and decryption. CLR reduces the computational complexity for the customer side. Cloud needs more than $\mathcal{O}(n^3)$ time for solving the CLR problem. But, in customer sides it only takes $\mathcal{O}(n^2)$. So overall the proposed system allow the customer to solve their CLR problem to the cloud and achieve massive computational savings

Chapter 6

Results and Discussions

6.1 Introduction

This chapter shows the comparison between real and simulation cloud environment for the problem of laupunov equation and linear regression problem. Linear regression is studied here both with constrained and without constrained. The comparison works actually on encrypted problem solving time. Encrypted problem is run both cloud and simulation environment and the run time is selected for comparison.

6.2 Simulation Environment

simulation Both client and cloud server computations conducted on a same workstation. If we implement the protocol for both client side and cloud side on a same workstation and measure their running time, then the ratio of time means the asymmetric amount of computation performed in both sides. The implementation is done using matlab 2016 in an identical computer with an Intel(R) Core(TM)i7-7500 CPU 2.70 GHz processor and 16GB RAM. The cloud side is run on matlab online cloud platform 'MathWorks Cloud'[35].

6.3 Evaluation Criteria

For evaluation criteria, three terms are needed. $T_{original}$ that describes the time to solve original problem, T_{client} tells the summation time of key generation, problem encryption, Verification and result decryption time. $T_{original}$ mainly very small compared to T_{client}. Otherwise, outsourcing has no value. T_{cloud} tells the solving time of encrypted problem time. In simulation environment, it closes to $T_{original}$ but in real environment, it is very small compared to $T_{original}$. Client Speedup denotes by $\frac{t_{original}}{t_{client}}$, which represents the speedup in the customer side when SE problem is outsourced. Efficiency of our protocol denotes by $\frac{t_{original}}{t_{cloud}}$, represents the speedup of outsourcing protocol. It is desirable that both speedup and efficiency is greater than 1.

6.3.1 Lyapunov Equation

In Table 6.1 shows the experimental result of LE outsourcing in simulation environment. Here problem size increased from 1000 to 5000. From this, it is clear that $t_{original}$ time is close to t_{cloud} and t_{client} is less than $t_{original}$.

The fig 6.1 shows the comparison between real and simulation cloud for LE. Real cloud normally has better resource facility than local cloud. Encrypted LE actually changes the domain of its parameter. So, encrypted LE is just the transformed version of original LE. On the other hand, cloud is a place where clients take the facility of on demand computation, pay as you go model. It works on virtualization

Table 6.1: Experimental Result of Outsourcing of Lyapunov Equation

Size m	$t_{original}$	t_{cloud}	t_{client}	$\frac{t_{original}}{t_{client}}$	$\frac{t_{original}}{t_{cloud}}$
1000	13.4370	15.2324	3.4453	0.2564	0.8821
2000	119.8130	121.2310	18.3023	6.6111	0.98823
3000	396.2478	395.3423	48.7209	8.2523	1.0023
4000	670.4534	678.5678	78.1201	8.5823	0.9909
5000	1100.5465	1200.3487	101.1234	10.8911	0.9169

technique. So, when LE encrypted problem is run both on cloud and local computer, then cloud takes less time than local or simulation environment.

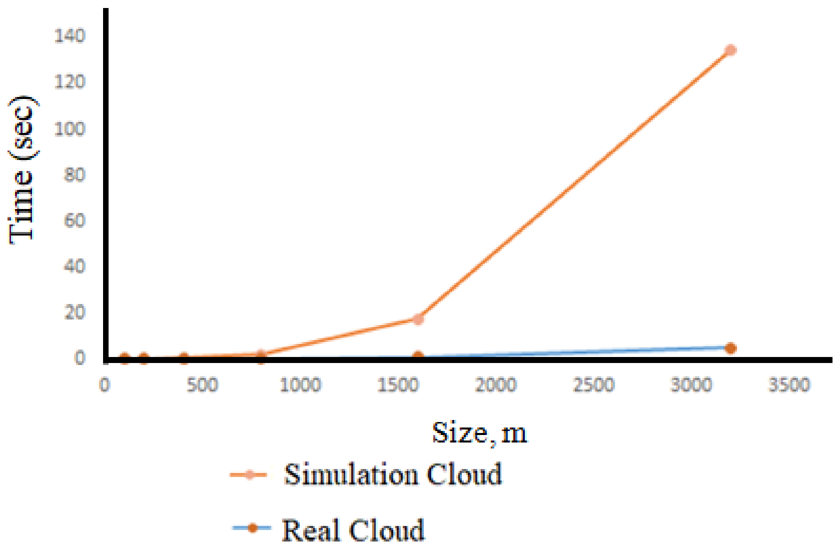

Figure 6.1: Comparison Between Real and Simulation Cloud Time for LE

6.3.2 Linear Regression by Dimension Hiding

Table 6.2 shows linear regression outsourcing result by dimension hiding. Here, experimental result for CLR by dimension hiding is shown. The encrypted problem is here the transformed of original problem respect to both dimension and domain. By this proposed method, dimension of problem is increasing after encryption. It is secure but encrypted problem time t_{cloud} is high than $t_{original}$ time. It is drawback of this system.

Figure 6.2 shows the comparison between real and simulation cloud for CLR by dimension hiding. Dimensionality is a curse and in this method dimension is increasing. So, it has shown in fig 6.2 that simulation time takes less time than cloud. So, it breaks the efficiency which is the one of our design goals. Our another design goal is security and it meets it. So, when we don't want to take the burden of computation in locally and also we compromise our efficiency than it is the best option. It is also the trade off between efficiency and security.

Table 6.2: Experimental Result for ULR by Dimension Hiding

size m×n ×G	$t_{original}$	t_{client}	t_{cloud}	$\frac{t_{original}}{t_{client}}$	$\frac{t_{original}}{t_{cloud}}$
600×50 ×1200	0.0106	0.0112	0.0697	0.964	0.152
700×100 ×140	0.0112	0.0169	0.1000	0.657	0.112
800×200 ×1600	0.0168	0.0193	0.1529	0.870	.109
900×400 ×1800	0.0374	0.2339	0.1816	0.159	0.216
1000×500 ×2000	0.0456	0.0444	0.2569	1.027	0.185

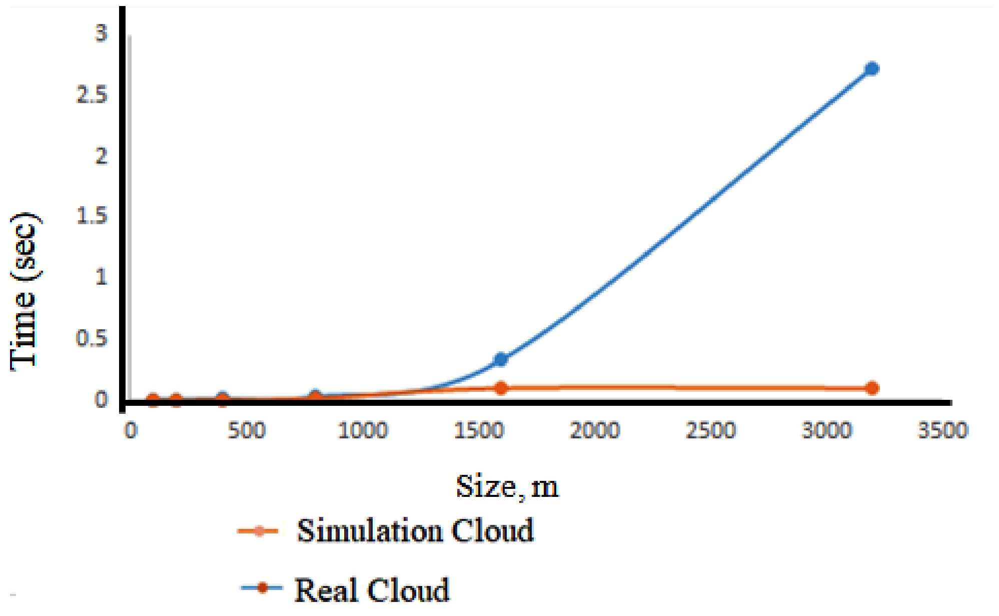

Figure 6.2: Comparison Between Real and Simulation Cloud Time of LR by Dimension Hiding

In, literature review, it is studied that there are also some method on unconstrained linear regression. The fig 6.3 shows the comparison of t_{client} between among them. It is clear that our proposed method is not efficient like as other. Because, dimension is increasing and so problem encryption time is also increasing with respect to other.

6.3.3 Linear Regression by Chaotic and Frobenius Matrix

Table 6.3 shows the experimental data for LR outsourcing by chaotic map and frobenius matrix in simulation environment. Here, also $t_{original}$ and t_{cloud} are close and t_{client} is less than other. So, here also shows the customer speedup is greater than 1 and efficiency is very close to 1.

Figure 6.4 shows the comparison between real and simulation cloud for CLR by chaotic and frobenius matrix. Simulation cloud takes more time than real cloud as usual.

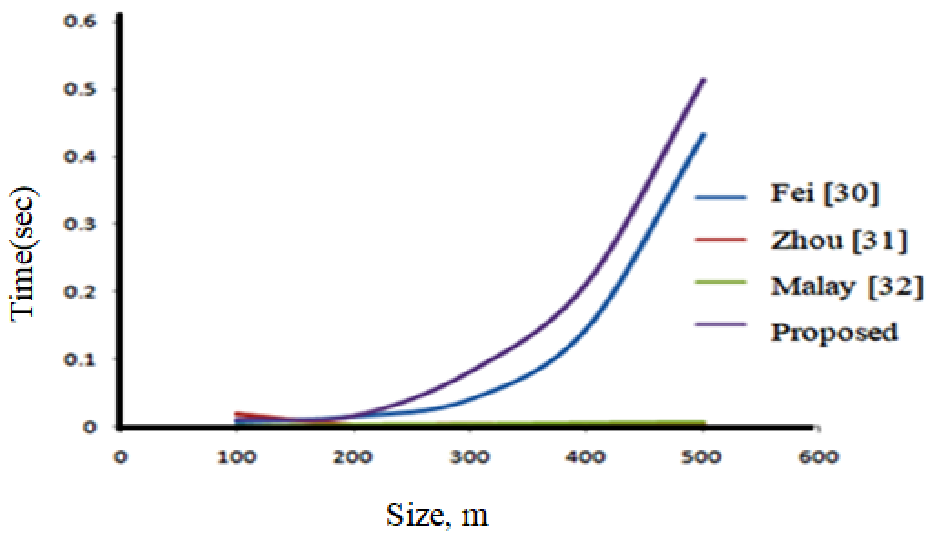

Figure 6.3: Comparison Between Previous Methods and proposed Dimension Hiding Method

Table 6.3: Experimental Result for ULR by Chaotic Map and Frobenius Matrix

size m×n	$t_{original}$	t_{client}	t_{cloud}	$\frac{t_{original}}{t_{client}}$	$\frac{t_{original}}{t_{cloud}}$
100×50	0.0100	0.0066	0.0111	1.515	0.900
200×100	0.0104	0.0078	0.0122	1.333	0.852
400×200	0.0132	0.0146	0.0159	0.904	.830
800×400	0.0340	0.0388	0.0490	0.876	0.694
1600×800	0.3348	0.2669	0.4927	1.254	0.679
3200×1600	2.7275	1.7404	6.4130	1.567	0.917
6400×3200	27.0871	12.9309	38.4005	2.095	0.705

6.3.4 Constrained Linear Regression

In Table 6.4 shows the experimental result of CLR outsourcing in simulation environment. From this, it is clear that $t_{original}$time is close to t_{cloud} and t_{client} is less than $t_{original}$. So, here also shows the customer speedup is always greater than 1 and efficiency is very close to 1.

Figure 6.5 shows the comparison between real and simulation cloud for constrained linear regression. Like as previous, simulation time takes less than cloud time.

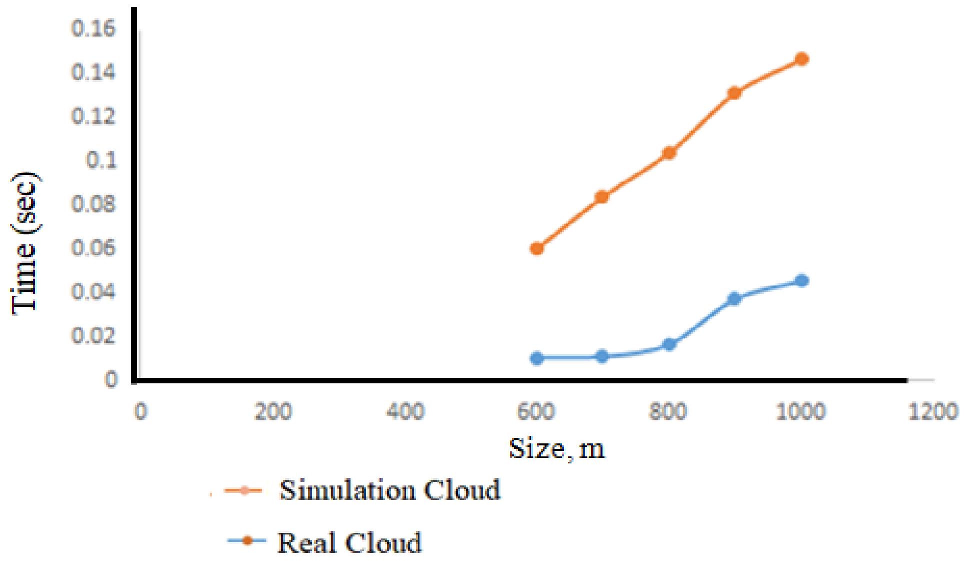

Figure 6.4: Comparison Between Real and Simulation Cloud Time of LR by Chaotic and Frobenius Matrix

Table 6.4: Experimental Result for Constrained Linear Regression

size m×n ×o ×k	$t_{original}$	t_{client}	t_{cloud}	$\frac{t_{original}}{t_{client}}$	$\frac{t_{original}}{t_{cloud}}$
100×50 ×25 ×20	0.199	0.005	0.173	39.800	1.15
200×100×50 ×40	0.255	0.006	0.226	42.500	1.128
400×100 ×50 ×80	0.416	0.009	0.468	46.220	.889
800×200 ×100 ×160	1.846	1.001	1.907	1.844	1.001
1600×400 ×200 ×320	17.261	1.085	17.457	15.909	0.989
3200×800 ×400 ×640	123.231	2.435	134.424	50.608	0.917
640×1600 ×800 ×1280	1059.31	3.368	1112.41	314.519	0.953

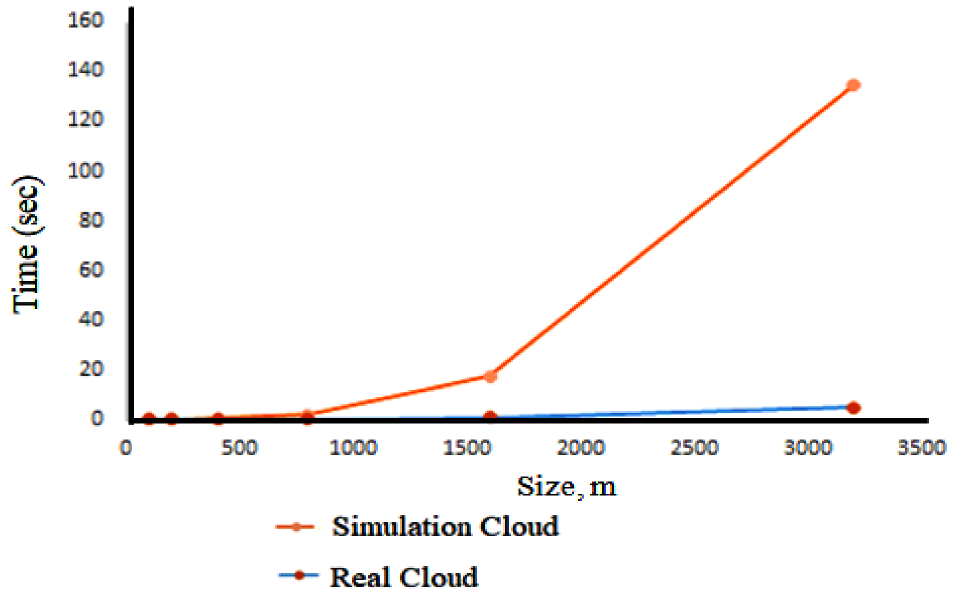

Figure 6.5: Comparison Between Real and Simulation Cloud Time of CLR

6.4 Conclusion

We investigated the problem of securely outsourcing large-scale LE and LR problem and designed a protocol for outsourcing of LE and LR value problem to public cloud. We have shown that the proposed protocol simultaneously fulfills the goals of correctness, security (input/output privacy), robust cheating resistance, and high-efficiency. It requires only one-time encryption phase.

6.4.1 Recommendation for Future Work

Cloud outsourcing is growing field. So, there is huge scope to work with this field. Different research fields can be created based on client and cloud side, Some, directions for further research include:

- Identifying new meaningful scientific and engineering computational tasks and then designing protocols to solve them.
- Adding result verification for some early protocols, which do not handle result verification, as a counter offensive to malicious cloud.
- How further reduction stage can be included in client side by fog computing can be a strong research direction.
- Normally, every outsourcing schema follows same system model and so it is the demand of time to think a different angle about cloud outsourcing basic model.

References

1. Data Growth,"https://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm", access date: 11 nov. 2018.

2. Xing Hu, Chunming Tang, "Secure Outsourced Computation of the Characteristic Polynomial and Eigenvalues of Matrix", Springer Journal of Cloud Computing, advances, systems and applications, volume 4, page 1-6, 2015.

3. Sagarika Dev Roy, Srividhya Ganesan, "Secure Outsourcing of Linear Computations into the Cloud", International Journal of Advanced Research in Computer Science and Software Engineering, volume 4, page 669-674, 2014.

4. The V's of Big Data, "https://www.xsnet.com/blog/bid/205405/the-v-s-of-big-data-velocity-volume-value-variety-and-veracity", access date: 10 Nov. 2018.

5. 5-vs-of-big-data, "https://java2blog.com/big-data-introduction/5-vs-of-big-data/", access date: 31 dec. 2018.

6. Lyapunov Equation, "https:/en.wikipedia.org/wiki/Lyapunov/equation", access date: 11 Nov. 2018.

7. Scottedward Hodel, Stephen Hung, "Solution and Applications of the Lyapunov Equation for Control Systems", IEEE Transactions on Industrial Electronics, volume 39, page 194-202, 1992.

8. Denis Valle, Sreten Cvetojevic, Ellen P. Robertson, Brian E. Reichert, Hartwig H. Hochmair and Robert J. Fletcher, "Individual Movement Strategies Revealed through Novel Clustering of Emergent Movement Patterns", Nature Scientific Reports, volume 7, page 1-12, 2017.

9. David Benjamin and Mikhail Atallah, "Private and Cheating Free Outsourcing of Algebraic Computations", IEEE Transaction on Scientific Security and Trust, volume 7, page 240-245, 2008.

10. Mikhail Atallah, Keith Frikken, "Securely Outsourcing Linear Algebra Computations", Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security - ASIACCS, page 48-59, 2010.

11. Xinyu Lei, Xiaofeng Liao, Tingwen Huang, Feno Heriniaina, "Achieving Security, Robust Cheating Resistance and High Efficiency for Outsourcing Large Matrix Multiplication Computation to a Malicious Cloud", Elsevier Journal of Information Sciences, volume 280, page 205-217, 2014.

12. Kun Jia, Hongwei Li, Dongxiao Liu and Shui Yu,"Enabling Efcient and Secure Outsourcing of Large Matrix Multiplications ", IEEE Global Communications Conference (GLOBECOM), San Diego, CA, page 1-6, 2015.

13. Shenmin Zhang, Hongwei Li, Kun Jia, Yuan-Shun Dai, Lian Zhao, "Efficient Secure Outsourcing Computation of Matrix Multiplication in Cloud Computing", IEEE Global Communications Conference (GLOBECOM), Washington, DC, page 1-6, 2016.

14. Malay Kumar, Jasraj Meena, Manu Vardhan and Robert Adams, "Privacy Preserving, Verifiable and Efficient Outsourcing Algorithm for Matrix Multiplication to a Malicious Cloud Server", Journal of Intelligent Fuzzy Systems, volume 4, page 1-20, 2017.

15. Shan Kong, Yong quan, Fei Xue,Haiyang Yu, Allah Ditta, "Cloud Outsourcing Computing Security Protocol of Matrix Multiplication Computation Based on Similarity Transformation", International Journal of Wireless and Mobile Computing, volume 14. page 90-96, 2018.

16. Xinyu Lei, Xiaofeng Liao, Tingwen Huang, Feno Heriniaina, "Achieving Security, Robust Cheating Resistance and High-efficiency for Outsourcing Large Matrix Multiplication Computation to a Malicious cloud", Elsivier Journal of Information Sciences, vol 280, page 205-217, 2013.

17. Tingwen Huang, Chunqiang Hu, Xiaofeng Liao and Xinyu Lei, "Outsourcing Large Matrix Inversion Computation to a Public Cloud", IEEE Transactions on Cloud Computing, volume 1, page 78-87, 2013.

18. Payman Mohassel, "Efficient and Secure Delegation of Linear Algebra", IACR Cryptology Eprint Archive, 2011.

19. Duan Jia, Zhou Jiantao and Li Yuanman"Secure and Verifiable Outsourcing of Nonnegative Matrix Factorization(nmf)", 4th ACM Workshop on Information Hiding and Multimedia Security, page 668, 2016.

20. Lifeng Zhou and Chunguang Li, "Outsourcing Eigen Decomposition and Singular Value Decomposition of Large Matrix to a Public Cloud", IEEE Access, volume 4, page 869-879, 2016.

21. Sergio Salinas, Luo Changqing, Weixian Liao and Pan Li, "Efficient Privacy Preserving Outsourcing of Large Scale QR Factorization", 2017 IEEE Conference on Trustcom, page 917-924, 2017.

22. Shiran Pan, Fangyu Zheng, Zhu Tao, Wang Wen and Qiong xiao, "Harnessing the Cloud for Secure and Efficient Outsourcing of Non-negative Matrix Factorization", 6th IEEE Conference on Communications and Network Security (CNS), 2018.

23. Bin Lei, Zhenhua Liu and Qi Han, "Secure and Verifiable Outsourcing Protocol for Non Negative Matrix Factorization", International Journal of High Performance Computing and Networking, volume 11,page 14-23, 2018.

24. Xinyu Lei, Xiaofeng Liao, Tingwen Huang and Huaqing Li, "Cloud Computing Service: The Case of Large Matrix Determinant Computation", IEEE Transactions on Services Computing, volume 8, page 688-700, 2015.

25. Kui Ren, Jia Wang and Qian Wang, "Harnessing the Cloud for Securely Outsourcing Large Scale Systems of Linear Equations", IEEE Transactions on Parallel and Distributed Systems, volume 24, page 1172-1181,2013.

26. FeiChen, Tao Xiang, Yuanyuan Yang, "Privacy Preserving and Verifiable Protocols for Scientific Computation Outsourcing to the Cloud", Journal of Parallel and Distributed Computation, volume 74, page 2141-2151, 2014.

27. Xiaofeng Chen, Xinyi Huang, Jin Li, Jianfeng Ma, Wenjing Lou and Duncan S. Wong, "New Algorithms for Secure Outsourcing of Large Scale Systems of Linear Equations", IEEE Transactions on Information Forensics and Security, volume 10, page 69-78, 2015.

28. Haixin Nie, Xiaofeng Chen, Jin Li, Josolph Liu and Wenjing Lou, "Efficient and Verifiable Algorithm for Secure Outsourcing of Large Scale Linear Programming", IEEE 28th International Conference on Advanced Information Networking and Applications, page 591–596, 2014.

29. Cong Wang, Kui Ren and Jia Wang, "Secure Optimization Computation Outsourcing in Cloud Computing: A Case Study of Linear Programming", IEEE Transactions on Computers, volume 65, page 216-229, 2016.

30. Lifeng Zhou and Chunguang Li, "Outsourcing Large Scale Quadratic Programming to a Public Cloud", IEEE Access, volume 3, page 2581-2589, 2015.

31. Xinyu Lei, Jian yong, Fei Chen and Tao Xiang,"Highly Efficient Linear Regression Outsourcing to a Cloud", IEEE Transactions on Cloud Computing, volume 2, page 499-508, 2014.

32. Cheng Qian, Jian Wang,Youwen Zhu and Zhikuan Wang, "On Efficiently Harnessing Cloud to Securely Solve Linear Regression and Other Matrix Operations", Elsevier Journal of Future Generation Computer Systems, volume 81, page 404-413, 2018.

33. Shailesh Tiwari, Manu Vardhan, Malay Kumar and Jasraj Meena, "Privacy Preserving, Verifiable and Efficient Outsourcing Algorithm for Regression Analysis to a Malicious Cloud", Journal of Intelligent Fuzzy Systems, volume 32, page 3413-3427, 2017.

34. Xuhui Chen, Sergio Salinas, Luo Changqing and Pan Li, "Efficient Secure Outsourcing of Large Scale Sparse Linear Systems of Equations", IEEE Transactions on Big Data, volume 4, page 26-39, 2018.

35. Mathwork Cloud, "https://www.mathworks.com/cloud.html", access date: 11 Nov. 2018.

36. Zihao Shan, Kui Ren, Marina Blanton and Cong Wang, "Practical Secure Computation Outsourcing: A Survey", ACM Computer Survey,volume 11, page 1-40, 2017.

37. Chaotic Map, "https://en.wikipedia.org/wiki/List_of_chaotic_maps", access date: 31 dec. 2018.

38. Logistic Map, "https://en.wikipedia.org/wiki/Logistic_map", access date: 31 dec. 2018.

39. Hanan Abdullah, Rasul Enayatifar, Malrey Lee, "A hybrid genetic algorithm and chaotic function model for image encryption", International Journal of Electronics and Communications, volume 66, page 806-816, 2012.

40. Frobenius Matrix, "https://en.wikipedia.org/wiki/Frobenius_matrix", access date: 31 dec. 2018.

41. Affine Transformation, "https://en.wikipedia.org/wiki/Affine_transformation", access date: 31 dec. 2018.

42. Homomorphic Encryption, "https://en.wikipedia.org/wiki/Homomorphic_encryption", access date: 31 dec. 2018.

List of Publications

1. Jannatul Ferdush and M. M. A. Hashem, "Secured cloud computing service: A case study of large Scale Lyapunov Equation," Ist International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox’s Bazar, page 339-343, 2017
2. Jannatul Ferdush and M. M. A. Hashem, "A New Secured Cloud Computing Service: A Case Study of Large Scale Linear Regression ," 4th International Conference on Electrical Engineering and Information Communication Technology (iCEEiCT), MIST, Dhaka, 2018
3. Torikul Islam, Jannatul Ferdush and M.M.A. Hashem, "Short Paper: Secured Cloud Computing Outsourcing: A Case Study of Constrained Linear Least Square Problem," 5th International Conference on Networking, Systems and Security(NSysS), BUET, Dhaka, 2018

Appendix A

A.1 Chaotic Map

Chaotic function is a function that produces signal with chaotic behaviour. Actually it is noise signal which can be reproduced if initial condition and function are known. It has following properties:

- It is sensitive to primary conditions. By this advantage, we mean that a minor change in the primary amount will cause a significant difference in subsequent measures. If we have a small change in the signal amount, the final signal will be completely different [37],[38].
- If we have the primary quantities and the drawn function, then we can reproduce the numbers.

One of the most popular chaotic function is logistic map that describes by:

$$x_{r+1} = r(x_r - 1) \tag{A.1}$$

It is the polynomial mapping of degree 2. Where x_n is a number between zero and one. r is called the value of interest that lies in $[0,4]$.

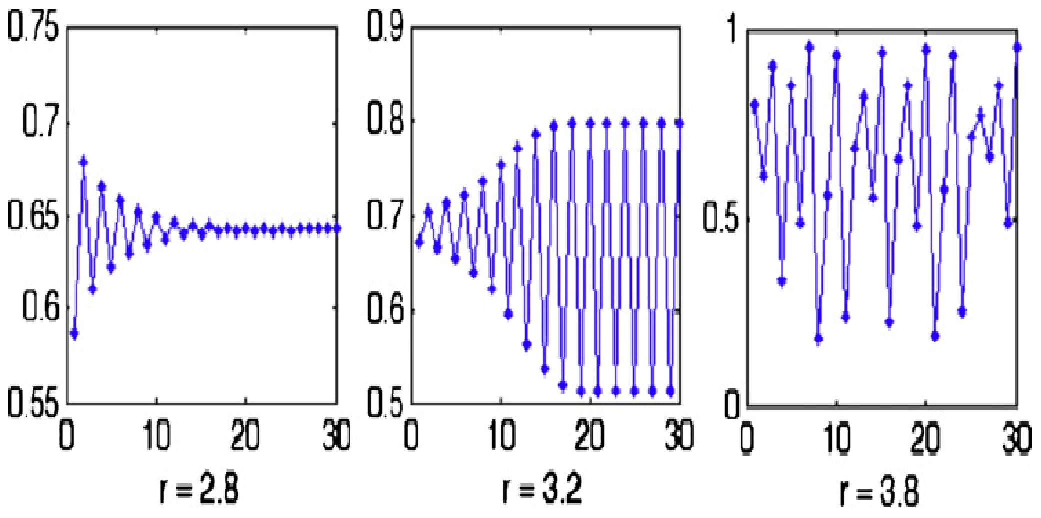

Figure A.1: Chaotic Behaviour with Different r [39]

- If $r \in [0,3]$, then the signal feature in the first 10 repetitions shows chaos, and after 10 repetitions the signal is fixed
- IF $r \in [3,3.57]$, then the signal feature in the first 20 repetition shows chaos, and after 20 repetitions the signal is fixed.
- If $r \in [3.57,4]$, then the signal feature is completely chaotic.

These behaviour are shown in figure A.1.

A.2 Frobenius Matrix

It is special type of square matrix. A matrix is said to be frobenius if it has following criteria:

- Its main diagonal entry should be one.
- Any arbitrary column elements below diagonal should be non zero.
- Other entry should be zero.

Frobenius matrix is always invertible. That means if A is frobenius matrix then $AA^{-1} = I_n$. If the all diagonal elements of frobenius matrix are changed to greater than one, it is also invertible. So, if A is diagonal frobenius matrix then $AA^{-1} = I_n$ is also true [40]. For example:

$$A = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 \\ 0 & 3 & 9 & 0 & 0 \\ 0 & 6 & 0 & 4 & 0 \\ 0 & 2 & 0 & 0 & 6 \end{bmatrix} \text{ and } \quad A^{-1} = \begin{bmatrix} \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{7} & 0 & 0 & 0 \\ 0 & -3 & \frac{1}{9} & 0 & 0 \\ 0 & -6 & 0 & \frac{1}{4} & 0 \\ 0 & -2 & 0 & 0 & \frac{1}{6} \end{bmatrix}$$

So,

$$AA^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

A.3 Affine Mapping

Affine mapping idea comes from computer graphics. Affine transformation or affine map or an affinity is a function between that preserves points, straight lines and planes in affine spaces.Here, sets of parallel lines remain parallel after an affine transformation. An affine transformation does not necessarily preserve angles between lines or distances between points, though it does preserve ratios of distances between points lying on a straight line [41].

Examples of affine transformations include translation, scaling, similarity transformation, reflection, rotation, shear mapping and compositions of them in any combination and sequence.

If X and Y are affine spaces, then every affine transformation $f\colon X \to Y$ is of the form $x \to Mx + b$, where M is a linear transformation on the space X , x is a vector in x , and b is a vector in Y.

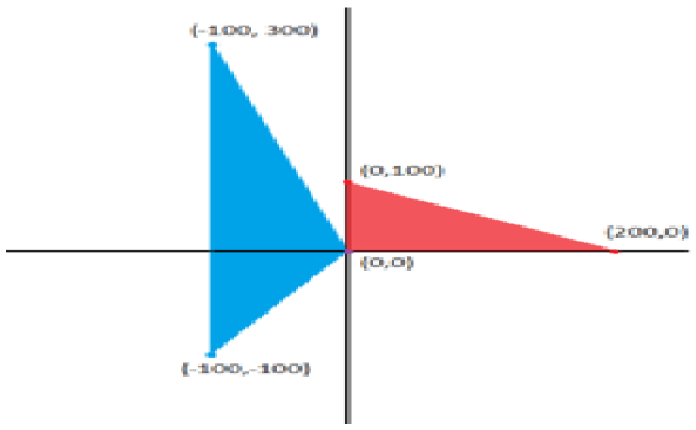

Figure A.2: Affine Transformation

Figure A.2 shows affine transformation of a triangle. Unlike a purely linear transformation, an affine map need not preserve the zero point in a linear space. Thus, every linear transformation is affine, but not every affine transformation is linear.

A.4 Homomorphic Encryption

Homomorphic encryption [42] applies where plain texts and cipher texts both are treated with an equivalent algebraic function. Homomorphic Encryption applies operation on encrypted data without knowing the original plaintext.

In cloud outsourcing, homomorphic encryption is popularly used without knowing sensitive original data. For example, services from different companies can calculate 1) the tax, 2) the currency exchange rate, and 3) shipping on a transaction without exposing the unencrypted data to each of those services. It can be used other secure systems such as secure voting systems, collision-resistant hash functions, private set intersection and private information retrieval schemes.