準同型暗号による統計解析のアウトソーシング II: 予測モデリング

川崎 将平 † 陸 文杰 † 佐久間 淳 ‡

† 筑波大学 大学院 システム情報工学研究科
305-8577 茨城県つくば市天王台 1 丁目 1-1
{kawasaki, riku}@mdl.cs.tsukuba.ac.jp
‡ 筑波大学 大学院 システム情報工学研究科 / JST CREST
305-8577 茨城県つくば市天王台 1 丁目 1-1
jun@cs.tsukuba.ac.jp

あらまし 統計分析の計算をクラウドに委託する場合，データをサーバに集約して必要な計算資源を柔軟に確保して分析を実施できる．しかし一方で，データを第三者に開示する必要があるため，データの安全性について課題がある．本稿では，統計分析として予測モデリングに焦点をあて，計算をクラウドサーバに委託する際にデータプライバシを保護することを目的とする．提案手法では，クラウドサーバにアップロードするデータを準同型暗号と呼ばれる暗号方式を利用して暗号化することで，データ通信中やクラウドでの演算中において，データプライバシの保護を実現する．

Cryptographically-secure Outsorcing of statistical Data Analysis II: Predictive Model Building

Shohei Kawasaki† Lu Wenjie† Jun Sakuma ‡

†Graduate School of SIE, University of Tsukuba.
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8577, JAPAN
{kawasaki, riku}@mdl.cs.tsukuba.ac.jp
‡Graduate School of SIE, University of Tsukuba/ JST CREST
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8577, JAPAN
jun@cs.tsukuba.ac.jp

Abstract When we outsource calculation of statistical analysis to the cloud server, we can aggregate the data at the server with appropriate computational resources. However, it is necessary for data contributor to disclose the data to a third party. Therefore, we have a problem about the safety or privacy of the data. We aim to preserve the privacy of data when we outsource the task to the cloud server. Especially, we focuses on predictive model buildings, in this manuscript. In our method, we apply a homomorphic encryption scheme to allow computation on ciphertexts.

1 Introduction

クラウド環境における秘密計算を実現することは準同型暗号の有望な応用のひとつであ

る．近年では，準同型暗号はプライバシ保護のための技術として広く認知され，Gentry らの研究 [1] をきっかけに多くの研究がされている [2, 3, 4, 5]．本研究では，[4] に提案されている完全準同型暗号をを用いて．暗号理論的に安全な統計分析のアウトソーシング (CODA: Cryptographically secure Outsourcing of statistical Data Analysis) の枠組みを提案する．CODA は記述統計量として平均，分散，共分散，最頻値，k-分位点をサポートし，予測モデリングとして主成分分析 (PCA: Principal Component Analysis) と線形回帰をサポートする．一本目 [6] では記述等計量として，平均，分散，共分散，最頻値，k- 分位点を提案した．二本目である本稿では，予測モデリングに焦点を当て，主成分分析 (PCA: Principal Component Analysis) と線形回帰のプロトコルを提案する．

予測モデリングを準同型暗号を用いて安全にアウトソースするには，いくつか問題点が存在する．ひとつは，予測モデリングに必要な標準的な計算を準同型暗号がサポートする準同型演算で実行できないという点である．そのため，実現可能な準同型演算のみを用いた反復法をもとにしたプロトコルを設計する．また，準同型演算を用いた行列演算の効率化のため，パッキングと呼ばれる手法を用いる．最後に，反復計算を用いる弊害として，実装に用いるライブラリの制限が問題となる．我々の手法は反復計算によって実現されるため，高い精度が必要であるが暗号系のライブラリである HElib では平文空間サイズが限られており，その制限の中では実用的な解が得られない．この問題を解決するため，中国剰余定理 (CRT:Chinese Remider Theorem) を用いる工夫をする．

Related Works. プライバシを保護した統計解析は大きく garbled circuit，秘密分散，準同型暗号の三種類に分けられる．

Garbled circuits は回路として表現できる任意の関数をプライベートに評価できる．この概念は Yao[7] によって初めて提案された．[8] では Garbled circuit を用いてプライバシーを保護し，線形回帰を計算する方法を提案した．彼らの手法では garbled circuit を生成，評価するため回路の深さに飛来する回数の通信が発生する．アウトソーシングにおいては，完全に非対話的な計算が望ましい．我々の提案手法は非対話的な計算を実現しており，アウトソースにより適している．

秘密分散はお互いに共謀しない三つ以上のいサーバが必要である．Amazon EC2 などの一般的な public cloud ではそのようなサーバを確保することは容易でなく，制約が大きい．そのため，単一のサーバにすべての計算を委託できる方式が望ましい．

準同型暗号を用いた先行研究は大きく二つ挙げられる．[9] は Cox 比例モデルとロジスティック回帰モデルを somewhat 準同型暗号上で実現した．しかし，モデル学習については議論されておらず，予測のみを対象としている．一般に予測よりもモデル学習のほうが実現が難しく，我々の研究はモデル学習を対象としている．[10] は線形分類モデルと Fisher の分類モデルのモデル学習を暗号上で実現した．彼らは線形回帰，主成分分析，スペクトラルクラスタリングについても実現可能であるとしているが，[10] 中ではアルゴリズムや実験結果は示されていない．

Contributions. 本研究では，暗号理論的に安全な統計解析のアウトソーシングの枠組み CODA を提案する．本稿ではその中で PCA と線形回帰のプロトコルを提案する．我々の提案手法である PCA と線形回帰のプロトコルは非対話的なプロトコルである．

我々の提案法では多項式の CRT をもとにしたパッキング手法 [11] を用いて，ベクトルをひとつの暗号文に暗号化する．その上で，行列積などの行列演算を暗号上の準同型演算を用いて実現する方法を導入する．これにより，準同型演算を用いた大規模な行列演算を効率化している．

また PCA と線形回帰において，出力値の任意の精度をサポートするために整数上の CRT を利用する．この方法により，暗号スキームの平文空間が制限されていても，任意の精度を実現できる．

最後に，PCA と線形回帰のプロトコルに関して出力値の精度と計算時間を評価する実験をおこない，実用的な計算時間と解の精度を実現

できることを示した．

2 (Leveled) Fully Homomorphic Encryption

完全準同型暗号 (FHE:Fully Homomorphic Encryption) は，暗号文を復号せずに暗号化された数値に対する加算･乗算の演算をおこなうことができる．我々は，FHE を用いることで統計分析計算のアウトソーシングにおいて，データプライバシの保護を実現する．本稿では，[4] に提案されている Brakerski–Gentry–Vaikuntanathan スキーム (以下，BGV スキーム) を利用する．また，BGV スキームには HElib [12, 13] と呼ばれるオープンソースのライブラリが存在する．本研究の実験では，HElib を用いて実装をおこなう．BGV スキームと HElib の詳細については本研究に関する一本目の予稿 [6] に記載されているため，本稿では割愛する．

2.1 Limitations of Precision

HElib はオープンソースによる FHE の実装であり，我々の知るかぎりでは唯一の実用的な公開ライブラリである．しかしながら，大規模データを用いた統計解析に適用するにはいくつかの制限がある，

BGV スキームは整数上で定義されているため，実数値を直接扱うことはできない．我々の提案では，実数 $x \in \mathbb{R}$ を BGV スキーム上で扱うとき，$\tilde{x} = \lfloor Mx \rceil \in \mathbb{Z}$ とし，x の代わりに $\tilde{x}$ を BGV スキーム上で扱う．ここで $\lfloor \cdot \rceil$ は，入力実数に最も近い整数を返す丸め関数である．

上記の表現を用いると，拡大係数の影響は計算を反復する度に指数的に増大してしまう．HElib の現在のバージョンでは内部で用いている NTL ライブラリ [14] の実装上の制限により，60-bit の平文空間しかサポートしていない．そのため，PCA や線形回帰の計算の秘密計算をおこなうためには平文空間が足りず，実用的な精度を達成できない．

我々は整数上の CRT を用いてこの問題を解決する．CRT を利用することで，複数の法のもとで同様のプロトコルを実行し，それらの結果を結合することでより大きな法のもとでの計算結果を一意に復元することができる．この詳細については 6.2 節で述べる．

3 Notation

本節では，本稿で用いる記号を定義する．行列は大文字のボールド体で表し (e.g., $\boldsymbol{A}$)，$\boldsymbol{a}_i^{\mathrm{T}}$ は行列 $\boldsymbol{A}$ の i 行目のベクトルを表す．本稿で用いるベクトルはすべて列ベクトルであるとし，列ベクトルは転置記号を用いて表現する (e.g.,$\boldsymbol{v}^{\mathrm{T}}$)．行列積および行列とベクトルの積は，$\boldsymbol{XY}$，$\boldsymbol{Xa}$ と表す．

暗号文は対応する平文の記号のゴシック体で表現する (e.g., $\mathfrak{x}$ は x の暗号文)．暗号文に対する準同型加算を $\oplus$，準同型乗算を $\odot$ で表す．また，暗号文パラメータを m, t, L とする．m, t は平文空間 $\mathbb{A}_t$ を決定し，L はレベルを表す，

CRT パッキングは整数ベクトルを入力として，それぞれの要素をスロットに持つ多項式に変換する関数 $\mathcal{E}_{\mathrm{crt}} : \mathbb{Z}_t^{\ell} \to \mathbb{A}_t$ と定義する．

$\mathcal{E}_{\mathrm{crt}}(\cdot)$ が行列 $\boldsymbol{X} \in \mathbb{Z}^{n \times \ell}$ を入力とする際には，$\boldsymbol{X}$ の各列ベクトルを CRT パッキングしていることを表す (i.e., $\mathcal{E}_{\mathrm{crt}}(\boldsymbol{X}) = [\mathcal{E}_{\mathrm{crt}}(\boldsymbol{x}_1), \mathcal{E}_{\mathrm{crt}}(\boldsymbol{x}_2), \cdots]$)．BGV スキームの暗号化関数および復号関数は Enc(·),Dec(·) とする．ここで，CRT パッキングを用いたベクトル $\boldsymbol{x}$ の暗号化を $\mathrm{Enc}(\boldsymbol{x})$ と表記し，$\mathrm{Enc}(\boldsymbol{X})$ は $\boldsymbol{X}$ の各列ベクトルを CRT パッキングを用いて暗号化したものとする．同様に，CRT パッキングを用いた暗号文 $\mathfrak{x}$ を復号する際は $\mathcal{E}_{\mathrm{crt}}^{-1}(\mathrm{Dec}(\mathfrak{x}))$ の代わりに $\mathrm{Dec}(\mathfrak{x})$ と表記する．

4 Problem Statement

我々の目的は，プライベートに PCA と線形回帰を計算するプロトコルを設計することである．本節では，PCA と線形回帰を導入し，セキュリティモデルを定義する．

PCA: PCA は観測した高次元データを分散が最大となるような低次元空間へ変換する手法である．PCA の問題は観測データ $\boldsymbol{X} \in \mathbb{Z}^{n \times \ell}$ の分散共分散行列 $\boldsymbol{\Sigma} = \frac{1}{n}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} - \boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}$ の固有値問題 $\boldsymbol{\Sigma}\boldsymbol{u} = \lambda\boldsymbol{u}$ に帰着される．ここで，$\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i$ である．固有ベクトル $[\boldsymbol{u}_1, \cdots, \boldsymbol{u}_i, \cdots, \boldsymbol{u}_{\boldsymbol{\ell}}]$ を $[\lambda_1 \geq \cdots \geq \lambda_i \geq \cdots \geq \lambda_\ell]$ に対応する固有ベクトルとする．このとき，$\boldsymbol{u}_i$ は第 i 主成分と

呼ばれ，i 番目に分散が大きくなる方向を表している．一般に，PCA では第一主成分に最も興味がある．

Linear Regression: 線形回帰問題は，観測された入力変数から目的変数を予測する線形モデルを見つけることを目的とする．$\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ を観測され入力変数と目的変数の事例とすると，線形回帰のモデルは $y \approx \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ となる．ここで，$\boldsymbol{w}$ は回帰モデルのパラメータである．回帰モデルのパラメータの最適値 $\boldsymbol{w}^*$ は以下の二乗誤差最小化問題を解くことで得られる．

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} \|y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i\|_2^2 \quad (1)$$

式 (1) の解析解は式 (2) で計算することができる，

$$\boldsymbol{w}^* = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} \quad (2)$$

ここで，$\boldsymbol{X} := [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^{\mathrm{T}}$，　$\boldsymbol{y} := [y_1, \cdots, y_n]^{\mathrm{T}}$ である．

Security Model: 本研究では，3 人の登場人物 data contributors, cloud, analyst の間で統計分析のアウトソーシングを定義する．

analyst は data contributor から集めたデータを用いて統計分析を実施したいパーティである．data contributor は分析のためにデータを提供するが，プライベートなデータを他のどのパーティからも隠したいという要望がある．そのため，data contributor は analyst が生成した公開鍵を用いて自身のデータを暗号化してから cloud に暗号文を送信する．cloud は data contributor から受け取ったデータから統計分析の計算をし，analyst に結果を送信する．このモデルにおけるセキュリティは次の二点である．

1. cloud は data contributor から集めたプライベートなデータに関して何も学ばない．
2. analyst は統計分析の結果以外に data contributor のプライベートなデータに関して何も学ばない．

本研究では，analyst が cloud に委託する統計分析のクエリに関してはプライバシを考えず，cloud と analyst は semi-honest なパーティとする．data contributor は自身のデータを暗号化して cloud にアップロードした後にオフラインになるとする．そのため，data contributor の振る舞いはセキュリティには関与しない．また，すべての通信路は安全であるとする (e.g., SSL や PKI など)．そのため通信路の傍受等の攻撃については考慮しない．

5 Matrix Operation

本節では，後に示す提案プロトコルで用いる暗号文上の行列演算について導入する．正方行列 $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{Z}_t^{\ell\times\ell}$ とし $\boldsymbol{u} \in \mathbb{Z}_t^{\ell}$ の暗号文を $\mathfrak{u}$ とする．ここで，$\boldsymbol{X}, \boldsymbol{Y}$ の i 行目のベクトルをそれぞれ $\boldsymbol{x}_i^{\mathrm{T}}$，$\boldsymbol{y}_i^{\mathrm{T}}$ とし，その暗号文を $\mathfrak{x}_i$，　$\mathfrak{y}_i$ とする．

Scalar Multiplication: $s\boldsymbol{X}$．暗号化された行列のスカラ倍は単純にすべての i に対して $s \otimes \mathfrak{x}_i$ を計算することで実現可能である．

Matrix–matrix Addition: $\boldsymbol{X} + \boldsymbol{Y}$．暗号化された行列同士の和の計算はそれぞれの列ベクトルの暗号文の準同型加算で実現できるすなわち，すべての i に対して $\mathfrak{x}_i \oplus \mathfrak{y}_i$ を計算すればよい．行列同士の減算も同様にして実現できる．

Matrix–vector Multiplication: $\boldsymbol{X}\boldsymbol{u}$．Halevi らは暗号化された対称行列とベクトルの積の計算手法を [13] 中で提案している．我々の提案の中で，ベクトルと行列の積の計算は対称行列しか扱わないため，この計算方法を利用する．暗号化された対称行列とベクトルの積は以下のように計算できる．

$$\mathrm{Enc}(\boldsymbol{X}\boldsymbol{u}) = \bigoplus_{i=1}^{\ell} \{\mathfrak{x}_i \odot \mathrm{replicate}(\mathfrak{u}, i)\}$$

ここで replicate，すべての要素が入力されたベクトルの暗号文の指定されたインデックスの要素であるベクトルの暗号文を返す．例えば，$\mathfrak{a}$ を $\boldsymbol{a} := [a_1, \cdots, a_\ell]$ の暗号文とすると．$\mathrm{replicate}(\mathfrak{a}, 1)$ は $[a_1, \cdots, a_1]$ の暗号文を返す．この関数は HElib で提供されている．

Type I Matrix–matrix Multiplication: $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}$．$\mathfrak{x}'_j$ を $\mathrm{Enc}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y})$ の j 行目のベクトルの暗号文とすると，$\mathfrak{x}'_j$ は以下のように計算できる．

$$\mathfrak{x}'_j = \bigoplus_{i=1}^{\ell} \{\mathfrak{x}_i \odot \mathrm{replicate}(\mathfrak{y}_i, j)\}$$

表 1: CRT パッキングを用いた暗号文上の行列・ベクトル演算の complexity.

Operation	Add.	Mult.	Replicate
$s\boldsymbol{X}$	-	$O(\ell)$	-
$\boldsymbol{X}+\boldsymbol{Y}$	$O(\ell)$	-	-
$\boldsymbol{X}\boldsymbol{u}$	$O(\ell)$	$O(\ell)$	$O(\ell)$
$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}$	$O(\ell^2)$	$O(\ell^2)$	$O(\ell^2)$
$\boldsymbol{X}\boldsymbol{Y}$	$O(\ell^2)$	$O(\ell^2)$	$O(\ell^2)$

Type II Matrix–matrix Multiplication: $\boldsymbol{X}\boldsymbol{Y}$. 暗号化された行列に対して，転置の操作を適用してから Type I の行列積を計算するのは，要素のレイアウト変更を含むため計算コストが高い．そのため，$\boldsymbol{X}\boldsymbol{Y}$ を直接計算する方法を導入する．Type I と同様に，$\mathfrak{x}'_j$ を $\mathrm{Enc}(\boldsymbol{X}\boldsymbol{Y})$ の j 行目のベクトルの暗号文とすると，$\mathfrak{x}'_j$ は以下のように計算できる．

$$\mathfrak{x}'_j = \bigoplus_{i=1}^{\ell} \{\mathrm{replicate}(\mathfrak{x}_j, i) \odot \mathfrak{y}_i\},$$

表 1 に各演算の complexity を示す．

6 PCA

前に述べたように，PCA の問題は固有値問題に帰着できる．しかしながら，暗号文上の準同型演算だけで固有値分解を実現することは難しい．そのため，より簡単な power method を採用する．このアルゴリズムは加算と乗算のみで構成される計算式を反復することで最大固有値と対応する固有ベクトルを求めることができる．

$\boldsymbol{X}$ を与えられた観測データとし，その分散共分散行列を $\boldsymbol{\Sigma}$ とする．また，$\boldsymbol{\Sigma}$ の固有値を $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$ とする．観測データの第一主成分は $\boldsymbol{\Sigma}$ の最大固有値 λ_1 に対応する固有ベクトル $\boldsymbol{u}_1$ である．このとき，$\boldsymbol{\Sigma}$ の最大固有値と対応する固有ベクトルは以下に示す power method によって求めることができる．

1. ランダムに $\boldsymbol{u}^{(0)} \in \mathbb{Z}^{\ell}$ を生成する
2. 以下の式を $\tau = 0$ から $T-1$ まで繰り返す

$$\boldsymbol{u}^{(\tau+1)} = \boldsymbol{\Sigma}\boldsymbol{u}^{(\tau)}$$

Σ の最大固有値 λ_1 は $\lambda_1 = \|\boldsymbol{u}^{(T)}\|/\|\boldsymbol{u}^{(T-1)}\|$ であり，対応する固有ベクトルは $\boldsymbol{u}_1 = \boldsymbol{u}^{(T)}/\|\boldsymbol{u}^{(T)}\|$ となる．power method は一次収束し，その収束率は $O\left(|\lambda_2/\lambda_1|^T\right)$ である，

Algorithm 1 PCA Protocol.

- **Input of the i-th data contributor:** $\boldsymbol{x}_i$
- **Output of the analyst:** the principal component $\boldsymbol{u}$ and the associated eigenvalue λ_1.

1: **Upload:** The i-th contributor locally computes the matrix $\boldsymbol{\Sigma}_i := \lfloor M^2 \boldsymbol{x}_i \boldsymbol{x}_i^{\mathrm{T}} \rceil$ and submits the encryption $\mathrm{Enc}(\boldsymbol{\Sigma}_i)$ to the cloud, where $\lfloor \cdot \rceil$ is the rounding function.
2: **Evaluation:** The cloud randomly chooses a vector $\boldsymbol{u}^{(0)}$ from $\mathbb{Z}_t^{d_n}$.
3: The cloud joins the collected ciphertexts as $\mathrm{Enc}(\boldsymbol{\Sigma}) := \sum_{i=1}^{n} \mathrm{Enc}(\boldsymbol{\Sigma}_i)$.
4: For $0 \leq \tau < T$, the cloud evaluates with the matrix-vector multiplication primitive.

$$\mathrm{Enc}(\boldsymbol{u}^{(\tau+1)}) = \mathrm{Enc}(\boldsymbol{\Sigma})\mathrm{Enc}(\boldsymbol{u}^{(\tau)}).$$

5: **Download:** The analyst downloads two ciphertexts $\mathrm{Enc}(\boldsymbol{u}^{(T)})$ and $\mathrm{Enc}(\boldsymbol{u}^{(T-1)})$ from the cloud, decrypts them, and outputs $\boldsymbol{u}$ and λ_1.

6.1 PCA Protocol

前節に示した power method を用いてプライベートに PCA を計算するプロトコルを Algorithm 1 に示す．2 節で示したとおり，BGV スキームでは整数のみを扱う．そのため BGV スキーム上では，実数値に拡大係数 M をかけ，丸め関数 $\lfloor \cdot \rceil$ によって整数化した値を扱う．このプロトコルにおいて，拡大係数 M を十分大きくすれば丸め関数の影響は無視できる．しかしながら，2.1 節で述べたように現在の HElib の実装では 60-bit までの平文空間しか扱えないことが問題となる．

仮に，$\ell \times \ell$ の分散共分散行列 $\boldsymbol{\Sigma}$ の最大要素が B であるとし，power method の反復回数を T とする．このとき，Algorithm 1 の出力値の上限は $(M\ell)^T B^{T+1}$ となる．$\ell = 5$，$B = 3$ とすると (i.e., 5 次元データを 3 桁の精度で扱う)，三回目の反復後の出力の上限は $M^3\ell^3B^4 \approx 2^{77}$ となり，HElib がサポートする 60-bit を越える．このように，60-bit の制限のもとでは T と M の両方を十分に大きくとることができない．我々は，整数上の CRT を用いてこの問題を解決する．

6.2 Larger Plaintext Precision

前節で述べたように，本研究では 60-bit 以上の値を扱うために，CRT を用いる工夫をする．具体的には，K 個の相異なる素数 t_k $(k = 1, \cdots, K)$ を用いて平文空間の法 t を $t = \prod_{k=1}^{K} t_k$ とする．そして，$\mathbb{A}_t$ 上でプロトコルを実行する代わりに，K 個の多項式環 $\mathbb{A}_{t_k}$ 上でプロトコルを実行する．最後に，t_1 から t_K を法とした K 個の計算結果から $\mathbb{A}_t$ 上の値を CRT の性質を用いて求める．CRT から，$t_k \quad k = 1, \cdots, K$ は互いに素であるため，K 個の計算結果から一意に $\mathbb{A}_t$ 上の値を得ることができる．この方法により，t_k の大きさと K を変えることで任意の精度を扱うことができる．一方で，計算・通信のコストは K の大きさに比例して大きくなる．

以上の技術を PCA に適用すると，次のような手順となる．また，CRT を利用しても提案手法は非対話性を実現できる．

1. analyst は t_k $(k = 1, \cdots, K)$ を平文の法とする鍵ペア $(\mathbf{pk}_k, \mathbf{sk}_k)$ を生成する
2. data contributor は K 個の公開鍵 $\mathbf{pk}_k$ を用いて自身のデータをそれぞれ暗号化し，cloud に送信する
3. cloud ではそれぞれ異なる鍵で暗号化された暗号文を入力とし，K 回 PCA プロトコルを実行し，K 個の計算結果の暗号文を analyst に送信する
4. analyst は K 個の計算結果をそれぞれ復号し，それらを用いて法 t のもとの計算結果を得る

7 Linear Regression

観測された入力変数を $\boldsymbol{X}$，目的変数を $\boldsymbol{y}$ とすると，線形回帰の最適なパラメータは $\boldsymbol{w}^* = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$ と計算できる．暗号文上の行列演算は 5 節に述べたように，既に計算ができるため，$(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}$ を暗号文上で評価できれば線形回帰の解を暗号上で求めることが可能である．本節では，はじめに Newton 法を用いて逆行列を計算する方法について述べる．この手法は，行列の加算・乗算のみで実現できるため，我々の設定に適している．そして，線形回帰のプロトコルを示す．

Algorithm 2 Matrix Inversion Protocol.

- **Input:** An encrypted matrix, Enc($\boldsymbol{Q}$), a ciphertext Enc(a)
- **Output:** An encrypted matrix Enc($\boldsymbol{R}^{(T)}$)

1: Initialize $\mathfrak{a}^{(0)}$, Enc($R^{(0)}$) and Enc($A^{(0)}$) as

$$\mathfrak{a}^{(0)} := \mathrm{Enc}(a)$$
$$\mathrm{Enc}(\boldsymbol{R}^{(0)}) := \mathrm{Enc}(\boldsymbol{I}) \qquad \mathrm{Enc}(\boldsymbol{A}^{(0)}) := \mathrm{Enc}(\boldsymbol{Q}).$$

2: For $0 \le \tau < T$, the iteratively evaluate

$$\mathrm{Enc}(\boldsymbol{R}^{(\tau+1)}) = \mathfrak{a}^{(\tau)}\mathrm{Enc}(\boldsymbol{R}^{(\tau)}) - \mathrm{Enc}(\boldsymbol{R}^{(\tau)})\mathrm{Enc}(\boldsymbol{A}^{(\tau)})$$
$$\mathrm{Enc}(\boldsymbol{A}^{(\tau+1)}) = \mathfrak{a}^{(\tau)}\mathrm{Enc}(\boldsymbol{A}^{(\tau)}) - \mathrm{Enc}(\boldsymbol{A}^{(\tau)})\mathrm{Enc}(\boldsymbol{A}^{(\tau)})$$
$$\mathfrak{a}^{(\tau+1)} = \mathfrak{a}^{(\tau)} \odot \mathfrak{a}^{(\tau)}$$

3: Output the ciphertext of the matrix Enc($\boldsymbol{R}^{(T)}$).

7.1 Matrix inversion by Newton iteration

正定値行列 $\boldsymbol{Q}$ が与えられたとき，Newton 法によって反復的に $\boldsymbol{Q}^{-1/p}$ を求める手法が Guo らによって提案されている [15]．我々は，逆行列を計算したいため特に $p = 1$ のときに着目する．[15] で提案されている反復式を式変形した反復式を式 (3) に示す．

$$\begin{aligned} \boldsymbol{R}^{(\tau+1)} &= 2\alpha^{(\tau)}\boldsymbol{R}^{(\tau)} - \boldsymbol{R}^{(\tau)}\boldsymbol{A}^{(\tau)}, \quad \boldsymbol{R}^{(0)} = \boldsymbol{I} \\ \boldsymbol{A}^{(\tau+1)} &= 2\alpha^{(\tau)}\boldsymbol{A}^{(\tau)} - \boldsymbol{A}^{(\tau)}\boldsymbol{A}^{(\tau)}, \quad \boldsymbol{A}^{(0)} = \boldsymbol{Q} \\ \alpha^{(\tau+1)} &= \alpha^{(\tau)}\alpha^{(\tau)}, \quad \alpha^{(0)} = \lfloor \alpha \rceil, \end{aligned} \tag{3}$$

ここで，$\boldsymbol{I}$ は単位行列であり，α はある実数である．Guo らは，α を $\boldsymbol{Q}$ の最大固有値とすれば，式 (3) による更新で $\boldsymbol{R}^{(\tau)}$ は $\lfloor \alpha \rceil^{2^\tau}\boldsymbol{Q}^{-1}$ に二次的に収束することを示している．式 (3) をもとに，暗号上で逆行列を計算するプロトコルを Algorithm 2 に示す．Algorithm 2 は暗号化された行列 Enc($\boldsymbol{Q}$) とスカラ値の暗号文 Enc(a) を入力にとる．Enc(a) は式 (3) における α に対応する．

7.2 Linear Regression Protocol

線形回帰を計算するプロトコルを Algorithm 3 に示す．Algorithm 3 では，サブプロトコルとして Algorithm 2 を呼び出し，逆行列を計算する．Algorithm 3 中で analyt の入力である Enc(a) は，式 (3) の α にあたるため，適切な値を選ばなければならない．α を $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ の最大

Algorithm 3 Linear Regression Protocol.

- **Input of the i-th data contributor:** $(\boldsymbol{x}_i, y_i)$
- **Input of the analyst:** a ciphertext $\mathrm{Enc}(a)$,
- **Output of the analyst:** $\tilde{\boldsymbol{w}} \approx (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}$.

1: **Upload**: First, the i-th data contributor locally computes the matrix $\boldsymbol{A}_i := \lfloor M^2 \boldsymbol{x}_i \boldsymbol{x}_i^{\mathrm{T}} \rceil$ and the vector $\boldsymbol{b}_i := \lfloor M^2 y_i \cdot \boldsymbol{x}_i \rceil$. Then he submits ciphertexts $\mathrm{Enc}(\boldsymbol{A}_i)$ and $\mathrm{Enc}(\boldsymbol{b}_i)$ to the cloud.

2: **Evaluation:** The cloud joins the collected ciphertexts with the matrix–matrix addition primitive and homomorphic addition.

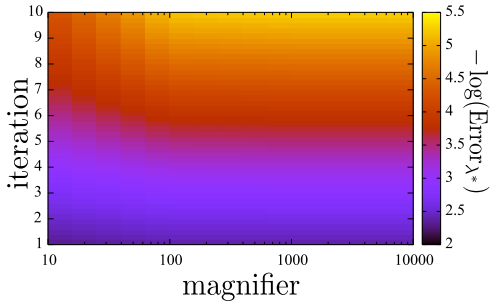
$$\mathrm{Enc}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}) = \sum_{i=1}^{n} \mathrm{Enc}(\boldsymbol{A}_i), \quad \mathrm{Enc}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}) = \bigoplus_{i=1}^{n} \mathrm{Enc}(\boldsymbol{b}_i).$$

3: The cloud then calls Algorithm 2 with inputs $\mathrm{Enc}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})$ and $\mathrm{Enc}(a)$ and obtains the result matrix $\mathrm{Enc}(\boldsymbol{R}^{(T)})$.

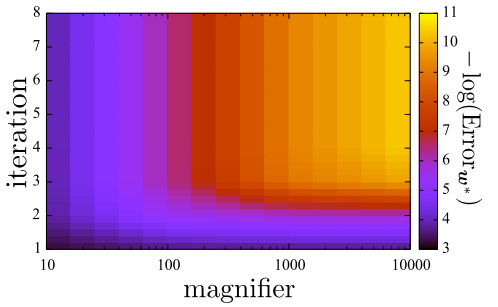
4: Finally, the cloud evaluates the following equation with the matrix–vector multiplication primitive.

$$\mathfrak{w} := \mathrm{Enc}(\boldsymbol{R}^{(T)})\mathrm{Enc}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y}).$$

5: **Download**: The analyst downloads a ciphertext $\mathfrak{w}$ from the cloud, decrypts it to obtain $a^{2^T}\tilde{\boldsymbol{w}}$, then he divides it by a^{2^T} and outputs $\tilde{\boldsymbol{w}}$.



(a) Tradeoffs of $\mathrm{Error}_{\lambda^*}$



(b) Tradeoffs of $\mathrm{Error}_{\boldsymbol{w}^*}$

図 1: PCA，線形回帰プロトコルにおける拡大係数，反復回数，誤差のトレードオフ.

固有値で与えることで，二次収束が保証されることから，α はあらかじめ PCA プロトコルにより求めておく．Algorithm 3 のサブプロトコルとして Algorithm 1 を呼ぶことができる．線形回帰プロトコルも PCA プロトコルと同様に，CRT を用いて平文空間サイズの制限の問題を解決することができる．

8 Experiment

8.1 Experiment Settings

Data set. 本実験では UCI repository [?] の Adult Data Set を標準化して用いた．このデータセットには 6 次元の数値属性の 32561 件のレコードが含まれている．これらのデータは BGV スキーム上で扱うために，拡大係数 M をかけて整数に丸めて取り扱う．

Running Environment. 本実験は Xeon 2.60 GHz，32G RAM の計算機上でおこなった．プログラムは HElib を利用して C++で実装し，8 並列で実行した．

Parameter choosing. 本実験では，BGV スキームのパラメータを 80-bit セキュリティを満たすように パラメータを設定し，それぞれ $t_k \approx 2^{36}, L = 32, m = 27893, \ell = 36$ とした．

8.2 Accuracy

本節では，プロトコル中の反復回数 T，拡大係数 M とプロトコルの出力の誤差を評価する．PCA，線形回帰プロトコルの出力の誤差はそれぞれ以下に定義する λ^*，$\boldsymbol{w}^*$ で評価する．

$$\mathrm{Error}_{\lambda^*} = \frac{|\lambda^* - \hat{\lambda}|}{\lambda^*} \qquad \mathrm{Error}_{\boldsymbol{w}^*} = \frac{\|\boldsymbol{w}^* - \hat{\boldsymbol{w}}\|_2}{\|\boldsymbol{w}^*\|_2}.$$

反復回数 T，拡大係数 M とプロトコルの出力の誤差を評価するために，M,T をそれぞれ変化させながら，平文上でプロトコルと同様の操作をおこない，λ^*，$\boldsymbol{w}^*$ を観察する実験をおこなった．PCA，線形回帰に関する実験結果をそれぞれ図 1(a), 1(b) に示す．図 1(a), 1(b) において，横軸・縦軸はそれぞれ拡大係数・反復回数を示し，カラーマップは誤差の負の対数を示す．例えば，PCA において analyst が 10^{-2} の精度を得たい場合には，拡大係数を $M = 100$ に設定し，6 回反復する必要があることがわかる．

表 2: PCA，線形回帰プロトコルの計算時間 [s](5 回の実行の平均)．M は拡大係数，K は平文空間の法のための素数の数，T は反復回数．

(a) Principle Component Analysis

	$M = 100$			$M = 1000$		
T	K	eval.	download	K	eval.	download
3	3	71.0	1.25	3	70.0	1.23
4	4	103	1.22	4	110	1.23
5	4	141	1.18	5	152	1.25

(b) Linear Regression

	$M = 100$			$M = 1000$		
T	K	eval.	download	K	eval.	download
1	2	161	0.386	2	169	0.382
2	4	386	0.611	4	383	0.604
3	7	788	0.956	8	872	0.966

8.3 Efficiency

表 2(a) に PCA の計算時間の実験結果を示す．表中で，eval. は cloud でのプロトコルの実行時間を示し，download は analyst が cloud から計算結果を復号する際の計算時間を示す．図 1(a) と表 2(a) の結果から，例えば analyst が 10^{-2} の精度で PCA の結果を得たいとき，$M = 1000, T = 5$ とすると，およそ 3 分の計算時間を要することがわかる．

Table 2(b) は線形回帰に関する同様の実験結果を示している．線形回帰では $\mathrm{Error}_{\boldsymbol{w}^*} = 10^{-5}$ を達成するために，$M = 1000, T = 3$ とすると約 15 分の計算時間となる．7 節で述べたように．LR プロトコルを実行するには PCA プロトコルを呼び出す必要があるが，PCA プロトコルの計算時間を含めてもおよそ 18 分程度で計算できる．

9 Conclusion

本稿では．PCA と線形回帰をアウトソーシングの枠組みでデータプライバシを保護して計算するプロトコルを提案した．また，実データを用いた実験により，大規模データに対しても現実的な計算時間と精度を実現できることを示した．

謝辞

本研究は，JST CREST「ビッグデータ統合利活用のための次世代基盤技術の創出・体系化」領域におけるプロジェクトおよび科学研究費 24680015 の助成を受けました．

参考文献

[1] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[2] Nigel P Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography–PKC 2010*, pages 420–443. Springer, 2010.

[3] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Advances in Cryptology–CRYPTO 2011*, pages 505–524. Springer, 2011.

[4] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 309–325. ACM, 2012.

[5] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM CCSW 2011*, pages 113–124. ACM, 2011.

[6] Lu Wenjie, Shohei Kawasakia, and Jun Sakuma. Cryptographically-secure outsorcing of statistical data analysis i: Descriptive statistics. *Computer Security Symposium 2015 (CSS2015)*.

[7] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE, 1986.

[8] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 334–348. IEEE, 2013.

[9] Joppe W Bos, Kristin Lauter, and Michael Naehrig. Private predictive analysis on encrypted medical data. *Journal of biomedical informatics*, 50:234–243, 2014.

[10] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In *Information Security and Cryptology–ICISC 2012*, pages 1–21. Springer, 2013.

[11] Nigel P Smart and Frederik Vercauteren. Fully homomorphic SIMD operations. *Designs, codes and cryptography*, 71(1):57–81, 2014.

[12] Victor Shoup Shai Halevi. HELib. `http://shaih.github.io/HElib/index.html`. Accessed: 2014-12-10.

[13] Shai Halevi and Victor Shoup. Algorithms in helib. In *Advances in Cryptology–CRYPTO 2014*, pages 554–571. Springer, 2014.

[14] Victor Shoup. NTL, number theory library. `http://www.shoup.net/ntl/`. Accessed: 2015-8-03.

[15] Chun-Hua Guo and Nicholas J Higham. A schur-newton method for the matrix p th root and its inverse. *SIAM Journal on Matrix Analysis and Applications*, 28(3):788–804, 2006.