Secure Relative Detection in (Forensic) Database with Homomorphic Encryption

Jingwei Chen[1,2], Weijie Miao[1,2], Wenyuan Wu[1,2(✉)], Linhan Yang[1,3], and Haonan Yuan[1,2]

[1] Chongqing Key Laboratory of Secure Computing for Biology, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China
{chenjingwei,miaoweijie,wuwenyuan,yuanhaonan}@cigit.ac.cn
[2] Chongqing School, University of Chinese Academy of Sciences, Chongqing, China
[3] School of Information Science and Engineering, Chongqing Jiaotong University, Chongqing, China
linhanyang@mails.cqjtu.edu.cn

Abstract. Although relative (kinship) detection has important applications in biological research, forensic identification, and many other fields, the privacy of genotype data used in the process is often overlooked. Homomorphic encryption allows for computing on encrypted genotype data directly without decryption, making it particularly suitable for privacy-preserving relative detection. Therefore, it became the competition topic for iDASH-2023 Track 1. However, combining existing kinship estimation with homomorphic encryption has two challenges: the high-dimensional matrix multiplication over encrypted data and the more time-consuming comparison over encrypted data. In this paper, we propose a secure relative detection protocol that uses homomorphic encryption to estimate the kinship between samples from two parties while protecting data privacy. We devise two new kinship estimation methods avoiding ciphertext comparisons while reducing matrix multiplication to matrix-vector multiplication. Additionally, we convert high-dimensional matrix-vector multiplication to multiple small-dimensional matrix-vector multiplications using binary dividing, which can then be processed with Halevi and Shoup's algorithm. We test the accuracy and efficiency of the protocol on the iDASH-2023 dataset. Experimental results indicate that the presented protocol outperforms existing methods with similar setups.

Keywords: Relative detection · Privacy-preserving computation · Homomorphic encryption · iDASH

1 Introduction

Relative detection (or kinship detection) via genetic databases uses DNA profiles in a genetic database to identify potential biological relatives of an unknown individual who contributed a query DNA sample. It relies on the principle that bio-

W. Peng et al. (Eds.): ISBRA 2024, LNBI 14955, pp. 410–422, 2024.
https://doi.org/10.1007/978-981-97-5131-0_35

logically related individuals share more genetic similarities than unrelated individuals. Close relatives like parents, siblings, and children share long stretches of identical DNA. The applications of relative detection include forensic investigations [20], ancient DNA and archeological studies [19], family DNA searching [10], population genetics and association studies [22], etc. However, some concerns allowing raw genetic data uploads to genealogical databases, even if hidden from users, could enable genotype reconstruction of individuals in the database through strategic uploads of datasets by attackers [9].

Related Work. In the literature, the main technical measures currently employed for privacy protection in kinship detection include: anonymization [15], Intel Software Guard Extensions (SGX) [3], and differential privacy [8]. Although these methods alleviate the privacy leakage problem to a certain extent, they have some drawbacks. According to [25], SGX is currently deprecated on client central processing units and differential privacy may severely degrade the genetic data quality. Homomorphic encryption, considered resistant to quantum computer attacks, allows computation over ciphertexts without decryption. Therefore, those kinship detection methods based on homomorphic encryption become particularly important. De Cristofaro et al. [7] presented a protocol for privacy-preserving genetic relatedness test that allows a cloud server to conduct relatedness tests on encrypted genetic data, reducing the test to a data matching problem using searchable encryption [24]. Their privacy-preserving algorithms encountered low efficiency for some situations, e.g., for edit distance and longest common subsequence, requiring over 10^4 s. A projection-based approach for estimating kinship and related statistics in admixed populations, named SIGFRIED, was proposed in [25], which utilizes existing reference genotype datasets to estimate admixture rates for individuals, and the homomorphic encryption scheme CKKS [5] to protect the privacy. The modular formulation allows for efficient and secure kinship estimation among multiple sites, leveraging homomorphic encryption for privacy protection. The method shows promise in accurately estimating kinship with reduced computational burden compared to traditional methods like principal component analysis [6] or expectation-maximization [13]. However, drawbacks include the need for optimization to reduce memory usage and the trade-offs between privacy protection and data quality. To address these issues, the iDASH-2023 competition [18] specifically set up a track (Track 1) aimed at developing a secure and efficient method based on homomorphic encryption technology to utilize genetic genealogy databases to assist law enforcement while minimizing the risk of privacy violations against innocent individuals.

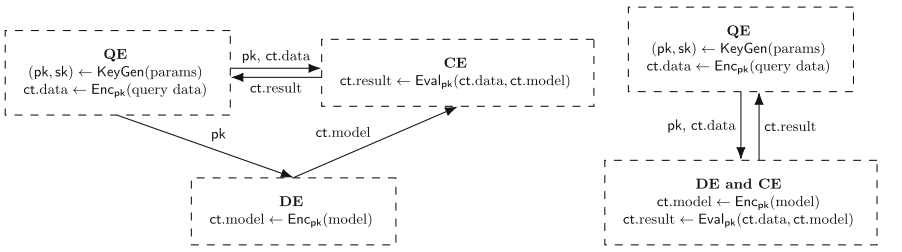
Two Challenges for Kinship Estimation over Encrypted Data. Track 1 of iDASH-2023 required participants to identify the relatedness between genetic samples and genetic databases based on encrypted genetic data. There are three entities: A querying entity (QE) that holds the genome of a target individual, a database owner (DE) who manages a genetic genealogy database, and a non-colluding trusted computing entity (CE) that performs genome detection using the encrypted data from QE and DE. The goal is for QE to find out if the genome of the target individuals (or relatives) is in the database. Assume that

the single-nucleotide polymorphism (SNP) data from DE is $\boldsymbol{A} \in \{0,1,2\}^{m\times d}$ and $\boldsymbol{Q} \in \{0,1,2\}^{m\times n}$, where d (resp. n) is the numbers of individuals in the database (resp. query) and m is the number of available SNP variants for each individual. The goal is to compute a *kinship estimation* r_i $(i \leq n)$ for each query sample, from which one can decide if the query individual is related to any individuals in the database. In this work, we use the *allele-sharing kinship coefficient* (see, e.g., [21, p. 39]) for kinship detection. Roughly, r_i can be computed as follows: Compute $\boldsymbol{R} = (\boldsymbol{Q} - \boldsymbol{E})^{\mathrm{T}}(\boldsymbol{A} - \boldsymbol{E}) \in \mathbb{Z}^{n\times d}$ with $\boldsymbol{E}$'s entries all 1; Compute $r_i = \max(\boldsymbol{R}_i)$, where $\boldsymbol{R}_i$ is the i-th row of $\boldsymbol{R}$. The iDASH-2023 competition requires that participating teams must complete all computational tasks within 10 min, and both $\boldsymbol{A}$ and $\boldsymbol{Q}$ must be processed in an encrypted form, where $(m, d, n) = (16\,344, 2\,000, 400)$. However, when applying the above process with homomorphic encryption, there are *two main challenges*. The one is high-dimensional matrix multiplication over encrypted integers, which is too costly. For instance, with the homomorphic encryption library SEAL [17], a matrix multiplication of two matrices with dimensions 256×259 and 259×257 costs about 43 s over encrypted data [4]. The other is the need of n maximum of vectors with dimension d, which is even more costly than matrix multiplication. In fact, the comparison is one of the well-known operations that is unfriendly for those homomorphic encryption schemes (e.g., BGV [2]) supporting integer arithmetic. Based on the performance of the state-of-the-art homomorphic comparison method reported in [14], we estimate that it would take more than 500 s to compute once the maximum of a 2000-dimensional integer vector.

Our Contribution. This paper is partly based on the participation of the authors (team LARC) in this competition. Although we did not ultimately become the winners, we believe our methods and results can still provide references and be useful for solving related problems. In particular, we propose two adaptions of kinship estimation, which are friendly for homomorphic encryption. Besides, we consider another framework (Fig. 1b) in this paper, which is different from that of the competition (Fig. 1a). We also implement our protocol for secure relative detection with the BGV scheme [2] implemented in the homomorphic encryption library SEAL [17] and test its performance with the data from iDASH-2023.

Two Adaptions of Kinship Estimation. To address the previously mentioned challenges, we propose two adapted kinship estimation methods, which involve only one matrix-vector multiplication. Based on an observation of the competition data, we propose a simple method called *negative-sum kinship estimation* that uses the negative-sum of the vector as a substitute for the vector's maximum. Experiments demonstrate that this simplified method is highly applicable to the competition dataset, achieving Area under the ROC Curve (AUC) of 0.85. We also propose another technique named *minority-sum kinship estimation* based on singular value decomposition (SVD) that extracts the minor components of genetic data $\boldsymbol{A}$ into only one vector and uses it to replace the entire gene database $\boldsymbol{A}$, which is precisely the opposite of the principal component analysis (PCA). The underlying principle is roughly that (1) the closer the kinship,

the smaller the genetic differences; (2) since most human genes are identical, the principal components make only a limited contribution to kinship detection. This technique simplifies matrix-matrix multiplication to matrix-vector multiplication, significantly reducing the computational cost. Combining with the above negative-sum method, the AUC achieves surprisingly 1.0 for the test data.



(a) Three-party framework (b) Two-party framework

Fig. 1. Two frameworks for secure relative detection

In the Track 1 of iDASH-2023 competition, participants are required to identify the relatedness between genetic samples and genetic databases based on the ciphertext of genetic data in the framework in Fig. 1a. It is suitable for outsourced computation. However, according to our experiments, the computational overhead required for secure relative detection is not substantial (see Sect. 4). Moreover, in this framework, there is a major drawback: CE can easily collude with either of the other two parties (QE or DE), causing the third party's privacy to be leaked. To address this problem, we adopt a two-party computation framework, as shown in Fig. 1b. In this framework, only two participants are involved: QE and DE, where DE is also CE, undertaking the computational tasks in the query. The main steps of the secure relative detection protocol under this framework are as follows: QE generates a public-private key pair of a certain homomorphic encryption scheme, encrypts the query data, and sends the encrypted data to DE; DE uses the querying party's evaluation key to compute queries on encrypted data, obtains the encrypted results, and sends them to QE; QE decrypts to obtain the query results. Note that QE's genetic data and query results are always encrypted, and only the private key owner can correctly decrypt them. Since the data of DE never leaves its server, the data can be in plaintext form during the computation, thus further reducing the overhead of encrypted computation.

Secure Relative Detection Protocol. Computing the above two adapted kinship estimations essentially involves computing ultra-high-dimensional matrix-vector multiplication. To handle ultra-high-dimensional matrix-vector multiplication over encrypted data, we reduce it into multiple small-dimensional matrix-vector multiplications that exactly match the homomorphic encryption parameters by

using a binary dividing method. For the small-dimensional matrix-vector multiplication over encrypted data, we use the diagonal encoding method [12]. This results in a secure relative detection protocol (Protocol 3). We implement the proposed methods for secure relative detection with the BGV [2] scheme in SEAL. The code is publicly available at https://github.com/velenchan/larc23. For the competition dataset, our methods can finish all the computations (400 individuals with each 16 384 SNP variants, totally 6 553 600 SNP data) of the framework in Fig. 1a and b with 8 threads, 3.5 GB of memory and within 5.6 and 1.8 s, respectively, which includes encryption, encoding, evaluation, decoding and decryption. For a comparison, securely computing the kinship estimation for 86 individuals with 60 000 SNP variants (totally 5 160 000 SNP data) in SIGFRIED [25] requires approximately 90 s and 4 GB of memory with 40 threads.

2 Preliminaries

We first revoke some basics on relative detection and homomorphic encryption.

Relative Selection. The method of *single-SNP averages* is an efficient and effective method for relative detection. It works as follows (taken from [21, p. 39]): Let S_{B_j} be the genotype of an individual B at the j-th SNP, coded as 0, 1 or 2. Analogous to the definition of co-ancestry, a natural way to score the similarity of two individuals at each SNP is as the probability of a match between alleles drawn randomly from each. Matching homozygotes $(0,0)$ or $(2,2)$ score 1; discordant homozygotes $(0,2)$ score 0; while $(0,1)$, $(1,1)$ and $(1,2)$ all score 0.5. Averaged over m SNP variants, the *allele-sharing kinship coefficient* [23] between individuals B and C is defined as $K(B,C) = \frac{1}{2} + \frac{1}{2m}\boldsymbol{x}_B\boldsymbol{x}_C^{\mathrm{T}}$, where $\boldsymbol{x}_B$ is the (row) vector with j-th entry $S_{B_j} - 1$.

The BGV Homomorphic Encryption Scheme. For secure relative detection, we utilize the BGV scheme [2], which is a good choice for integer arithmetic operations. Let $R := \mathbb{Z}[X]/\langle X^N + 1\rangle$ with N an integer. In BGV, the plaintext space is $R_p = R/qR$, where p is the *plaintext modulus*. The ciphertext space of BGV is $R_q = R/qR$, where q is the *ciphertext modulus*, a large integer. Roughly, the BGV scheme consists of four algorithms KeyGen, Enc, Dec, and Eval. It is *semantic secure* and *weakly circular* secure under the RLWE assumption [16] and the circular security assumption. The BGV scheme supports single instruction multiple data (SIMD) operations, i.e., performing an operation on a ciphertext corresponds to performing the same operation on ℓ *slots* of the message in parallel. In fact, the messages for BGV are vectors in $(\mathbb{Z}/(p\mathbb{Z}))^\ell$ with $\ell = N$ for a power-of-two N. For $\boldsymbol{x} = (x_i)_{0\le i<\ell}$ and $\boldsymbol{y} = (y_i)_{0\le i<\ell}$, let $\mathsf{ct}.\boldsymbol{x}$ and $\mathsf{ct}.\boldsymbol{y}$ be the ciphertexts of the encrypted by BGV under a same public key, i.e., $\mathsf{ct}.\boldsymbol{x} \leftarrow \mathsf{Enc}(\boldsymbol{x})$ and $\mathsf{ct}.\boldsymbol{y} \leftarrow \mathsf{Enc}(\boldsymbol{y})$. Then BGV supports the following basic operations: $\mathsf{Add}(\mathsf{ct}.\boldsymbol{x}, \mathsf{ct}.\boldsymbol{y})$: Output a new ciphertext $\mathsf{ct}.\boldsymbol{z}$ satisfies $\mathsf{Dec}(\mathsf{ct}.\boldsymbol{z}) = \boldsymbol{x} + \boldsymbol{y}$; $\mathsf{Mul}(\mathsf{ct}.\boldsymbol{x}, \mathsf{ct}.\boldsymbol{y})$: Output a new ciphertext $\mathsf{ct}.\boldsymbol{z}$ satisfies $\mathsf{Dec}(\mathsf{ct}.\boldsymbol{z}) = \boldsymbol{x} \circ \boldsymbol{y}$, where $\circ$ is for Hadamard product, i.e., component-wise multiplication; $\mathsf{CMul}(\boldsymbol{x}, \mathsf{ct}.\boldsymbol{y})$: Output a new ciphertext $\mathsf{ct}.\boldsymbol{z}$ satisfies $\mathsf{Dec}(\mathsf{ct}.\boldsymbol{z}) = \boldsymbol{x} \circ \boldsymbol{y}$; $\mathsf{Rot}_k(\mathsf{ct}.\boldsymbol{x})$: Convert $\mathsf{ct}.\boldsymbol{x} = \mathsf{Enc}(x_0, \ldots, x_{\ell-1})$ into a new

ciphertext $\mathsf{ct}.\boldsymbol{z}$ that encrypts $(x_k, \ldots, x_{\ell-1}, x_0, \ldots, x_{k-1})$. Let $\boldsymbol{A} = (a_{i,j})$ be an $n \times m$ matrix and $\boldsymbol{v}$ a vector of dimension m with $m = r \cdot n$. To compute $\boldsymbol{u} = \boldsymbol{A} \cdot \boldsymbol{v}$, we use the Halevi-Shoup's diagonal vector encoding [11]. In diagonal encoding, the i-th diagonal vector of the matrix $\boldsymbol{A}$ (where $0 \leq i < n$) is defined as $\boldsymbol{d}_i(\boldsymbol{A}) \leftarrow (a_{[j]_n,[i+j]_m})_{0 \leq j < m}$.

Algorithm 1 (Encrypted linear transformation on ciphertexts)

Input: $\mathsf{ct}.\boldsymbol{d}[k \cdot i + j]$ that encrypts the rotated diagonal vector $\boldsymbol{d}_{k \cdot i + j}$ towards right by $-k \cdot i$ positions, where $0 \leq i < l$, $0 \leq j < k$, $m = r \cdot n$ and $n = k \cdot l$; $\mathsf{ct}.\boldsymbol{v}$, a ciphertext of an m-dimensional vector $\boldsymbol{v}$.

Output: $\mathsf{ct}.\boldsymbol{u}$, the resulting vector of ciphertexts.

Initialize $\mathsf{ct}.\boldsymbol{u} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\boldsymbol{0})$.
for $j = 0$ to $k - 1$ **do**
  $\mathsf{ct}.\boldsymbol{v}[j] \leftarrow \mathsf{Rot}_j(\mathsf{ct}.\boldsymbol{v})$
for $i = 0$ to $l - 1$ **do**
  $\mathsf{ct}.\boldsymbol{u}[i] \leftarrow \mathsf{Add}(\mathsf{ct}.\boldsymbol{v}[j], \mathsf{ct}.\boldsymbol{d}[k \cdot i + j])_{0 \leq j < k}$
  Update $\mathsf{ct}.\boldsymbol{u} := \mathsf{Add}\,(\mathsf{ct}.\boldsymbol{u}, \mathsf{Rot}_{k \cdot i}(\mathsf{ct}.\boldsymbol{u}[i]))$
for $i = 0$ to $\lfloor \log r \rfloor$ **do**
  Update $\mathsf{ct}.\boldsymbol{u} := \mathsf{Add}\,(\mathsf{ct}.\boldsymbol{u}, \mathsf{Rot}_{2^i \cdot n}(\mathsf{ct}.\boldsymbol{u}))$
return $\mathsf{ct}.\boldsymbol{u}$

3 Training Model with Plaintext Data

The problem that we mainly considered here is secure relative detection in (forensic) databases. The querying entity (QE, such as law enforcement) holds the genome of a target individual, denoted by $\boldsymbol{Q} \in \{0, 1, 2\}^{m \times n}$, where n is the number of individuals and m is the number of SNP variants for each individual. The database owner (DE) manages a genetic genealogy database $\boldsymbol{A} \in \{0, 1, 2\}^{m \times d}$, where d is the number of individuals in the database. The goal is to compute a *kinship estimation* for each individual, which can be used to decide if the query individual has relatives in the database.

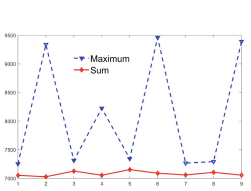
Data. We validate the methods for relative detection with a genomic database from DE and a query genotype dataset from QE. The genomic database consists of 2 000 genomes, each containing genotypes for 16 344 genetic variants. The query genotype dataset contains 400 genomes with the genotypes for the same set of variants. Among these 400 genomes, there are exactly 200 genomes that have relatives in the database. All data are from iDASH-2023 [18].

Kinship Estimation. We will use an adapted allele-sharing kinship coefficient as our kinship estimation between two individuals. Since m is the number of SNP variants for each individual, it is fixed, and hence independent of the computational task. Thus, we can simplify the above formula as $K'(B, C) = \boldsymbol{x}_B \boldsymbol{x}_C^{\mathrm{T}}$. We call $K'(B, C)$ the *adapted allele-sharing kinship coefficient* between B and C. According to the definition of $K'(B, C)$, we can first compute $\boldsymbol{R} = (\boldsymbol{Q} - \boldsymbol{E})^{\mathrm{T}}(\boldsymbol{A} - \boldsymbol{E}) \in \mathbb{Z}^{n \times d}$, where $\boldsymbol{E}'s$ entries are all 1. In fact, the i-th row $\boldsymbol{R}_i$ of $\boldsymbol{R}$ are the adapted allele-sharing kinship coefficients between

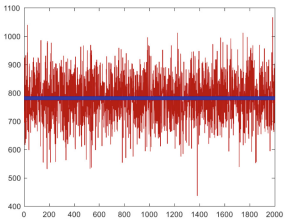
the i-th query individual and each individual in the database. Now we define the *maximum kinship estimation* for the i-th query individual as

$$r_i = \max(\boldsymbol{R}_i),\ i = 1, \cdots, n. \tag{1}$$

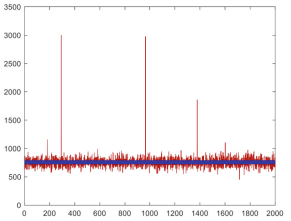
Nevertheless, directly computing the kinship estimation in (1) on encrypted data is considerably difficult. The difficulty primarily stems from two factors: Firstly, directly computing over encrypted data requires the multiplication of high-dimensional encrypted matrices (for computing $\boldsymbol{R}$), a costly operation for encrypted data. Secondly, for each query individual, the maximum of a d-dimensional vector needs to be determined, which is even more expensive for encrypted computation. Consequently, we need to adapt the plaintext algorithm further. We now introduce two methods to avoid computing the maximum of a vector and high-dimensional matrix multiplication.



(a) Sum and maximum for random chosen individuals



(b) $\boldsymbol{Q}_4^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})$, avg ≈ 781.6



(c) $\boldsymbol{Q}_5^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})$, avg ≈ 756.8

Fig. 2. Observations for the negative-sum kinship estimation

Negative-Sum Kinship Estimation. The first method involves using the negative sum of the vector entries to replace the vector's maximum value, thereby avoiding the need to compute the maximum. This is based on an observation of the query dataset (400 individuals) provided by iDASH-2023: for these individuals, the larger the maximum value of the vector $\boldsymbol{R}_i$, the smaller the sum of its entries tend to be. Although this observation does not hold for all individuals, it is valid for the majority. As shown in Fig. 2a, for nine randomly chosen individuals, we consider the maximum values and the sum of the entries for these nine $\boldsymbol{R}_i$ vectors, with only two points, violated this observation. More specifically, for those samples no kinship matching in the database, e.g., $\boldsymbol{Q}_4$, the distribution of $\boldsymbol{Q}_4^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})$ appears to be a uniform distribution centered around the mean ≈ 781.6 (see Fig. 2b); whereas for those samples existing a kinship matching in the database, e.g., $\boldsymbol{Q}_5$, the maximum value of $\boldsymbol{Q}_5^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})$ is quite large, but the mean ≈ 756.8, and hence the sum, are smaller (see Fig. 2c).

Based on these observations, we replace maximum by the negative sum, which means instead of computing $r_i = \max(\boldsymbol{R}_i)$, we compute $r_i' = -\sum_{j=1}^{n} R_{i,j}$, where $R_{i,j}$ is the j-th entry of $\boldsymbol{R}_i$. For all query individuals, we have $-\boldsymbol{R}\cdot\mathbf{1} = -(\boldsymbol{Q}-\boldsymbol{E})^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})\cdot\mathbf{1} = -\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})\cdot\mathbf{1}+\boldsymbol{E}^{\mathrm{T}}(\boldsymbol{A}-\boldsymbol{E})\cdot\mathbf{1} \in \mathbb{Z}^n$, where $\mathbf{1}$ and $\boldsymbol{E}$ are the

vector and matrix with all entries 1. After removing the constant independent of the queries (the latter term), we obtain the *negative-sum kinship estimation*:

$$\boldsymbol{r}_1 = -\boldsymbol{Q}^{\mathrm{T}} \cdot \boldsymbol{m}_1 \in \mathbb{Z}^n \quad (\boldsymbol{m}_1 = (\boldsymbol{A} - \boldsymbol{E}) \cdot \mathbf{1}_{d\times 1} \in \mathbb{Z}^m), \tag{2}$$

which clearly only involves a matrix-vector multiplication.

Minority-Sum Kinship Estimation. In our second adaption on the kinship estimation, we use the *minor component* of the database, which is exactly the opposite of principal component analysis (PCA). The basic idea can be summarized as the following: (1) There exists an inverse relationship between the degree of kinship and the number of genetic variations, with closer kinship exhibiting smaller genetic differences. (2) Given the high degree of genetic homogeneity across the human population, the contribution of principal components to relative detection is very limited. Thus, we first extract the minor component of the database by singular value decomposition (SVD).

For $\boldsymbol{A} \in \mathbb{R}^{m\times d}$ with $m \geq d$, the SVD of $\boldsymbol{A}$ can be reformulated as $\boldsymbol{A} = \sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^{\mathrm{T}} + \sigma_2 \boldsymbol{u}_2 \boldsymbol{v}_2^{\mathrm{T}} + \cdots + \sigma_d \boldsymbol{u}_d \boldsymbol{v}_d^{\mathrm{T}}$, where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_d \geq 0$ are the *singular values* of $\boldsymbol{A}$ and $\boldsymbol{u}_i \in \mathbb{R}^m$ (resp. $\boldsymbol{v}_i \in \mathbb{R}^n$) are the *left* (resp. *right*) *singular vectors* of $\boldsymbol{A}$. If there exists an integer $t < d$ such that $\sigma_t \gg \sigma_{t+1}$, we usually call $\sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^{\mathrm{T}} + \sigma_2 \boldsymbol{u}_2 \boldsymbol{v}_2^{\mathrm{T}} + \cdots + \sigma_t \boldsymbol{u}_t \boldsymbol{v}_t^{\mathrm{T}}$ the *principal components* of $\boldsymbol{A}$, denoted by $\boldsymbol{A}_p$, and call $\boldsymbol{A} - \boldsymbol{A}_p$ the *minor components* of $\boldsymbol{A}$. For the database matrix $\boldsymbol{A}$ of DE in the iDASH-2023 competition, we have $\sigma_1 \approx 4669.6 \gg \sigma_2 \approx 161.2 \geq \ldots \geq \sigma_{2000} \approx 34.9$. Therefore, the principal components of $\boldsymbol{A}$ is $\boldsymbol{A}_p = \sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^{\mathrm{T}}$, from which we extract the minor components of $\boldsymbol{A}$ (that show the specificity of SNPs): $\boldsymbol{A} - \boldsymbol{u}_1 \boldsymbol{u}_1^{\mathrm{T}} \boldsymbol{A}$. Again, we use the sum to approximate the maximum (Since we removed the principal components, the sum no longer takes the negation):

$$\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{A} - \boldsymbol{u}_1 \boldsymbol{u}_1^{\mathrm{T}} \boldsymbol{A}) \cdot \mathbf{1}_{d\times 1} = \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{A} \cdot \mathbf{1}_{d\times 1} - \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{u}_1 \boldsymbol{u}_1^{\mathrm{T}} \boldsymbol{A} \cdot \mathbf{1}_{d\times 1} \in \mathbb{R}^k. \tag{3}$$

Notice that the above result is in $\mathbb{R}^k$ while the negative-sum kinship estimation in (2) is always in $\mathbb{Z}^k$. To adapt to the BGV scheme [2], which only supports integer operations, we need to convert the results of (3) into integers as well. For instance, we can simply define

$$\boldsymbol{r}_2 = \boldsymbol{Q}^{\mathrm{T}} \boldsymbol{m}_2 \in \mathbb{Z}^{n\times 1}, \tag{4}$$

as the *minority-sum kinship estimation* for the query dataset, which again only involves a matrix-vector multiplication. Here,

$$\boldsymbol{m}_2 = c \cdot \boldsymbol{A} \cdot \mathbf{1}_{d\times 1} - \boldsymbol{u} \tag{5}$$

with a constant c and $\boldsymbol{u} = \lfloor c \cdot \boldsymbol{u}_1 \boldsymbol{u}_1^{\mathrm{T}} \boldsymbol{A} \cdot \mathbf{1}_{d\times 1} \rfloor \in \mathbb{Z}^m$. In fact, a larger c implies more accuracy but a larger plaintext modulus for BGV. We also note that we tested multiple times by randomly extracting 80% of the data from the database (1 600 individuals), and the resulting $\boldsymbol{u}$'s are almost identical.

Evaluation on Plaintext Data. We tested the three types of kinship estimation provided above with the query data $\boldsymbol{Q} \in \mathbb{Z}^{16384\times 400}$ and genetic data

Table 1. Accuracy for different kinship estimation

Kinship estimation	Maximum (1)	Negative-sum (2)	Minority-sum (4)
AUC	1.0000	0.8519	1.0000

$\boldsymbol{A} \in \mathbb{Z}^{16384\times 2000}$ given in iDASH-2023, where the constant c in (5) was set to 10. The experimental results in Table 1 show that these test data fully match the original definition of the maximum kinship estimation in (1). There exist some loss of accuracy for the negative-sum kinship estimation in (2), but surprisingly, the minority-sum kinship estimation in (4) can also achieve the same level of accuracy as the maximum one.

4 Secure Relative Detection

Adversaries Model. As pointed out in the introduction, we consider the scenario involving two parties, namely, the querying entity (QE) and the database owner (DE). QE generates the public and private keys for homomorphic encryption, while DE undertakes the computational tasks. We prove the security under the *semi-honest* adversary model, also known as the *honest-but-curious* model.

Algorithm 2 (High-dimensional encrypted matrix-vector multiplication)

Input: For $0 \le \alpha < \lceil \frac{n}{\bar{n}} \rceil, 0 \le \beta < \lceil \frac{m}{\bar{m}} \rceil$, $\mathsf{ct}.\boldsymbol{d}_{\alpha,\beta}[k \cdot i + j] = \mathsf{Enc}(\rho(\boldsymbol{d}_{\alpha,\beta,k\cdot i+j}; -k \cdot i))$, where $\boldsymbol{d}_{\alpha,\beta,k\cdot i+j}$ is the $(k \cdot i + j)$-th diagonal vector of the matrix $\bar{\boldsymbol{A}}_{\alpha,\beta}$, $i < l$, $j < k$, $\bar{m} = r \cdot \bar{n}$ and $\bar{n} = k \cdot l$; $\mathsf{ct}.\boldsymbol{v}_\beta$, ciphertexts of $\bar{m}$-dimensional vector $\bar{\boldsymbol{v}}_\beta$.
Output: $\mathsf{ct}.\boldsymbol{u}$, a ciphertext of $\boldsymbol{Av}$.

```
Initialize ct.u ← Enc_pk(0).
for α = 0 to ⌈n/n̄⌉ − 1 do
    Initialize ct.u_α ← Enc_pk(0).
    for β = 0 to ⌈m/m̄⌉ − 1 do
        Calling Algo. 1 with input as (ct.d_{α,β}[k·i + j])_{i,j} and ct.v_β returns ct.u_α.
    Update ct.u := Add(ct.u, CMul({0_{α·n̄}, 1_{n̄}, 0_{(r−α−1)·n̄}}, ct.u_α))
return ct.u
```

High-Dimensional Encrypted Matrix-Vector Multiplication. For the BGV scheme, if N is a power-of-two integer, then the number of slots is $\ell = N$, which means that one ciphertext can pack exactly ℓ integers modulo p. If $m \gg \ell \ge n$, then computing $\boldsymbol{Av}$ for an $n \times m$ matrix $\boldsymbol{A}$ and an m-dimensional vector $\boldsymbol{v}$ over encrypted data can be reduced to multiple matrix-vector multiplications with small dimension. For instance, the matrix $\boldsymbol{A}$ can be represented by a block matrix $(\bar{\boldsymbol{A}}_{\alpha,\beta})_{\alpha<\lceil \frac{n}{\bar{n}} \rceil,\beta<\lceil \frac{m}{\bar{m}} \rceil}$, where each block $\bar{\boldsymbol{A}}_{\alpha,\beta}$ is a $\bar{n} \times \bar{m}$ matrix. The vector $\boldsymbol{v}$ can be represented by $(\bar{\boldsymbol{v}}_\beta)_{\beta<\lceil \frac{m}{\bar{m}} \rceil}$ in the same manner. For efficiency, we may further assume that $\bar{m} = \ell$ and $\bar{n} = \sqrt{\ell}$. Note that it follows from $n \le \ell$ that the output of Algorithm 2 is a single ciphertext.

The Main Protocol. Note that both (2) and (4), simplified from (1), only involve a matrix-vector multiplication, which can be computed by Algorithm 2. Now we present the main protocol as Protocol 3.

Communication. By partitioning both the matrix and the model vector into blocks of $\bar{\boldsymbol{Q}}^{\mathrm{T}} \in \mathcal{R}^{\bar{n}\times\bar{m}}$ and $\bar{\boldsymbol{m}} \in \mathcal{R}^{\bar{m}}$, where $\bar{m} = \ell$. For QE, the matrix is divided into $\lceil \frac{n}{\bar{n}} \rceil \cdot \lceil \frac{m}{\bar{m}} \rceil$ blocks, and each block is encrypted into $\bar{n}$ ciphertexts. As a result, QE sends $\bar{n} \cdot \lceil \frac{n}{\bar{n}} \rceil \cdot \lceil \frac{m}{\bar{m}} \rceil$ ciphertexts to DE. Since the output of Algorithm 2 is a single ciphertext, DE needs to send only one ciphertext to QE.

Security. The semantic secure of the BGV scheme preserves the privacy of the query data $\boldsymbol{Q}$ and the result $\boldsymbol{u}$ of QE, preventing DE from obtaining any information about $\boldsymbol{Q}$ and $\boldsymbol{u}$. Although the model data $\boldsymbol{m}$ is involved in the computation, it is difficult to infer more information about $\boldsymbol{m}$ than that implied by $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{m} = \boldsymbol{u}$. Note that the equation can still be established even if relative detection is carried out in an ideal world. Therefore, Protocol 3 does not leak more information on $\boldsymbol{m}$ than in the ideal world and hence is secure for DE.

Protocol 3 (Secure relative detection)

Input of QE: The query data $\boldsymbol{Q} \in \{0,1,2\}^{m\times n}$, block size $0 \le \alpha < \lceil \frac{n}{\bar{n}} \rceil, 0 \le \beta < \lceil \frac{m}{\bar{m}} \rceil$ and the security parameter λ, where block matrix size is $\bar{n} \times \bar{m}$, $\bar{n} = k \cdot l$, and $\bar{m} = r \cdot \bar{n}$.
Input of DE: The model data $\boldsymbol{m} \in \mathbb{Z}^m$ (either $\boldsymbol{m}_1$ in (2) or $\boldsymbol{m}_2$ in (5)).
QE:

1: $(p, q, N, \gamma) \leftarrow \mathsf{BGV.Setup}(1^\lambda)$. ▷ *$N$ is a power-of-two integer greater than n.*
2: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BGV.KeyGen}(p, q, N, \gamma)$.
3: Set $\mathsf{ct}.\boldsymbol{d}_{\alpha,\beta}[k \cdot i + j] \leftarrow \mathsf{BGV.Enc}_{\mathsf{pk}}(\rho(\boldsymbol{d}_{\alpha,\beta,k\cdot i+j}; -k \cdot i))$ for $0 \le i < l$ and $0 \le j < k$, where $\boldsymbol{d}_{\alpha,\beta,k\cdot i+j}$ is the $(k \cdot i + j)$-th diagonal vector of the block matrix $\bar{\boldsymbol{Q}}^{\mathrm{T}}_{\alpha,\beta}$.
4: Send $(\mathsf{ct}.\boldsymbol{d}_{\alpha,\beta}[k \cdot i + j])_{\alpha,\beta,i,j}$ to DE.

DE:

5: $\mathsf{ct}.\boldsymbol{m}_\beta \leftarrow \mathsf{BGV.Enc}_{\mathsf{pk}}(\boldsymbol{m}_\beta)$. ▷ *Encrypting $\boldsymbol{m}_\beta$ may not be neccessary.*
6: Calling Algo. 2 with input as $(\mathsf{ct}.\boldsymbol{d}_{\alpha,\beta}[k \cdot i + j])_{\alpha,\beta,i,j}$ and $(\mathsf{ct}.\boldsymbol{m}_\beta)_\beta$ returns $\mathsf{ct}.\boldsymbol{u}$.
7: Send $\mathsf{ct}.\boldsymbol{u}$ to QE.

QE:

8: Decrypt $\boldsymbol{u} \leftarrow \mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}.\boldsymbol{u})$ and **return** $\boldsymbol{u}$.

Performance. We implement Protocol 3 with the BGV scheme in SEAL [17]. In this section, we report the computational efficiency, communication overhead, and accuracy of Protocol 3. All the computations are carried out in Ubuntu 22.04 on a personal computer with Intel Core i9-12900K CPU (3.20 GHz) and 32 GB RAM. In addition, For further speedup, we use 8 threads parallel computation with OpenMP. All experiments have identical testing settings with the same security parameters. In particular, the plaintext modulus of BGV is $p = 33538049 \approx 2^{25}$, the polynomial degree is $N = 2^{13} = 8,192$, and the bit size of the ciphertext modulus is $\log q \approx 200 = 41 + 39 + 40 + 40 + 40$, which implies 128-bit security according to [1]. For the data described in Sect. 3, we compute in advance the negative-sum kinship detection model $\boldsymbol{m}_1$ in (2) and the minority-sum kinship detection model $\boldsymbol{m}_2$ in (5) from all data (2 000 individuals) in the dataset, respectively.

The performance of Protocol 3 is presented in Table 2, where AUC is computed with the resulting kinship estimation and the 0–1 sequence extracted from the query genotype dataset. As illustrated in Table 2, both models share an acceptable memory and a computational overhead. The timings include initialization, reading the query and model data, encrypting the data, computing with encrypted data, and decrypting the results. Furthermore, the accuracy of the results is almost the same as that of computing with plaintext data.

Table 2. Performance of Protocol 3

Model	Memory (MB)	Total Time (s)	AUC
$\boldsymbol{m}_1$	3459.98	1.71	0.85
$\boldsymbol{m}_2$	3483.64	1.80	1.00

For comparison, we also present the performance of our submission for iDASH-2023 in Table 3. As shown in the Table 3, compared to the computation model required by iDASH-2023 (Fig. 1a), Protocol 3 achieves a 3x speedup. The reason is that, in Protocol 2, it is only necessary to read the model parameters $\boldsymbol{m}$, without the need to read and encrypt the database matrix $\boldsymbol{A}$.

Table 3. Detailed timing (s) cost in iDASH-2023 style

Model	Init	Read $\boldsymbol{Q}$	Enc. $\boldsymbol{Q}$	Read $\boldsymbol{A}$	Enc $\boldsymbol{A}$	Comput	Dec	Total
$\boldsymbol{m}_1$	0.187	0.153	0.924	0.818	3.001	0.446	0.002	5.531
$\boldsymbol{m}_2$	0.185	0.152	0.959	0.819	2.963	0.495	0.006	5.579

As another comparison, securely computing the kinship estimation for 86 individuals with 60 000 SNP variants (totally 5 160 000 SNP data) by the state-of-the-art method SIGFRIED [25] requires approximately 90 s and 4 GB of memory with 40 threads. Their experiments were based on the CKKS [5] scheme implemented in SEAL, with 128-bit security as well. Recall that in our test, the query data include 400 individuals with 16 384 SNP variants (totally 6 553 600 SNP data). With 8 threads, our methods cost 3.5 GB of memory, and 5.6 and 1.8 s for the two frameworks in Fig. 1, respectively.

5 Conclusion

Partly based on the homomorphic encryption track (Track 1) of the iDASH-2023 competition, we presented an efficient privacy-preserving protocol for relative detection based on homomorphic encryption (Protocol 3). To meet the needs of homomorphic encryption, we also proposed two new methods for kinship

estimation, which perform well on the competition dataset. However, it should be noted that the effectiveness of these two methods for more general cases requires further research.

Acknowledgment. We thank the organizers of iDASH-2023 for their efforts in organizing this event and for preparation of the data and challenge. This work was partially supported by National Key Research and Development Program of China (2020YFA0712303), Natural Science Foundation of Chongqing (cstc2021jcyj-msxmx0821, cstc2021yszx-jcyjx0004, 2022yszx-jcx0011cstb, cstb20 23yszx-jcx0008), and Western Young Scholars Program of CAS.

Author contributions. Authors are listed in alphabetical order. All authors reviewed, edited, and approved the final manuscripts.

References

1. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. J. Math. Cryptol. **9**(3), 169–203 (2015)
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. ACM Trans. Comput. Theor. **6**(3), 13 (2014)
3. Chen, F., Dow, M., Ding, S., et al.: PREMIX: privacy-preserving estimation of individual admixture. In: AMIA, vol. 2016, pp. 1747-1755. AMIA (2016)
4. Chen, J., Yang, L., Wu, W., et al.: Homomorphic matrix operations under bicyclic encoding (2024, submitted)
5. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 409–437. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_15
6. Conomos, M.P., Miller, M.B., Thornton, T.A.: Robust inference of population structure for ancestry prediction and correction of stratification in the presence of relatedness. Genet. Epidemiol. **39**(4), 276–293 (2015)
7. De Cristofaro, E., Liang, K., Zhang, Y.: Privacy-preserving genetic relatedness test. arXiv arXiv:1611.03006 (2016)
8. Dervishi, L., Wang, X., Li, W., et al.: Facilitating federated genomic data analysis by identifying record correlations while ensuring privacy. arXiv arXiv:2203.05664 (2022)
9. Edge, M.D., Coop, G.: Attacks on genetic privacy via uploads to genealogical databases. eLife **9**, e51810 (2020)
10. Egeland, T., Mostad, P.F., Mevåg, B., Stenersen, M.: Beyond traditional paternity and identification cases: selecting the most probable pedigree. Forensic Sci. Int. **110**(1), 47–59 (2000)
11. Halevi, S., Shoup, V.: Algorithms in HElib. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 554–571. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_31
12. Halevi, S., Shoup, V.: Faster homomorphic linear transformations in HElib. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 93–120. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_4
13. Huff, C.D., Witherspoon, D.J., Simonson, T.S., et al.: Maximum-likelihood estimation of recent shared ancestry (ERSA). Genome Res. **21**(5), 768–774 (2011)

14. Iliashenko, I., Zucca, V.: Faster homomorphic comparison operations for BGV and BFV. Proc. Priv. Enhancing Technol. **2021**(3), 246–264 (2021)
15. Kale, G., Ayday, E., Tastan, O.: A utility maximizing and privacy preserving approach for protecting kinship in genomic databases. Bioinformatics **34**(2), 181–189 (2017)
16. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. J. ACM **60**(6), 43:1-43:5 (2013)
17. Microsoft: Microsoft SEAL (release 4.1.1). https://github.com/microsoft/SEAL
18. iDASH Privacy Protection Challenge. http://www.humangenomeprivacy.org/
19. Popli, D., Peyrégne, S., Peter, B.M.: KIN: a method to infer relatedness from low-coverage ancient DNA. Genome Biol. **24**(1), 10 (2023)
20. Ram, N., Guerrini, C.J., McGuire, A.L.: Genealogy databases and the future of criminal investigation. Science **360**(6393), 1078–1079 (2018)
21. Speed, D., Balding, D.J.: Relatedness in the post-genomic era: is it still useful? Nat. Rev. Genet. **16**, 33–44 (2014)
22. Städele, V., Vigilant, L.: Strategies for determining kinship in wild populations using genetic data. Ecol. Evol. **6**(17), 6107–6120 (2016)
23. Toro, M., Barragán, C., Óvilo, C., et al.: Estimation of coancestry in Iberian pigs using molecular markers. Conserv. Genet. **3**, 309–320 (2002)
24. Wang, P., Wang, H., Pieprzyk, J.: Common secure index for conjunctive keyword-based retrieval over encrypted data. In: Jonker, W., Petković, M. (eds.) Secure Data Management, SDM 2007. LNCS, vol. 4721, pp. 108–123. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75248-6_8
25. Wang, S., Kim, M., Li, W., et al.: Privacy-aware estimation of relatedness in admixed populations. Brief. Bioinform. **23**(6), bbac473 (2022)