Supplementary Data: SecureMA: protecting participant privacy in genetic association meta-analysis

Wei Xie, Murat Kantarcioglu, William S. Bush, Dana Crawford,
Joshua C. Denny, Raymond Heatherly, Bradley A. Malin

This supplement is organized as follows: Section S1 provides additional figures to complement the description of the SecureMA protocol. Section S2 introduces the details of the meta-analysis and the specific workflows associated with SecureMA. Section S3 describes the three datasets we utilize to evaluate the SecureMA protocol. Section S4 provides additional experiments on the computational accuracy by controlling for tunable parameters associated with the protocol. Finally, Section S5 describes SecureMA in greater detail, while covering the specific technical aspects regarding how each computation is securely performed to support meta-analysis.

S1 Supplementary Figures

This section provides additional figures to describe the SecureMA protocol in greater detail.

Specifically, Fig. S1 illustrates the Setup step around cryptographic keys in the protocol (described in the main manuscript, Section 2.2). We emphasize that, as illustrated later in Fig. S2, the Key Manager who facilitates key generation and distribution is isolated from the rest of the SecureMA system and thus has no access to any data or computations. In practice, this role could be played by a semi-trusted third-party, who is outside the set of participants. For instance, the role could be assumed by a neutral organization with a good reputation in key management, a trustworthy computing module, or even a virtual party representing a distributed and secure mechanism for key generation among many protocol participants [Kate *et al.*, 2009].

Fig. S2 presents the complete activity diagram of SecureMA in sequential order, including the Setup and Secure Computation steps (main manuscript, Section 2).

S2 Meta-analysis and Protocol Participants

Here, we provide additional explanation regarding the computation of meta-analysis, as well as the specific workflow details of SecureMA that are not covered in the main manuscript.

S2.1 Meta-analysis of Genetic Association Studies

To the best of our knowledge, there is no easy and efficient method for performing the square root operation securely (in the denominator of meta-analysis equation). Thus, we square Equation 1 in the main manuscript for easier implementation:

$$Z^2 = (\sum_i \beta_i w_i)^2 \Big/ \sum_i w_i \ , \tag{1}$$

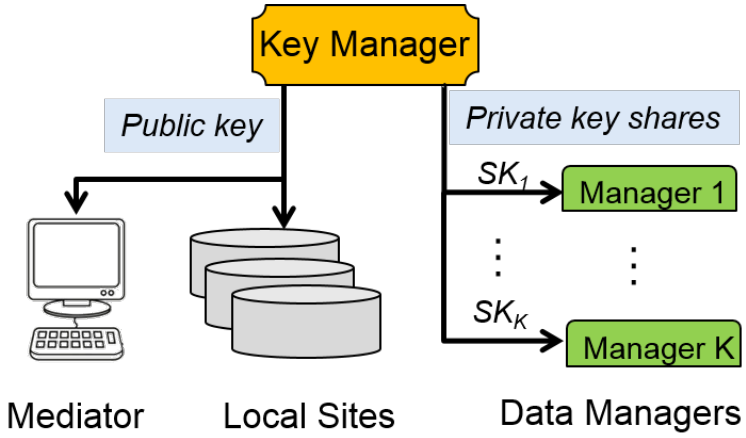


Figure S1: During the Setup step of the SecureMA protocol, encryption/decryption keys are generated and distributed. The public key (for encryption) is broadcast to the mediator and local sites, while the private key (for decryption) is split into secret shares (SK_1, ..., SK_K) which are securely transmitted to the respective data managers.

where the final square root, as well as conversion from Z-score to p-value, of the result of the meta-analysis is performed by the software running on the computer of the scientist issuing the inquiry.

S2.2 Protocol Participants

The major participants of the secure meta-analysis protocol and their roles are summarized below:

- A *Scientist* (e.g., genomicist) issues meta-analysis queries to the protocol and receives the encrypted final results which only he can fully decrypt.
- The *Local Sites* are the individual sites who collect genomic and phenotypic data, as well as conduct their local association studies.
- (Optional) The *Data Managers* (e.g., coordination centers in practice) manage the (encrypted) genomic information on behalf of *local sites*. This optional optimization makes the protocol more practical by supporting meta-analysis while reducing the number of participants required at runtime (e.g., one manager can delegate multiple local sites). The data managers only have limited decryption capabilities, as introduced later.
- The *Mediator* computes the secure meta-analysis equations and responds to the scientist's queries with encrypted results.

S3 Details of Study Data

Our study uses data from three recent multi-site genotype-phenotype association studies. Here we provide a detailed description of these datasets.

eMERGE hypothyroidism study. The first dataset is from the Electronic Medical Records and Genomics (eMERGE) network ([McCarty *et al.*, 2011]). It consists of five different

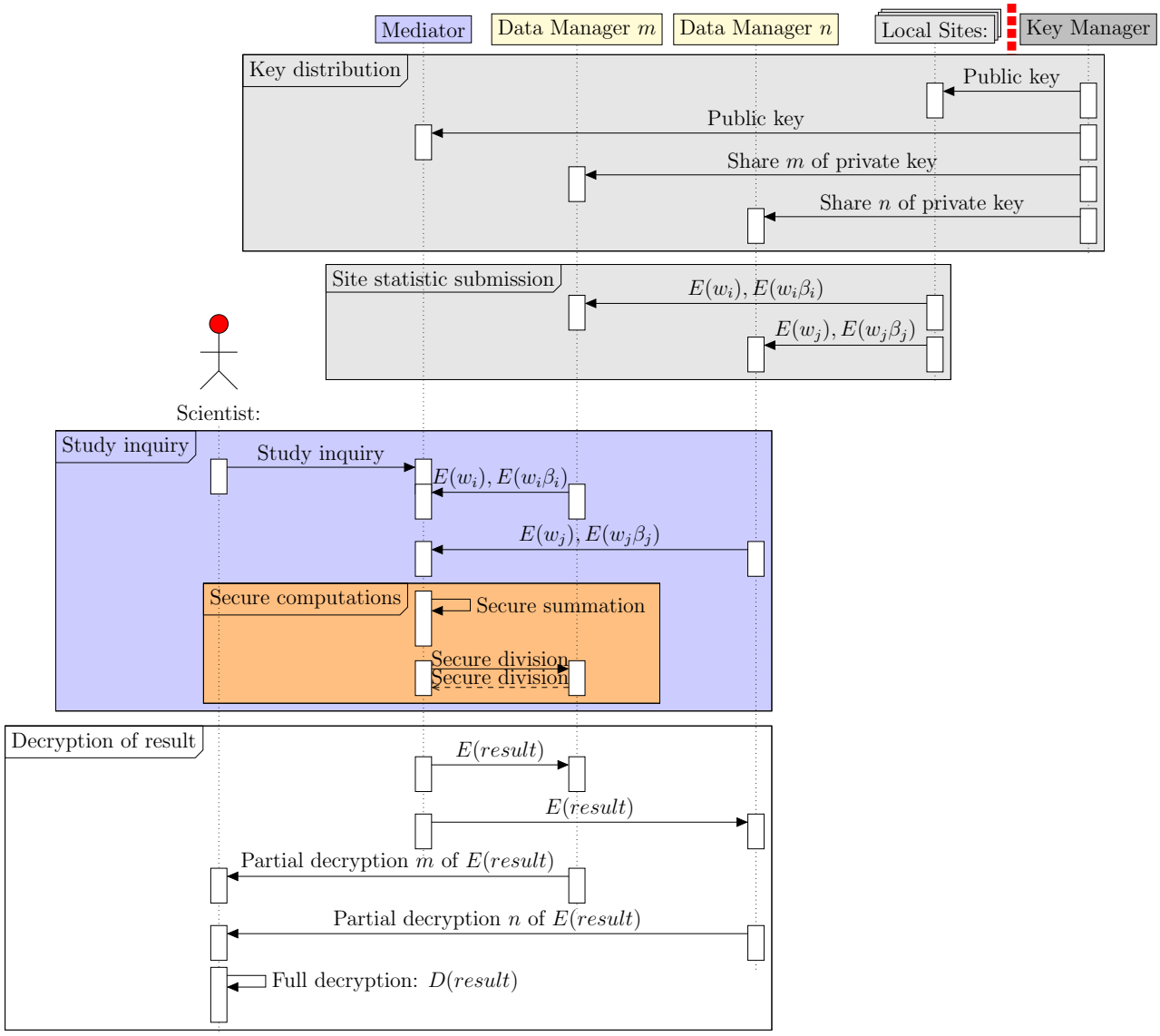


Figure S2: The activity diagram of the SecureMA protocol. Denoted in gray boxes is the one-time Setup step covering key distribution and submission of encrypted site statistics (main manuscript, Section 2.2). In a typical running, a scientist issues a study inquiry to start the protocol, and obtains the study result in the end. In the figure, $E(data)$ and $D(data)$ correspond to the encryption and decryption of data, respectively. There can be multiple *local sites* and *data managers*. The *key manager* is isolated from the rest of the system and his only involvement is key generation and distribution.

sites (e.g., sub-studies) who contributed data: i) the Group Health Cooperative, ii) the Marshfield Clinic, iii) the Mayo Clinic, iv) Northwestern University Medical Center, and v) Vanderbilt University Medical Center. Local-site studies were adjusted for birth decade and sex following the approach described in ([Denny *et al.*, 2011]).

PAGE obesity study. The second dataset is from the Population Architecture using Genomics and Epidemiology (PAGE) study ([Matise *et al.*, 2011]). It consists of 37,823 European Americans and 15,415 African Americans, and spans across six different sites: i) the Atherosclerosis Risk in Communities Study (ARIC), ii) the Coronary Artery Risk in Young Adults (CARDIA), iii) the Cardiovascular Health Study (CHS), iv) the Epidemiologic Architecture for Genes Linked to Environment (EAGLE) accessing the National Health and Nutrition Examination Surveys (NHANES), v) the Multiethnic Cohort (MEC), and vi) the Women's Health Initiative (WHI). Local-site studies were completed following the processing procedures described in ([Fesinmeyer *et al.*, 2013]).

EAGLE obesity study. The third dataset was from the Epidemiologic Architecture for Genes Linked to Environment (EAGLE) group, which is a sub-site of PAGE ([Haiman *et al.*, 2012]). EAGLE itself can be divided into two sub-studies associated with the National Health and Nutrition Examination Surveys (NHANES): i) NHANES III and ii) NHANES 1999-2002. This study contains 14,998 DNA samples and spans several ethnicities (e.g., non-Hispanic white, non-Hispanic black, Mexican-American, and others).

S4 Computational Accuracy in a Controlled Setting

In the main text, we pointed out that the secure computation results were close to the "true" association values (from the original publications), but not perfect. We note that in replication studies, it is not uncommon for there to be minor variability in the statistical routines performed. Thus, to present a more controlled evaluation on the computational accuracy, we performed additional comparisons with a non-secure meta-analysis as the baseline (i.e., results taken directly from the widely-used METAL software [Willer *et al.*, 2010] instead of using the reported results from their original studies).

The comparisons are reported as QQ-plots on a negative logarithmic scale (Fig. S3). It can be seen that our secure results are extremely close to the non-secure results. Specifically, a linear regression with the y-intercept forced to zero, yielded both a slope and correlation coefficient of $\sim$1.000 for all three datasets. These results lend further evidence that our SecureMA protocol is accurate.

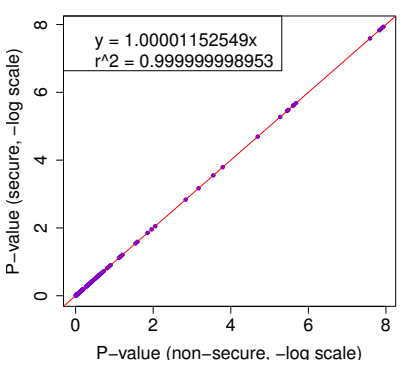
S5 Details on Securely Computing Meta-analysis

Here we provide the technical details regarding the various sub-protocols underpinning the secure meta-analysis computation.

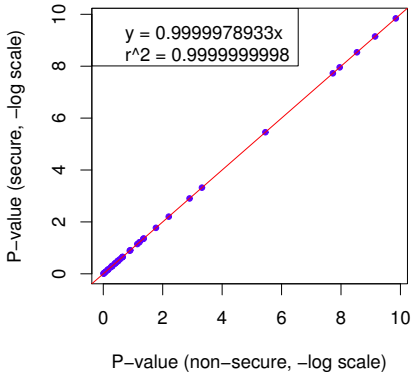
S5.1 (Threshold) Paillier Encryption Scheme

In the main text, we briefly mentioned that submissions of local-site statistics are protected leveraging encryption. Here we describe the relevant cryptographic schemes in SecureMA.

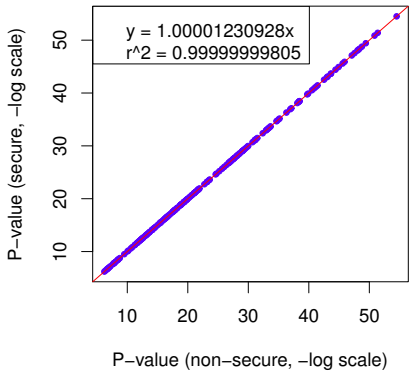
We leverage a "semantically secure" homomorphic public-key encryption (HPE) framework. Generally speaking, in a public-key encryption system, a person, say Alice, generates two keys: 1) a *public key*, which is made available to another entity, say Bob, who wishes to communicate



(a) eMERGE



(b) PAGE



(c) EAGLE

Figure S3: A controlled comparison of the P-values derived from a non-secure and secure meta-analysis protocol. These results are based on (a) 100 SNPs from eMERGE, (b) 40 SNPs from PAGE, and (c) 216 SNPs from EAGLE.

messages to Alice in an encrypted manner (i.e., the ciphertext) and 2) a *private key*, which is known only to Alice and is applied to decrypt the ciphertext sent by Bob.

An encryption scheme is said to be semantically secure when it is infeasible for an adversary (with finite computational capability), say Mallory, to gain knowledge about a message when it observes a ciphertext and the corresponding public encryption key. This property implies that even when the same message is encrypted multiple times, the ciphertexts will be indistinguishable to Mallory. In other words, if Bob and Charlie encrypt the same genotype-phenotype association statistics, say a regression coefficient with a value of 10 using the same public key, then the resulting ciphertexts will appear to be different. This mechanism further enhances the security of the encryption scheme (e.g., by protecting against attacks which leverage a pre-computed lookup table with raw data and their corresponding encryptions).

In addition, we require the encryption framework to possess an "additive homomorphic" property. This enables the computation of the encrypted sum of two messages to be completed using only the corresponding ciphertexts (e.g., without decryption).

The Paillier crypto-system ([Paillier *et al.*, 1999]) is a probabilistic public key encryption protocol with a high confidentiality guarantee. Its additive homomorphic property enables direct support for several arithmetic operations, including addition and multiplication by a constant value, over encrypted data.

The following provides a basic introduction to Paillier encryption:

- Keys: Let $n = pq$, where p and q are large prime numbers, and $\lambda = lcm(p-1, q-1)$, where $lcm(.)$ denotes the function for least common multiple. We define function $L(x) = (x-1)/n$ and let g be an integer, such that $gcd(L(g^{\lambda}\, mod\, n^2), n) = 1$, where $gcd(.)$ is the function for greatest common divisor. The public and private cryptographic keys then consist of (n, g) and (p, q, λ), respectively. Note that there is only one private key.

- Encryption: The encryption of a message m (e.g., the value of a regression coefficient) into a ciphertext c is accomplished by $E(m, r) = g^m r^n\, mod\, n^2$, where g and n correspond to the public key, and r is a random value. For future reference, we will simply refer to this value as $E(m)$.

- Decryption: The decryption of a ciphertext c is computed as:

$$D(c) = \frac{L(c^{\lambda} \mod n^2)}{L(g^{\lambda} \mod n^2)} \mod n$$

Public key crypto-systems are vulnerable in that the system can be compromised if a private key is disclosed (either unintentionally or maliciously). To enhance the security of the system and ensure that the participants cannot easily violate the protocol, a private key can be split into l distinct "shares", where each share is provided to a different participant (e.g., a data manager in our protocol). This variation on cryptography is called a "threshold" system because it requires at least w out of the l participants to correctly decrypt information for Alice. When fewer than w participants attempt to decrypt, the system will be unable to reveal the corresponding message. A threshold version of the Paillier crypto-system was introduced in ([Cramer *et al.*, 2001]) and was utilized in our protocol. In practice, we assume that the majority of participants in cryptographic systems are honest and thus, it is unlikely collusion will lead to illegitimate decryption. To achieve good security in practice, we set $w > \frac{2}{3}l$ according to the Byzantine fault tolerance principle [Lamport *et al.*, 1982].

For the purposes of the SecureMA protocol, these participants correspond to the data managers who help maintain the encrypted genotype-phenotype association statistics. To perform

decryption, the participants independently decrypt the result of the meta-analysis to obtain partial decryptions. The scientist (i.e., inquiry issuer) will complete the decryption process by aggregating these partial decryptions.

S5.2 SHARES: Converting Encryptions to Secret Shares

The secure logarithm protocol (a step of secure division) introduced later requires inputs to be in the form of secret shares (i.e., data split and distributed across different participants for protection), while all data in our protocol are encrypted using the Paillier crypto-system. We propose the following *SHARES* sub-protocol to convert Paillier encryptions into two-party secret shares (i.e., two participants collaboratively keep the secret). Given an encryption $E(x)$, the goal is to find two random values x_1 and x_2 (to be held by two participants respectively), such that $x_1 + x_2 = x$. These values are randomized to ensure it is *not* possible to predict the value of one from the other. This is accomplished as follows. First, a data manager generates a random value $rand$ to obfuscate the given (encrypted) value $E(x)$ by computing $E(x + rand)$ (via the secure summation sub-protocol ADD introduced later). The resulting encryption $E(x + rand)$ is then transmitted to the mediator. Later, a decryption process helps obtain the mediator's data share $x_2 = x + rand$, while the data manager holds his share $x_1 = -rand$.

S5.3 Garbled Circuits for Secure Division

In our protocol, we leverage a garbled circuit ([Yao, 1982]) (for secure computation) to perform part of the secure division operation introduced below. This approach allows two participants to collaboratively evaluate an arbitrary function on their individual data without disclosing anything other than the final output . This is enabled by implementing the function to compute as a binary circuit and the security is achieved by randomizing (garbling) the data in the circuit. We design our own circuits and enhance the low-level FastGC framework ([Huang *et al.*, 2011]) for execution (released as open-source software).

S5.4 Secure Arithmetic Operations

The Paillier crypto-system supports secure summation through an additive homomorphic property. The secure addition sub-protocol, ADD, is defined as follows: given two messages m_1, m_2 (and n being the Paillier field size), the encryption of sum $(m_1 + m_2)$ can be computed as:

$$E(m_1 + m_2) = E(m_1) \cdot E(m_2) \mod n^2$$

It is also straightforward to implement multiplication of an encrypted value by a known constant in the Paillier crypto-system. The multiplication-by-a-constant sub-protocol ($MULC$) proceeds as follows. Suppose we are provided with encryption $E(m)$ of message m and need to compute $E(k \cdot m)$, where k is a known constant. This can be accomplished by computing:

$$E(k \cdot m) = (E(m))^k \mod n^2$$

Secure subtraction (SUB sub-protocol) can be achieved by taking advantage of the multiplication-by-constant and addition protocols described above. In brief, given two encryptions $E(m_1), E(m_2)$, we can compute the subtraction as:

$$E(m_1 - m_2) = ADD(E(m_1),\ MULC(m_2, -1))$$

It can further be observed in Equation 1 that a meta-analysis requires the final division of a numerator by a denominator. However, there is no existing protocol for directly computing

the division of two Paillier-encrypted numbers. We therefore choose to convert the division operation (denoted as *DIV* sub-protocol) into a subtraction problem using a secure logarithmic transformation. For simplicity, we denote: $a = \sum_i \beta_i w_i$ and $b = \sum_i w_i$. Via the logarithmic transformation, the goal in Equation 1 becomes:

$$\ln Z^2 = \ln \frac{a^2}{b} = 2 \ln a - \ln b \tag{2}$$

We leverage the secure logarithm sub-protocol described below to compute $\ln a$ and $\ln b$ for the transformed division operation. The final Z^2 can be easily derived by taking the exponential, exp(.), on the final subtraction result.

S5.5 Secure Logarithmic Transformation

As described earlier, secure logarithmic transformation is utilized in our protocol to perform the division operation. Our $\ln x$ transformation builds upon the secure $ln(x)$ sub-protocol in ([Lindell *et al.*, 2000]). Given input x, which is composed of secret shares x_1 and x_2 from two participants (following the *SHARES* sub-protocol), a two-phase process is applied to approximate the logarithm and output two secret shares of the result.

More specifically, x is approximated by 2^y, with a relative error of ϵ :

$$\ln x = \ln(2^y(1+\epsilon)) = y \ln 2 + \ln(1+\epsilon) \tag{3}$$

Based on this representation, approximating $\ln x$ requires securely computing the two terms in Equation 3, which is facilitated by the two-phase process described below.

S5.5.1 Logarithm Phase 1: Rough Estimate via Garbled Circuits

In the first phase, the logarithm $\ln x$ is approximated by 2^y using a garbled circuit evaluation to protect sensitive data. The output of this phase contains two portions, γ and α, each of which is composed of two secret shares obfuscated to prevent disclosure and is scaled up (i.e., multiplied by a power of 2 and truncated) to avoid numbers with decimals and use only integers:

$$\gamma_{true} + \gamma_{rand} = 2^N \cdot y \ln 2 \tag{4}$$

$$\alpha_{true} + \alpha_{rand} = 2^N \cdot \epsilon \tag{5}$$

Equation 4 approximates the first term in Equation 3, which is a rough estimate of $\ln x$. The terms are scaled up to avoid decimal values because the computation is performed over encrypted data, which requires the operands to be integers. Here, the term 2^N is as a scaling factor, where N is the upper bound for the exponent estimate y.

Equation 5 denotes the scaled relative error of the approximation, and will be applied in the next phase to boost the accuracy of approximating Equation 3.

Since a garbled circuit evaluation involves two participants and no meaningful information should be disclosed to any single participant, we adopt random values γ_{rand} and α_{rand} contributed by one of the two participants in the computation for proper protection.

At the end of this process, one participant will hold α_{rand} and γ_{rand}, while a second participant will be in possession of α_{true} and γ_{true}, as illustrated in Equations 4 and 5.

S5.5.2 Logarithm Phase 2: Refined Estimate via Taylor Series

In the second phase, we further refine our $\ln x$ approximation by estimating the second term in Equation 3. This is accomplished via an oblivious polynomial evaluation ([Naor *et al.*, 1999]), such that a secure polynomial from one participant is computed on the data contributed by the other participant without disclosing additional information. To perform the approximation, ϵ is substituted with $\frac{\alpha_{true}+\alpha_{rand}}{2^N}$ (derived from Equation 5). Next, we apply the following Taylor series (with proper scaling up to avoid fractional values):

$$\begin{aligned} &\ln(1+\epsilon)\cdot 2^{Nk} lcm(2,...,k) \\ \approx &\sum_{i=1}^{k} (-1)^{i-1} 2^{N(k-i)} \cdot \frac{lcm(2,...,k)}{i} \cdot (\alpha_{true}+\alpha_{rand})^i \end{aligned} \tag{6}$$

The polynomial on the right side (denoted as $Q(\alpha_{true})$) will be expanded and evaluated leveraging our *MULC* and *ADD* sub-protocols. The result at this point is still encrypted.

S5.5.3 Result Assembly for Logarithm

Based on the results from the previous two phases, the final result of $\ln(x)$ is obtained through an assembly process. First, the γ's in Equation 5 are scaled up by a factor $2^{N(k-1)}lcm(2,..,k)$:

$$(\gamma_{rand}+\gamma_{true})\cdot 2^{N(k-1)} lcm(2,\ldots,k) = y\ln 2 \times 2^{Nk} lcm(2,\ldots,k) \tag{7}$$

Next, the scaled γ's are encrypted and securely summed via Equations 7 and 6:

$$\begin{aligned} &E((\ln(1+\epsilon)+y\ln 2)\cdot 2^{Nk} lcm(2,\ldots,k)) \\ \approx\ &E(\ln x \cdot 2^{Nk} lcm(2,\ldots,k)) \end{aligned} \tag{8}$$

After obtaining the encryptions of scaled-up $\ln a$ and $\ln b$, we can compute the scaled-up $E(\ln Z^2)$ via Equation 2. The final Z-score (in decimal) can easily be derived after decryption and scaling the result back down. And the desired p-value can be obtained following the instruction in Section S2.1.

References

[Kate *et al.*, 2009] Kate,A., Goldberg,I. (2009) Distributed key generation for the internet. *29th IEEE International Conference on Distributed Computing Systems*, 119-128.

[Cramer *et al.*, 2001] Cramer,R. *et al.* (2001) *Multiparty computation from threshold homomorphic encryption*, (Springer, Berlin, Heidelberg), 280-300.

[Denny *et al.*, 2011] Denny,J.C. *et al.* (2011) Variants near FOXE1 are associated with hypothyroidism and other thyroid conditions: using electronic medical records for genome-and phenome-wide studies. *Am. J. Hum. Genet.*, **89**, 529-542.

[Fesinmeyer *et al.*, 2013] Fesinmeyer,M.D. *et al.* (2013) Genetic risk factors for BMI and obesity in an ethnically diverse population: results from the Population Architecture Using Genomics and Epidemiology (PAGE) study. *Obesity*, **21**, 835-846.

[Haiman *et al.*, 2012] Haiman,C.A. *et al.* (2012) Consistent Directions of Effect for Established Type 2 Diabetes Risk Variants Across Populations The Population Architecture using Genomics and Epidemiology (PAGE) Consortium. *Diabetes* **61.6**, 1642-1647.

[Huang *et al.*, 2011] Huang,Y. *et al.* (2011) Faster secure two-party computation using garbled circuits. *Proceedings of the USENIX Security Symposium*, 201.

[Lamport *et al.*, 1982] Lamport,L. *et al.* (1982) The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, **4.3**, 382-401.

[Lindell *et al.*, 2000] Lindell,Y., Pinkas,B. (2000) Privacy preserving data mining. *Advances in Cryptology – CRYPTO 2000*, (Springer, Berlin), 36-54.

[Matise *et al.*, 2011] Matise,T.C. *et al.* (2011) The next PAGE in understanding complex traits: design for the analysis of Population Architecture Using Genetics and Epidemiology (PAGE) Study. *Am. J. Epidemiol.*, **174**, 849-859.

[McCarty *et al.*, 2011] McCarty,C.A. *et al.* (2011) The eMERGE network: a consortium of biorepositories linked to electronic medical records data for conducting genomic studies. *BMC Med. Genomics*, **4**, 13.

[Naor *et al.*, 1999] Naor,M., Pinkas,B. (1999) Oblivious transfer and polynomial evaluation. *Proc. of the ACM Symposium on Theory of Computing*, (ACM, New York), 245-254.

[Paillier *et al.*, 1999] Paillier,P. (1999) Public-key cryptosystems based on composite degree residuosity classes. *Advances in Cryptology – Eurocrypt'99*, (Springer, Berlin), 223-238.

[Willer *et al.*, 2010] Willer,J. *et al.* (2010) METAL: fast and efficient meta-analysis of genome-wide association scans. *Bioinformatics*, **26**, 2190-2191.

[Yao, 1982] Yao,A.C. (1982) Protocols for secure computations. *Foundations of Computer Science*, **82**, 160-164.