Bioinformatics, 33(15), 2017, 2273–2280
doi: 10.1093/bioinformatics/btx193
Advance Access Publication Date: 4 April 2017
Original Paper

OXFORD

Genome analysis

SQC: secure quality control for meta-analysis of genome-wide association studies

Zhicong Huang[1], Huang Lin[1], Jacques Fellay[2,3], Zoltán Kutalik[3,4] and Jean-Pierre Hubaux[1,*]

[1]School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland, [2]School of Life Sciences, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland, [3]Swiss Institute of Bioinformatics, 1015 Lausanne, Switzerland and [4]Institute of Social and Preventive Medicine, University Hospital Lausanne (CHUV), 1011 Lausanne, Switzerland

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on February 21, 2016; revised on February 27, 2017; editorial decision on March 25, 2017; accepted on March 31, 2017

Abstract

Motivation: Due to the limited power of small-scale genome-wide association studies (GWAS), researchers tend to collaborate and establish a larger consortium in order to perform large-scale GWAS. Genome-wide association meta-analysis (GWAMA) is a statistical tool that aims to synthesize results from multiple independent studies to increase the statistical power and reduce false-positive findings of GWAS. However, it has been demonstrated that the aggregate data of individual studies are subject to inference attacks, hence privacy concerns arise when researchers share study data in GWAMA.

Results: In this article, we propose a secure quality control (SQC) protocol, which enables checking the quality of data in a privacy-preserving way without revealing sensitive information to a potential adversary. SQC employs state-of-the-art cryptographic and statistical techniques for privacy protection. We implement the solution in a meta-analysis pipeline with real data to demonstrate the efficiency and scalability on commodity machines. The distributed execution of SQC on a cluster of 128 cores for one million genetic variants takes less than one hour, which is a modest cost considering the 10-month time span usually observed for the completion of the QC procedure that includes timing of logistics.

Availability and Implementation: SQC is implemented in Java and is publicly available at https://github.com/acs6610987/secureqc

Contact: jean-pierre.hubaux@epfl.ch

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Large human genomic databases enable researchers to conduct whole genome analysis on genomic variations and their associations with phenotypic traits. For instance, in a case-control GWAS scenario, researchers can compute χ^2-statistics to reveal whether some genetic variants and a given disease are independent or not up to a significance level. In another case, researchers might use linear regression for modeling the relationship between a continuous phenotypic trait (dependent variable) and a genetic variant (explanatory variable). Although small-scale GWAS can identify variants associated with diseases, these variants explain only a small portion of the risk variability for many diseases. Discovery of new significant associations requires much larger sample sizes. However, the privacy-sensitive nature of human genomic data prevents research groups from sharing genomic data of individual participants, especially when it comes to cross-border collaboration. Therefore, it is

usually hard or even impossible to directly combine data from several groups to enlarge sample sizes. Nevertheless, to increase statistical power and reduce false-positive findings, researchers can use GWAMA for the synthesis of results from multiple independent studies, thanks to the fact that GWAMA requires only aggregate statistics rather than individual genomic data from each study.

Generally speaking, GWAMA can be divided into two phases: the data-quality-control phase and the meta-analysis phase. Quality control usually takes place before meta-analysis, in order to filter out the studies whose data have quality issues and thus are not appropriate for meta-analysis. Both phases involve multiple processing steps of summary data from individual studies. Quality control is an important procedure for ensuring the quality or even the correctness of a meta-analysis. Our proposed system is built upon a comprehensive protocol describing the state-of-the-art procedure to conduct data QC for large-scale GWAMA (Winkler *et al.*, 2014). This particular framework has incorporated all the necessary QC steps proven to be helpful in practice. In this framework, each study shares its own association statistics for each genetic variant, such as allele frequencies, *P*-value and effect size estimates with standard errors, which are delivered to the analysts. The analysts will apply statistical techniques to detect potential errors in the aggregate data. For example, quality control will check whether the same allele is designated as the effect allele for all studies, in order to prevent that one study erroneously declares the other allele as the effect one. These quality issues could lead to unexpected or incorrect conclusions for the variant associations if the data were used blindly. We should emphasize that the quality control procedure is designed to detect systematic errors of a study, hence if such an error exists, many data points (i.e. statistics of single nucleotide variants (SNVs)) will deviate significantly from the expected ones, and thus this phenomenon can be easily observed by plotting the data.

However, even aggregate statistics may contain sensitive information from each participant; it has been demonstrated that individual information (e.g. cohort membership) can still be inferred from the published summary data (Homer *et al.*, 2008; Im *et al.*, 2012; Jacobs *et al.*, 2009; Lumley and Rice, 2010; Sankararaman *et al.*, 2009; Visscher and Hill, 2009; Wang *et al.*, 2009). As a matter of fact, the practice of publishing certain aggregate data has been discouraged since the revelation of the privacy breach of public summary statistics. For instance, the NIH removed aggregate genomic data (e.g. allele frequencies and *P*-values) from open-access databases, and urged the scientific community to take precautions before sharing any aggregate GWAS data (Zerhouni and Nabel, 2008). These statistical inference attacks are based on a similar idea: if an adversary has access to a large number of SNVs (both the victim's SNVs and the aggregate results of the SNVs), it can gain strong enough statistical power to tell whether the victim is a contributor to the dataset. If a dataset contains sensitive health information about participants such as their HIV status, such an inference is a serious privacy breach.

The privacy issue of sharing plaintext aggregate statistics resides in that they reveal aggregate statistics for a high number of SNVs (we call these *SNV-level statistics*, formally defined in Methods), and these statistics are exactly the input to a quality control procedure and a meta-analysis model. A secure approach is needed in order to complete the procedure without revealing the SNV-level statistics of any single study. A secure protocol for the meta-analysis phase has been proposed by operating on encrypted study statistics such that only the result of the meta-analysis is revealed (Xie *et al.*, 2014). However, as we shall see, quality control is a fundamentally more complicated procedure, and protecting this phase remains an open problem. In the proposed *secure quality control* (SQC), it guarantees that the analysts will receive nothing other than the final quality measurements. To the best of our knowledge, this is the first comprehensive solution for secure quality control for meta-analysis of genome-wide association studies.

2 Materials and methods

2.1 SNV-level aggregate statistics in quality control

A meta-analysis protocol usually requires the aggregate association statistics such as effect allele frequency (f), beta estimate (β), standard error (e) and *P*-value (p), for a predefined set of variants. We denote the set of *SNV-level statistics* for study j as $\{(f, \beta, e, p)_{i,j}\}$, $i \in \{1, 2, \ldots, n\}$, where n is the number of variants. We use these four types of statistics for demonstrating our solution in a secure quality control pipeline, but it is easy to extend the solution to handle other types of statistics. We also use N_j to denote the sample size (number of participants) of study j, and N_j is usually a public value.

2.2 Secure multi-party computation

A well-established cryptographic approach called secure multi-party computation (SMC) guarantees the following property (without loss of generality, we use two parties throughout the paper):

(**Secure two-party computation**). Suppose there are two parties, Alice and Bob, who have their own private data x and y, respectively, and want to securely compute a function $F(x, y)$. The SMC technique enables Alice and Bob to execute a protocol that outputs the result $R = F(x, y)$, without Alice learning any other information on Bob's input y, nor Bob learning any other information on Alice's input x.

Using this technique in the quality-control scenario, we can imagine splitting each statistics into two random and individually uninformative shares (secret shares), one stored by Alice and one stored by Bob. The quality control can then define a set of functions to be securely computed on the two shares. The underlying statistics are not revealed as long as Alice and Bob do not collude, which is realistic in practice if, for example, the two parties are chosen to be private, competitor companies (e.g. a Google cloud server and an Amazon cloud server). A detailed system setup of our solution is described later.

2.3 Differential privacy

Informally, differential privacy guarantees that a person's contribution to a dataset does not significantly alter the output distribution of a statistical query on the dataset. We can achieve differential privacy by adding Gaussian noise with standard deviation that depends on two privacy parameters ε and δ (Dwork and Roth, 2014), which is discussed in more detail in Supplemental Methods. Smaller ϵ and δ imply higher noise variance and stronger privacy. Although differential privacy reduces the accuracy of GWAS applications, we will justify its use in the case of quality control; in short, the way we apply differential privacy in quality control **does not** affect the utility of the data in the meta-analysis phase.

2.4 Adversary model and system structure

We adopt the standard security notion of semi-honest model for secure computation, which implies corrupted parties do not deviate from the protocol specification but might try to gather information out of the protocol. We assume the adversary knows the set of genomic variants SNVs $\{\mathrm{SNV}_1, \mathrm{SNV}_2, \ldots, \mathrm{SNV}_n\}$ shared by all studies. The storage and computation are outsourced to two non-colluding, semi-honest cloud providers (A and B). Each cloud

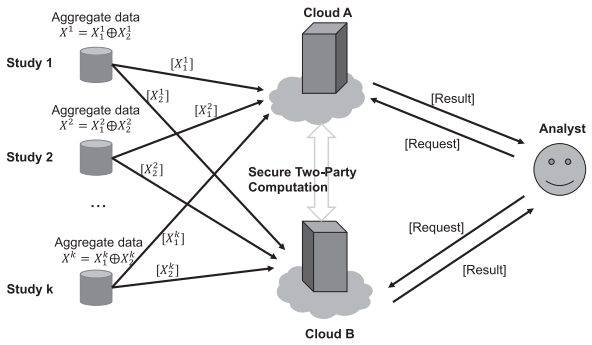

Fig. 1. System architecture. Each study splits its summary statistics into two secret shares that are sent to two non-colluding cloud servers (e.g. Google and Amazon). At the request of an analyst, the two clouds perform secure two-party computation to verify data quality or perform meta-analysis. In both cases, only the final results, either quality measurements or meta-analysis results, are revealed to the analyst. '[X]' represents the encryption of 'X'. Essentially, all the communication links are protected with authenticated symmetric encryption that is a well-established security feature in the Internet (e.g. when we use HTTPS instead of HTTP)

server could be equipped with a cluster of machines to enable parallel secure computation. The system works as follows (Fig. 1):

Step 1. Each study splits its aggregate statistics X (e.g. $f_{i,j}$) into two secret shares: $X = X_1 \oplus X_2$ (bit-by-bit XOR operation on the whole value), where X_1 is randomly chosen. X_1 is encrypted and sent to cloud A, whereas X_2 is encrypted and sent to cloud B. '[X]' represents the encryption of 'X' in Figure 1. Essentially, in our system, all the communication links are protected with authenticated symmetric encryption that is a well-established security feature in the Internet (e.g. when we use HTTPS instead of HTTP). The protection of X_1 in cloud A and X_2 in cloud B will be detailed in the Discussion section.

Step 2. An analyst sends a request to cloud A for checking data quality.

Step 3. The two clouds execute a secure two-party computation protocol to perform the quality control procedure on the secret shared input data of each study. The quality measurements are sent back to the analyst.

Step 4. If there is no quality issue, the analyst sends another request to cloud A for running a meta-analysis approach, and Step 5 follows. Otherwise, the analyst contacts the study whose data present quality issues, and the study administrator resolves these issues and goes back to Step 1.

Step 5. The two clouds execute another different secure two-party computation protocol to perform meta-analysis on the data. The result is sent back to the analyst. This step has been studied in previous work (Xie *et al.*, 2014) and is not the focus of this work.

2.5 Secure computation procedures

SQC is composed of several relevant procedures, each of which has an added value to the protection of the data. Unless otherwise specified, all the following procedures are performed in secure two-party computation on two secret shares of the input, which corresponds to Step 3 of the above system workflow.

2.5.1 Procedure 1: Variant hiding by oblivious sorting

As discussed before, the adversary knows the list of targeted SNVs in the study, which is an assumption for the aforementioned inference attacks. On the other hand, in quality control, the ordering of SNVs does not matter, therefore we obliviously sort the statistics of the SNVs. In fact, this step hides the original ordering (e.g. based on positions on genome), which has already been used by previous work on meta-analysis (Singh *et al.*, 2013). Procedure 1 serves as a preliminary step of SQC. We use bitonic sorting (Batcher, 1968) to guarantee obliviousness: the algorithm does not reveal information about the input (Knuth, 1998).

2.5.2 Procedure 2: Adding gaussian noise

Releasing original statistics, especially allele frequencies, results in a serious privacy leakage, as we show in the Results section. In this framework, we achieve differential privacy through adding Gaussian noise. Each cloud server independently draws a random sample from the Gaussian distribution, computes the addition of the two samples via SMC, and adds the result to the statistics. We can provide a theoretical guarantee that the noisy result is differentially private against the two servers that engage in SMC and any external adversary. More details about motivation, related work and a formal proof can be found in Discussion and Supplemental Methods.

Note that this procedure is executed on the fly for checking data quality without affecting the stored data, hence the original data can still be used at a later stage (e.g. meta-analysis) **without any accuracy loss**.

2.5.3 Procedure 3: Precision truncation

Truncating the precision of statistics serves two purposes in this framework: First, apart from differential privacy, it is an alternative way to add noise to the data; second, it helps to reduce the number of SNVs that can be revealed, undermining the power of inference attacks. The second feature will be discussed in oblivious de-duplication. In Procedure 3, we use fixed-point representations (more efficient than floating-point operations), reserve a predefined precision and round the statistics to the nearest reduced-precision representation. For a fixed-point statistics x that has w-bit width (total number of bits) and t-bit offset (number of bits for the fractional part), we preserve r bits after leading zeros.

2.5.4 Procedure 4: Oblivious de-duplication

After reducing precision, we might observe a high number of repetitive statistics. For example, if we have 1 million SNVs and preserve 10-bit precision (around 3 decimal digits), we could expect on average 1000 duplicates per SNV statistics. Not only is this frequency information useless for analysts to construct quality-control plots, but also a potential privacy leakage to adversaries. In Procedure 4, we perform an oblivious sorting to cluster duplicated items together, preserve one of them by setting the others to the infinity element, and perform another round of oblivious sorting in order to discard the infinity elements.

2.6 Secure quality control protocols

Using the above procedures, we propose our methodology to run several quality-control protocols (Winkler *et al.*, 2014) in a secure and privacy-preserving way. In absence of the secure protocols described below, a large number of original SNV-level statistics would be revealed, making them vulnerable to inference attacks, hence each SQC protocol is necessary to avoid leaking those (precise) SNV-level statistics.

2.6.1 Protocol 1: Secure SE-N plot

An analyst can plot the median standard error (SE) across all SNVs against the square root of the sample size (N) for each study, in order to identify analytical problems, e.g. inconsistent regression models. More specifically, for study j with sample size N_j (public value), the analyst finds the median in the corresponding list of standard errors: $e_{1,j}, e_{2,j}, \ldots, e_{n,j}$. We denote this median with

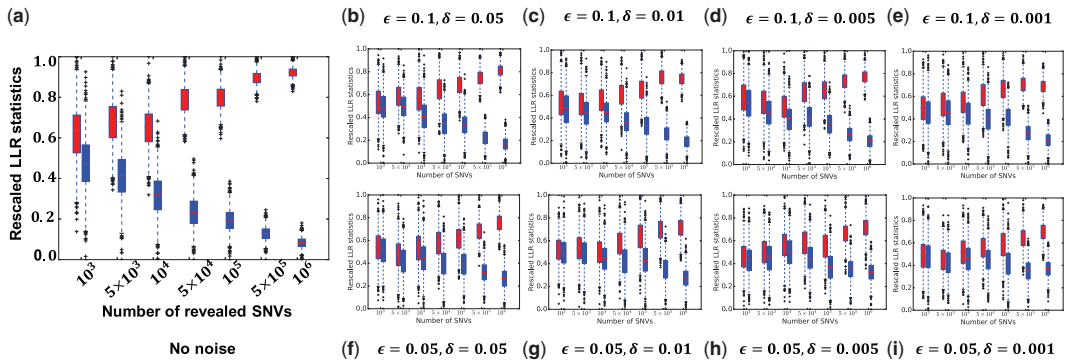

Fig. 2. Rescaled LLR statistics. The nine experiments are conducted on simulated datasets after adding different Gaussian noises by choosing parameters ϵ and δ. In each experiment, for the case group (red) and the test group (blue), we reveal different number of SNVs, from left to right: 1000, 5000, 10000, 50000, 100000, 500000, 1000000. In general, we observe that revealing more SNVs makes it easier to separate the two groups based on D statistics. Adding Gaussian noises by imposing smaller values of ϵ and δ helps to mitigate the effect of this attack. We could choose these parameters by basing them on the quantification of the resulting attack powers, which will be detailed in Figure 3 and Figure 4: for example, releasing 1000 SNPs after adding the least amount of differential privacy ($\epsilon = 0.1,\ \delta = 0.05$) is appropriate because the attack power is relatively weak in this case (Color version of this figure is available at *Bioinformatics* online.)

$e_{median,j}$. If the study data have no quality issues, N_j is proportional to the inverse of $e_{median,j}$ by a public constant c:

$$N_j \approx \frac{c}{e_{median,j}}.$$

Therefore, the study-specific data points of the SE-N plot should resemble a straight line. Protocol 1 applies oblivious sorting (Procedure 1) to each list of standard errors and chooses the corresponding median afterwards. We note that there is no need to go through Procedure 3 and Procedure 4 in this protocol because the output for each study contains only one overall statistics (median) across all SNVs, unlike the following two protocols.

2.6.2 Protocol 2: Secure EAF plot

The EAF (effect allele frequency) protocol plots the reported EAFs from a study against another study, which could pinpoint issues such as a wrong strand, allele miscoding, etc. For example, a study might consistently label the wrong allele as the effect allele, leading to wrong allele frequencies. If the two studies j_1 and j_2 have no quality issues and their samples come from the same ancestry, then $f_{i,\ j_1}$ should be close to f_{i,j_2}, $\forall i \in \{1, 2, \ldots, n\}$. The two sets of allele frequencies are taken as input to Protocol 2, which first reduces the precision of allele frequencies (Procedure 3) and then de-duplicates the pairs of EAFs (Procedure 4) from the two studies.

2.6.3 Protocol 3: Secure P-Z plot

The P-Z plot (*P*-value and Z-statistics) can reveal analytical problems of the computation of beta estimates, standard errors or *P*-values. More specifically, for SNV_i of study j, the Z-statistics is computed as: $Z - \text{statistics} = \frac{\beta_{i,j}}{e_{i,j}}$. The corresponding reported *P*-value $p_{i,j}$ should be close to the *P*-value associated with the above Z-statistics. In the first step of Protocol 3, the two servers perform a secure division protocol to compute the Z-statistics, which is known to be a costly operation in secure computation. We benchmark this step in the Results Section. The resultant Z-statistics are paired with the corresponding *P*-values to go through a similar precision-reduction and de-duplicated procedure as in Protocol 2.

3 Results

We use ObliVM-GC (Liu *et al.*, 2015a) for our backend secure two-party computation. This framework provides a Java library that helps programmers to build secure-computation protocols in a circuit representation. Nayak *et al.* wrap it into a parallel secure-computing paradigm that we use to scale up our solution (Nayak *et al.*, 2015). Our experiment uses two sets of data:

- Real data: GWAS data from 10 studies, each of which builds a linear regression model between human height and each of roughly three million SNVs (Wood *et al.*, 2014).
- Simulated data: Following another work (Simmons and Berger, 2015), we choose a study size (N) and a number of SNVs (n). For each SNV, we pick a random number in the range of 0.05 to 0.5 as the minor allele frequency. We then generated the genotypes of N individuals independently. For example, we choose N = 1000 and n = 3 000 000, which is similar to the size of our real data.

3.1 Privacy and utility analysis

Log likelihood ratio test. As we apply differential privacy to the allele frequencies that are the major vulnerability exploited by inference attacks, we experimentally analyze the privacy of our solution by using the attack power as a metric. The log likelihood ratio test is one of the most powerful inference attacks for individual detection in a pool (Sankararaman *et al.*, 2009). To construct the test, we assume a reference population with specified allele frequencies for variants, and we sample a pool of 1000 individuals from this reference population. The null hypothesis of the test is that a tested individual is not in the pool. An LLR statistic for the tested individual is computed based on the genotype frequencies of the reference population and the pool, in order to tell whether the individual is included in the pool.

The larger the LLR statistic is, the higher is the probability that individual i is in the case group (i.e. lower *P*-value for the test). To assess the attack power, we construct two groups: one case group (same as the pool) and one test group, for nine different experiments. We then calculate the LLR statistics for each individual in the two groups, rescale them to the range [0, 1], and analyze their distribution (Fig. 2). A wider gap between the case distribution and the test distribution indicates a higher attack power. From the results, it is evident that privacy can be enhanced by requiring a higher level of differential privacy and releasing fewer SNVs.

Note that the absolute values of the LLR statistics are invisible in Figure 2, due to the rescaling operation; in fact, for the eight

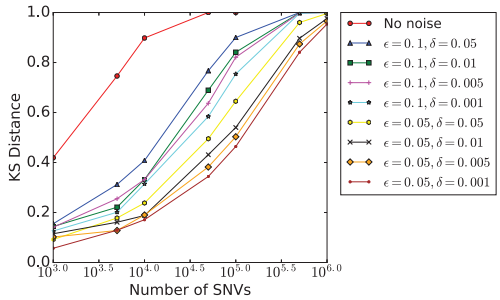

Fig. 3. Kolmogorov–Smirnov distances between case LLR statistics and test LLR statistics. The distances correspond to the visualized gaps in Figure 2. The larger the distance is, the stronger the attacker power is, hence the less the privacy is. Adding the least amount of differential privacy ($\epsilon = 0.1,\ \delta = 0.05$) in SQC can already greatly reduce the attack power and strengthen the privacy

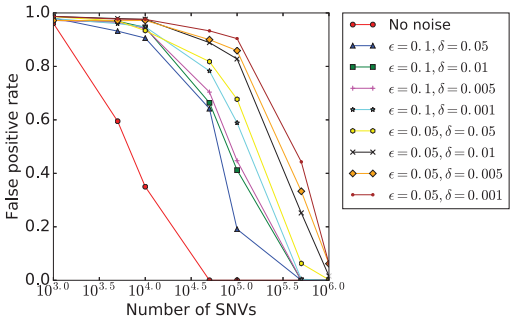

Fig. 4. False positive rates of inferring an individual in the case group when setting the true positive rate to be 99%. The nine plots correspond to the experiments in Figure 2

experiments of adding noise, the LLR statistics are extremely small (hence *P*-values close to 1). This is because the effect of adding noise, even with the least amount of differential privacy ($\epsilon = 0.1,\ \delta = 0.05$), dominates over the effect of sampling, which indicates that the tested individual is actually closer to the reference population than to the 'noisy' pool. However, we can still observe the interesting relative difference between the case group and the test group. To quantify the attack power, we measure the gap between the case distribution and the test distribution of Figure 2 by using Kolmogorov–Smirnov distance (instead of performing a Kolmogorov–Smirnov test to tell whether the two samples are drawn from the same distribution, we are more interested in measuring the distance between the two). The result is illustrated in Figure 3. A larger distance indicates less privacy, because it would be easier for an attacker to differentiate the two groups. We observe that adding the least amount of differential privacy in SQC can already greatly reduce the attack power and strengthen the privacy.

Most of the time, an attacker does not have access to genotypes of a large group of individuals. But assuming the attacker has the genotype of a victim, he must make an assessment about whether the victim is in the pool based on one LLR statistic. Note that using the *P*-value for the assessment is not effective in our scenario for the same reason discussed above. From an attacker's point of view, Chen *et al.* define a classifier with a threshold τ: if $LLR(i) \geq \tau$, then individual i is classified to be in the case group, otherwise it is in the test group (Chen *et al.*, 2012). To have a high true positive rate, we set a low τ value, e.g. at the lowest 1 percentile of the case group (99% true positive rate). We observe the false positive rates in the test group for the above nine experiments (Fig. 4). If the data is

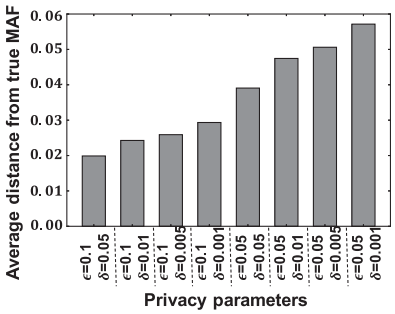

Fig. 5. Average distance from original MAF after adding different Gaussian noises

released without adding noise, an attacker can easily achieve low false positive rates (i.e. effective attack) even if only a small number of SNVs are released. For example, the false positive rate is less than 40% when only 10 000 SNVs are released. Differential privacy adding the lowest amount of noise in our experiment (i.e. $\varepsilon = 0.1,\ \delta = 0.05$) can effectively thwart the attack. With stronger differential privacy, more SNVs can be released at the same false positive rate for an attacker. Nevertheless, we will see that with SQC, only a small number of SNVs need to be released in order to achieve good utility, hence we can add a low level of Gaussian noise for differential privacy, e.g. $\varepsilon = 0.1,\ \delta = 0.05$.

Effects of precision *r*. From the above results, we observe that the number of SNVs used in the attack significantly influences the attack power. For example, when releasing one million SNVs, there is almost no privacy at all for all the privacy parameters in the experiments; but the strongest privacy parameters ($\varepsilon = 0.05,\ \delta = 0.001$) that we use already have a non-trivial negative impact on the utility of the data (Fig. 5). Therefore, it is crucial that our SQC protocols output a relatively small number of SNVs after de-duplication, hence achieving a good trade-off between privacy and utility. This number actually depends on the precision r in Procedure 3 used by Protocol 2 (analogous to Protocol 3): The higher the precision is, the more EAF pairs the protocol outputs. Roughly speaking, the maximum number of de-duplicated EAF pairs for precision r is 4^r.

We evaluate the effectiveness of our approach by setting different combinations of parameters ($\varepsilon,\ \delta,\ r$), and observe the patterns of EAF plots in Figure 6. We also calculate the false positive rates (at 0.99 true positive rate) of LLR test on the output EAF data by assuming, pessimistically, that the adversary is able to link the EAF data back to their SNV identifiers and then perform LLR test. As the precision increases, the number of de-duplicated EAF pairs grows exponentially and thus the false positive rate drops accordingly, but the plot is better preserved. Nevertheless, we can still observe the plot pattern even at a low precision $r = 5$, but it fades out when we further decrease the precision because points become sparse. By observing the distinction between Figure 6-b and Figure 6-c, the differential privacy parameter ε has a significant influence on the plot pattern, and $\varepsilon = 0.1$ is a reasonable choice; in contrast, parameter δ has a relatively modest effect on the pattern. To guarantee a reasonable privacy level (i.e. high false positive rate) and preserve useful plot patterns, it is a good choice to set $\varepsilon = 0.1$ and $r = 5$ or 7, and to fine tune the result by choosing δ.

3.2 Runtime analysis

Single machine. Secure computation provides strong security, but this comes at the cost of a high computational overhead. Even with

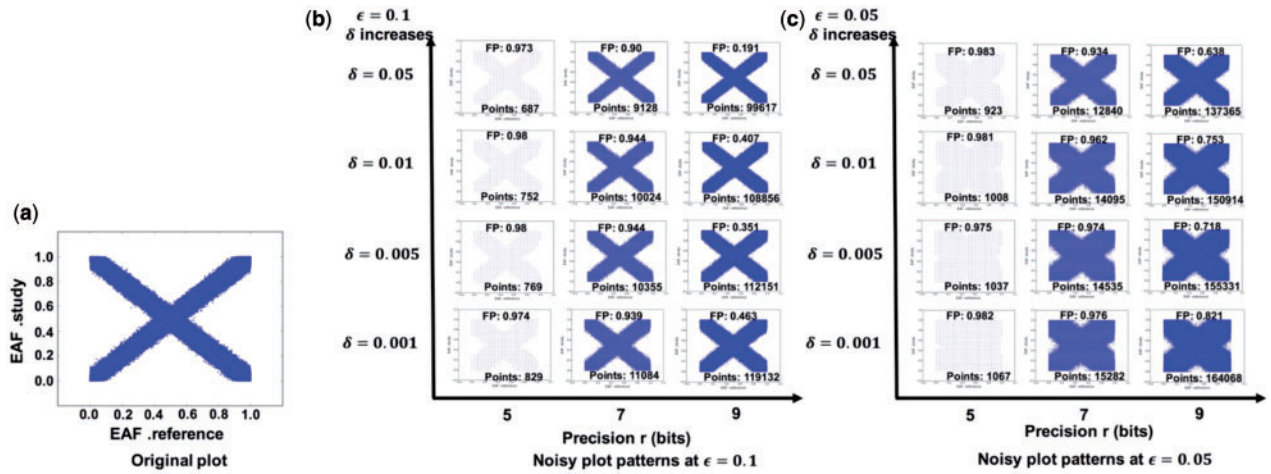

Fig. 6. An example EAF plot and its noisy plot patterns for different levels of differential privacy and different precisions. (**a**) The original plot shows a study in which a fraction of the effect alleles was mis-specified. (**b**) Noisy plots when fixing $\epsilon = 0.1$ and changing δ and r. (c) Noisy plots when fixing $\epsilon = 0.05$ and changing δ and r. The number (e.g. 'Points: 687') at the bottom of each noisy plot is the number of EAF pairs output by SQC, namely, the number of points in the plot. The number (e.g. 'FP: 0.973') on top of each noisy plot indicates the false positive rate by applying the LLR test in Figure 4, when we pessimistically assume the strongest adversary that is able to link the EAF data back to their SNV identifiers. Plot patterns are well preserved when $\epsilon = 0.1$, while δ has a relatively modest effect and could be used to fine tune the pattern. At a low precision $r = 5$, the scatter plot is sparse and looks faded. As the precision becomes higher, the number of de-duplicated EAF pairs grows exponentially, and the false positive rate drops accordingly. An interesting phenomenon is that when δ increases (adding less noises), points deviate less from their original positions (Figure 5) and are more concentrated, hence the number of points decreases monotonically. However, the false positive rate is not monotonically decreasing when increasing δ, because fewer points lead to higher false positive rates (Figure 4). For example, at $\epsilon = 0.1$ and $r = 9$, when increasing δ from 0.005 to 0.01, its effect on the false positive rate is weaker than the effect of decreased number of points, hence the false positive rate becomes higher. The SQC framework can provide a reasonable trade-off by setting $\epsilon = 0.1$, $r = 7$, and then fine-tuning δ

current techniques and implementation, the slowdown of secure computation is as high as thousands or even hundreds of thousands of times, compared with non-secure, plaintext computation (Liu *et al.*, 2015a). The following results are obtained on a single machine with Intel Core i7 processor clocked at 3.1 GHz and 16 GB of RAM.

Secure operation benchmark. Before executing the whole SQC protocols, we benchmark the different core computation procedures on a small set of 1000 SNVs. Table 1 shows the result of benchmarking. In secure computation based on garbled circuits, another measurement of performance is the number of AND gates generated during computation; this number is linearly proportional to the running time and communication cost. Oblivious sorting has a complexity of $O(n\log^2 n)$ (Batcher, 1968), whereas all other procedures have a linear complexity as the number of SNVs increase. For example, with sequential implementation, we estimate it will take around 7.5 hours to accomplish oblivious sorting of 3 million SNV statistics of 64-bit representation. We also see that secure division incurs a high overhead, especially when bit length of numerator (denominator) increases; indeed, the complexity of a secure division is $O(l^2)$ for two numbers of l bits. In our SQC framework, secure division is only used for beta estimates and standard errors that are represented by 32-bit fix-point numbers, hence it will take 5.46 seconds on average to execute secure division for 1000 SNVs. This is roughly the same cost as oblivious sorting on 64-bit numbers, which takes place in Protocol 2, but less than oblivious sorting on 96-bit numbers, which takes places in Protocol 3.

Small-scale experiments. To show the efficiency of each protocol on a single machine, we perform experiments with real data on small scales, varying from input of 1000 SNVs to 10 000 SNVs. In Figure 7-a, the secure P-Z protocol dominates the running time over the other two protocols, because of the secure division operations on 32-bit numbers and oblivious sorting on 96-bit numbers (64 bits for P-values and 32 bits for Z-statistics). Nevertheless, with a sequential implementation, these secure computation protocols seem impractical to run on a large-scale dataset that contains one million SNVs.

Table 1. Benchmark results of the computation steps

Procedure	Bits	Time	AND gates
Oblivious sorting	32	3.11 s	1726976
	64	5.94 s	3453952
	96	8.83 s	5180928
Differential privacy	32	0.813 s	192000
	64	1.472 s	384000
	96	2.126 s	576000
Precision reduction	32	3.57 s	2015000
	64	6.7 s	4127000
	96	9.84 s	6239000
De-duplication (without sorting steps)	32	0.26 s	94905
	64	0.4 s	190809
	96	0.57 s	286713
Secure division	32	5.46 s	3230000
	64	20.09 s	12606000
	96	46.19 s	28126000

Note: The results are conditioned on 1000 SNVs. Oblivious sorting has a complexity of $O(n\log^2 n)$, whereas all other procedures have a linear complexity as the number of SNVs increase.

Parallel SQC: Due to the heavy overhead of secure computation and the non-trivial complexity of running the whole SQC protocols, it would be impractical to deploy the solution on a single machine with a sequential implementation. Building upon the ObliVM-GC backend, Nayak *et al.* propose a parallel secure-computation framework (GraphSC) that parallelizes graph-based algorithms and scales well in a cluster environment. We reconstruct SQC with the primitives provided by GraphSC, and deploy the system on a cluster of machines. The major benefit of using this paradigm is that oblivious

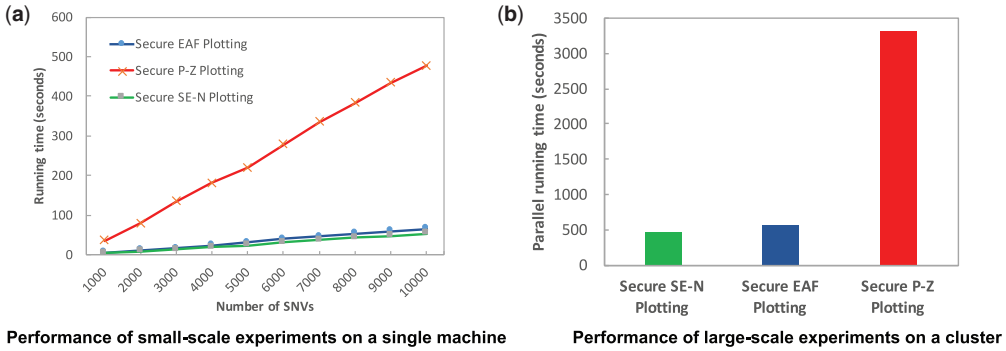

Fig. 7. Runtime performance of three SQC protocols. (**a**) Running time of three protocols on small real datasets. It is measured on a single Intel Core i7 processor clocked at 3.1 GHz and 16 GB of RAM. We construct small datasets, varying from 1000 SNVs to 10000 SNVs, from the real data that contains 3 million SNVs. Secure P-Z plotting takes more time than the other two protocols because of secure division and oblivious sorting on numbers of 96 bits. This sequential implementation, especially due to sequential oblivious sorting, is not efficient enough to run on large datasets. (**b**) Parallel performance on two private clouds. Each machine has 8 cores clocked at 2.5 GHz and 32 GB of RAM, and each cloud has 8 machines (64 cores in total). Any two cores inside a cloud can communicate with each other, whereas a core of Cloud A can communicate with only one core of Cloud B in a channel with bandwidth of 1 Gbps, representing secure two-party computation as defined in Definition 1. The performance of the three SQC protocols is measured by processing one million SNVs. The most expensive secure P-Z protocol takes less than one hour

sorting, which is the dominant overhead, can run in parallel. For our system configuration, we test the protocols on a cluster of 16 nodes, each equipped with Intel Xeon CPU E5-2680 v3 processors clocked at 2.5 GHz. Each machine consists of 8 cores and 32 GB of RAM. Half of the machines simulate cloud A, and the other half simulate cloud B. The bandwidth between machines is 1 Gbps.

In Figure 7-b, we show the performance of running the three SQC protocols on one million SNVs. In total, executing the three quality control tasks takes about one hour, which we deem to be an acceptable cost, considering the days or even months of data access-authorization procedures for biomedical studies. SQC safeguards the studies and minimizes the privacy concerns, producing a minimal interface where researchers retrieve sanitized and useful results in practical running time through parallel computation.

4 Discussion

Even though the data are not revealed as long as cloud A and cloud B do not collude, X_1 and X_2 should still be protected, considering that potential data breaches can occur on both clouds. The protection can be enforced on three levels, including hard disks, memory and cache. For example, on the level of the hard disk, HIPAA compliant cloud services could be one solution that normally encrypts the storage on disk. In this way, even if attackers steal the data on a disk from both clouds, they are not able to recover the original information. On the level of memory, the use of trusted hardware (e.g. Intel Software Guard eXtensions (SGX)) (Chen *et al.*, 2017) has become an increasingly popular and powerful approach in recent years. This approach encrypts the memory so that attackers cannot steal the information, even by compromising the memory. There are more sophisticated attacks that occur, however, on the CPU cache level (Liu *et al.* 2015b; Zhang *et al.*, 2014). There are different countermeasures proposed to thwart these attacks, but few of them are effective enough to be widely deployed in practice. The aforementioned cloud-protection methods are orthogonal to this work and could be added as components to our system.

For the effectiveness of hiding variants, we should emphasize that it is difficult to argue about the privacy guarantee of Procedure 1 alone, either theoretically or experimentally. Indeed, if the adversary has public reference statistics, it might be able to roughly map the identities by sorting the public statistics and comparing them with the sorted study statistics, but the precision of the mapping is highly data-dependent.

In most cases of genomic computation (e.g. GWAS, risk test), accuracy is of paramount importance. Therefore, many privacy-preserving techniques that introduce noise to data are criticized because of their unacceptable negative impact on computation accuracy (Erlich and Narayanan, 2014; Fredrikson *et al.*, 2014), even though a lot of advances have been made to apply these techniques in order to enable genomic data sharing (Fienberg *et al.*, 2011; Johnson and Shmatikov, 2013; Simmons *et al.*, 2016; Tramèr *et al.*, 2015; Yu *et al.*, 2014). However, quality control is meant to detect systematic errors of a study, hence analysts are concerned mainly about the global characteristics of the data, rather than the precise value of individual data points. For example, analysts might expect to see the data points cluster along the diagonal line of a pane rather than a horizontal line. Such a relatively loose requirement provides us room to change the data points of the results in a controllable manner such that the output patterns can still lead to the same conclusion on the data quality.

Note that if the statistics under consideration describe pairwise SNV correlation (linkage disequilibrium), reducing precision might not be a sufficient measure to defend against some categories of attacks (Wang *et al.*, 2009). Indeed, if the victim is in the case group, such pairwise statistics from a few SNVs might contribute a substantial amount of information to attackers because it is less common to find a combination of two alleles than to find either one of them. Although these pairwise correlations are not seen in our data, researchers should take more precautions about revealing such correlation statistics than statistics of independent SNVs.

In parallel to SQC based on secure multiparty computation and differential privacy, homomorphic encryption is also widely used for untrusted cloud computing. Such an encryption enables an untrusted cloud to perform computation on randomized ciphertext, which is projected into certain computation on the corresponding plaintext. This is a highly desirable solution for our purpose of utilizing the cloud computing power without revealing the sensitive plaintext data, although existing efficient schemes are still constrained by the limited number of possible operations (Fan and Vercauteren, 2012). In the privacy-preserving genomic and medical studies, researchers have also proposed various systems based on homomorphic encryption (Wang *et al.*, 2016; Shimizu *et al.*, 2016; Kim and Lauter, 2015).

Overall, SQC addresses the pressing issues of privacy-preserving aggregate data sharing. By using a set of sanitization processes and advanced cryptographic tools, SQC guarantees that users of the framework have access to only minimal but sufficient output information that is unlikely to be useful for inference attacks. In particular, SQC automates the quality control phase in GWAS meta-analysis in a privacy-preserving manner. By projecting the quality-control protocols in a secure computation framework, SQC offers an effective balance between the needs of researchers for GWAS meta-analysis and the needs of data owners to respect the genetic privacy of research participants. Moreover, running SQC does not incur any utility loss for subsequent meta analyses. Although the strong security and privacy guarantees of SQC comes at the cost of a high computation overhead, we demonstrate that it is practical to perform the task using a recently proposed parallel secure computation framework. Nonetheless, the computational drawback of using secure computation is still obvious; for example, the one-hour secure P-Z plotting time on 128 cores for a single study is not attractive when there are many studies, e.g. more than 100. This can be ameliorated by using advanced secure hardware (e.g. Intel SGX) (Chen *et al.*, 2017). It is imperative to continuously reevaluate the privacy and utility of aggregate data sharing as novel privacy-enhancing technologies are developed and security threats arise. Integrative solutions, such as SQC, that carefully consider the use and misuse of aggregate data are essential for ensuring its secure and privacy-conscious sharing and maximizing its utility in genomic research.

Acknowledgements

We thank Jean Louis Raisaro and Juan Ramon Troncoso-Pastoriza for thoughtful comments and discussion. We also thank Ruth Loos, Thomas Winkler and Iris Heid for their approval of this work. Cohort-specific summary statistics were kindly provided by the GIANT consortium.

Conflict of Interest: none declared.

References

Batcher,K.E. (1968) Sorting Networks and Their Applications. In: *Proceedings of the April 30–May 2, 1968, Spring Joint Computer Conference, AFIPS '68 (Spring)*, pp. 307–314, ACM, New York, NY, USA.

Chen,F. *et al.* (2017) PRINCESS: Privacy-protecting Rare disease International Network Collaboration via Encryption through Software guard extensionS. *Bioinformatics*.

Chen,Y. *et al.* (2012) Large-scale privacy-preserving mapping of human genomic sequences on hybrid clouds. In: *19th Annual Network and Distributed System Security Symposium (NDSS)*.

Dwork,C. and Roth,A. (2014) The algorithmic foundations of differential privacy. *Found Trends Theor. Comput. Sci.*, **9**, 211–407.

Erlich,Y. and Narayanan,A. (2014) Routes for breaching and protecting genetic privacy. *Nat. Rev. Genet.*, **15**, 409–421.

Fan,J. and Vercauteren,F. 2012. *Somewhat Practical Fully Homomorphic Encryption. Cryptology ePrint Archive* Report 2012/144.

Fienberg,S.E. *et al.* (2011) Privacy preserving GWAS data sharing. In: *2011 IEEE 11th International Conference on Data Mining Workshops (ICDMW)*, pp. 628–635.

Fredrikson,M. *et al.* (2014) Privacy in pharmacogenetics: an end-to-end case study of personalized warfarin dosing. In: *Proceedings of the 23rd USENIX Security Symposium*.

Homer,N. *et al.* (2008) Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genet.*, **4**, e1000167.

Im,H.K. *et al.* (2012) On sharing quantitative trait GWAS results in an era of multiple-omics data and the limits of genomic privacy. *Am. J. Hum. Genet.*, **90**, 591–598.

Jacobs,K.B. *et al.* (2009) A new statistic and its power to infer membership in a genome-wide association study using genotype frequencies. *Nat. Genet.*, **41**, 1253–1257.

Johnson,A. and Shmatikov,V. (2013) Privacy-preserving data exploration in genome-wide association studies. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13*, pp. 1079–1087, ACM, New York, NY, USA.

Kim,M. and Lauter,K. (2015) Private genome analysis through homomorphic encryption. *BMC Med. Inform. Decis. Mak.*, **15**, S3.

Knuth,D.E. (1998) The art of computer programming, Vol. **3**, 2nd edn. In: *Sorting and Searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.

Liu,C. *et al.* (2015a) ObliVM: a programming framework for secure computation. In: *2015 IEEE Symposium on Security and Privacy (SP)*, pp. 359–376.

Liu,F. *et al.* (2015b) Last-level cache side-channel attacks are practical. In: *2015 IEEE Symposium on Security and Privacy*, pp. 605–622

Lumley,T. and Rice,K. (2010) Potential for revealing individual-level information in genome-wide association studies. *JAMA*, **303**, 659–660.

Nayak,K. *et al.* (2015) GraphSC: parallel secure computation made easy. In: *2015 IEEE Symposium on Security and Privacy (SP)*, pp. 377–394.

Sankararaman,S. *et al.* (2009) Genomic privacy and limits of individual detection in a pool. *Nat. Genet.*, **41**, 965–967.

Shimizu,K. *et al.* (2016) Efficient privacy-preserving string search and an application in genomics. *Bioinformatics*, **32**, 1652–1661.

Simmons,S. and Berger,B. (2015) One size doesn't fit all: measuring individual privacy in aggregate genomic data. In: *2015 IEEE Security and Privacy Workshops (SPW)*, pp. 41–49.

Simmons,S. *et al.* (2016) Enabling privacy-preserving GWASs in heterogeneous human populations. *Cell Syst.*, **3**, 54–61.

Singh,A.P. *et al.* (2013) MetaSeq: privacy preserving meta-analysis of sequencing-based association studies. *Pac. Symp. Biocomput.*, 356–367.

Tramèr,F. *et al.* (2015) Differential privacy with bounded priors: reconciling utility and privacy in genome-wide association studies. In: *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15*, pp. 1286–1297, ACM, New York, NY, USA.

Visscher,P.M. and Hill,W.G. (2009) The limits of individual identification from sample allele frequencies: theory and statistical analysis. *PLOS Genet.*, **5**, e1000628.

Wang,R. *et al.* (2009) Learning your identity and disease from research papers: information leaks in genome wide association study. In: *Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS '09*, pp. 534–544, ACM, New York, NY, USA.

Wang,S. *et al.* (2016) HEALER: homomorphic computation of ExAct Logistic rEgRession for secure rare disease variants analysis in GWAS. *Bioinf. Oxf. Engl.*, **32**, 211–218.

Winkler,T.W. *et al.* (2014) Quality control and conduct of genome-wide association meta-analyses. *Nat. Protoc.*, **9**, 1192–1212.

Wood,A.R. *et al.* (2014) Defining the role of common variation in the genomic and biological architecture of adult human height. *Nat Genet.*, **46**, 1173–1186.

Xie,W. *et al.* (2014) SecureMA: protecting participant privacy in genetic association meta-analysis. *Bioinformatics*, **30**, 3334–3341.

Yu,F. *et al.* (2014) Scalable privacy-preserving data sharing methodology for genome-wide association studies. *J. Biomed. Inform.*, **50**, 133–141.

Zerhouni,E.A. and Nabel,E.G. (2008) Protecting aggregate genomic data. *Science*, **322**, 44–44.

Zhang,Y. *et al.* (2014) Cross-Tenant Side-Channel Attacks in PaaS Clouds. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS '14*, pp. 990–1003, ACM, New York, NY, USA.