Poster: Solving Private Systems of Linear Equations with Garbled Circuits

Phillipp Schoppmann and Adrià Gascón
University of Edinburgh
s1565903@sms.ed.ac.uk agascon@inf.ed.ac.uk

Borja Balle
Lancaster University
b.deballepigem@lancaster.ac.uk

I. Motivation

The advent of the big data revolution has made it possible to analyse ever more complex datasets for scientific and social insights. At the same time, a widespread concern for the use given to these datasets has spurred an interest in privacy-preserving data analysis. The goal of this line of research is to build frameworks allowing individuals to contribute their personal data for aggregate analysis without compromising their privacy. Secure multi-party computation (MPC) provides fundamental tools for the design of such frameworks, specially since MPC implementations published in recent years have showcased the practical feasibility of theoretical constructions pioneered by Yao [1]. However, generic approaches to MPC rarely scale to problem sizes commonly found in real-world applications of privacy-preserving data analysis involving massive datasets. The search for practical alternatives has lead to the design of hybrid approaches that carefully combine several MPC techniques. Identifying building blocks common to these solutions is an important endeavor if we hope to make progress in the design of robust and scalable tools for privacy-preserving data analysis. In this work we initiate a systematic study of methods for solving private systems of linear equations in a setting commonly found in private linear regression [2] and collaborative filtering [3] protocols. Methods for solving private systems of linear equations have been considered before, but no thorough comparison between different approaches can be found in the literature. Our goal is to provide such comparison and derive guidelines for practitioners that can lead to more efficient implementations tailored for privacy-preserving data analysis.

II. Problem, Methods, and Related Work

A. Systems of Linear Equations

We focus on systems of linear equations $Ax = b$ with $A \in \mathbb{R}^{d \times d}$ a symmetric positive definite matrix. Under these assumptions A is invertible and the system admits a unique solution. These systems arise naturally in many contexts, including data analysis problems involving least-squares and ridge regression. The three algorithms for solving $Ax = b$ that we consider in this paper are specially designed for positive definite systems. Even when $Ax = b$ admits a unique solution, numerically finding such solution exactly can be a challenging problem due to finite precision effects. This is a well-understood phenomenon in numerical linear algebra, and has led to the development of numerous numerical algorithms since the time of Gauss. An important observation is that these problems generally arise when the condition number $\kappa(A)$ given by the ratio between the largest and the smallest eigenvalues of A is large. We will see in Section III how this affects the accuracy in our experiments. Another pertinent observation is that when the system arises from a data analysis problem, a solution whose precision is on the same order of magnitude of the variance of the input data is typically good enough. This motivates the use of iterative methods producing approximate solutions after a small number of iterations.

B. Privacy Setup and Threat Model

In our setting, two parties P_1 and P_2 hold secret additive shares (A_i, b_i) of the matrix A and vector b in a linear system $Ax = b$; that is, $A = A_1 + A_2$ and $b = b_1 + b_2$. The goal is to have the parties compute the solution x of $Ax = b$, in a way that (i) does not rely on a trusted third party, and (ii) the parties do not learn anything about each others input beyond what is revealed by the solution x. We assume a semi-honest threat model, where, intuitively, the parties are motivated to learn about each other inputs, but will not deviate from the protocol. This exact problem arises in [2] as an intermediate step in a ridge regression protocol combining partially homomorphic encryption and garbled circuits. We rely on Yao's garbled circuits protocol [1] to solve the 2-party computation problem above. Yao's protocol allows two parties to evaluate a function f (represented as a circuit) in a secure way in the sense of our constraits (i) and (ii) above, and in the presence of a semi-honest adversary (see [4] for precise definitions of security in 2-party computation and the semi-honest threat model).

C. Solution Methods

Three different methods for solving positive definite linear systems are compared: Cholesky, LDLT, and CGD. The first two rely on well-known decompositions of matrices which simplify the solution of a system of linear equations by allowing efficient Gaussian elimination without pivoting. The main difference is that Cholesky computes a factorization of the form $A = LL^\top$ with L a lower-triangular matrix, while LDLT yields $A = LDL^\top$ with lower-triangular L and diagonal D. It is easy to convert one factorization to the other, but Cholesky requires the computation of square roots,

while LDLT can be done without square roots but involves a few more multiplications. We consider both methods because computing of square root in a garbled is done using an iterative procedure, which can be costly and introduce extra numerical errors. The last method is conjugate gradient descent (CGD), a well known iterative method frequently employed for solving large-scale positive definite systems in sparse settings. CGD iteratively minimizes the residual $\|Ax - b\|$ by providing a sequence of points guaranteed to converge to the solution of the system. Iterative methods are appealing for data analysis problems because they provide a natural way to trade-off computation time and accuracy by running a fixed number of iterations. This choice of methods is motivated by several reasons, namely: (a) they are standard methods and their computational complexity has been well-studied; (b) they are relatively robust to numerical errors arising from finite-precision effects which makes them suitable for GC implementations; (c) they admit data-agnostic implementations because they do not involve pivoting strategies. Solution of positive definite linear systems with a garbled circuits implementation of Cholesky's method was presented in [2]. To the best of our knowledge, LDLT and CGD have not been studied before in this context. Other authors have considered expensive privacy-preserving protocols for matrix factorization as an intermediate step for solving systems of linear equations; we do not consider these approaches because their computational complexity is higher than the methods we propose even in non-private scenarios.

D. Implementation Details

Our implementation is based on Obliv-C [5], a framework for automatic generation of MPC protocols. It uses Yao's garbled circuits, which get compiled from a high-level language based on C. In order to deal with real numbers, we implement fixed-point arithmetic on top of Obliv-C basic types, including the square root computation of [2]. In contrast to floating point arithmetic, fixed point computation is in general prone to numerical error propagation, specially in iterative algorithms. To cope with that, we adapted a scaling method originally developed for computing the Lanczos kernel in fixed-point settings [6] to CGD.

III. Preliminary Results

A. Experimental Setup

We test implementations of the three methods presented above with randomly generated linear systems with varying dimensionality and condition number. For a fixed dimension d, we randomly generate positive definite systems of linear equations by taking $A = XX^\top/m$ with $X \in \mathbb{R}^{d\times m}$ a random matrix with i.i.d. standard Gaussian entries, and $b = Az$ with $z \in \mathbb{R}^d$ a random vector with i.i.d. entries uniform in $[-1, 1]$. By taking different choices for $m \geq d$ we get positive definite systems (A, b) with varying condition number. In our experiments we use dimensions in the range $3 \leq d \leq 100$ and generate systems with $m = Cd$ for $C \in \{5, 10, 50, 10000\}$. In each of these settings we generate 5 different systems and report the mean where appropriate.

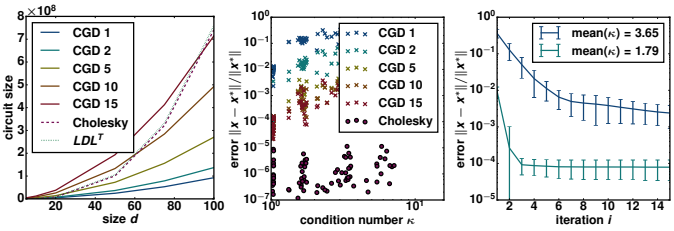

Fig. 1. Experimental results

B. Results

Our results are summarized in Fig. 1. The leftmost plot displays how the circuit size for different methods (and different number of iterations in CGD) scale with the dimension d. Circuit size is directly correlated with execution time. Thus, we see that if the number of iterations is independent of d then for large dimensions CGD is always more efficient than Cholesky and LDLT (which behave almost the same). The effect the number of iterations of CGD has on the accuracy (measured in terms of relative error) is displayed in the middle plot. As expected, since in general CGD needs d iterations for exactly solving a linear system, limiting the number of iterations makes the method less accurate than Cholesky/LDLT. Interestingly, we see that this difference diminishes as the condition number approaches one, which is the typical setting for least squares regression with bounded features and ridge regularization. The rightmost panel shows the progress in accuracy of CGD as the number of iterations increases, averaged over multiple systems and multiple d for two different settings of m. We see how (in average) when the condition number is small the error in CGD stabilizes after very few iterations.

C. Discussion and Future Work

While iterative methods do not reach the accuracy of exact approaches, we show how computation times can be reduced considerably, at calculable costs in terms of accuracy. Furthermore, the effect of the condition number on both accuracy and convergence speed indicates the value of our approach for low condition numbers often encountered in large real-world problems. In future work we will make our implementation freely available, test it with larger systems, and use it to build robust privacy-preserving data analysis tools.

References

[1] A. C. Yao, "How to generate and exchange secrets," in *Symposium on Foundations of Computer Science (FOCS)*, 1986.

[2] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *IEEE Symposium on Security and Privacy (SP)*, 2013.

[3] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2013.

[4] Y. Lindell and B. Pinkas, "A proof of security of yao's protocol for two-party computation," *J. Cryptology*, 2009.

[5] S. Zahur and D. Evans, "Obliv-c: A language for extensible data-oblivious computation," Cryptology ePrint Archive, Report 2015/1153, 2015.

[6] J. L. Jerez, G. A. Constantinides, and E. C. Kerrigan, "A low complexity scaling method for the lanczos kernel in fixed-point arithmetic," *IEEE Transactions on Computers*, 2015.