Efficient Secure Linear Algebra in the Presence of Covert or Computationally Unbounded Adversaries

Payman Mohassel* and Enav Weinreb**

Abstract. In this work we study the design of secure protocols for linear algebra problems. All current solutions to the problem are either inefficient in terms of communication complexity or assume that the adversary is honest but curious. We design protocols for two different adversarial settings: First, we achieve security in the presence of a covert adversary, a notion recently introduced by [Aumann and Lindell, TCC 2007]. Roughly speaking, this guarantees that if the adversary deviates from the protocol in a way that allows him to cheat, then he will be caught with good probability. Second, we achieve security against arbitrary malicious behaviour in the presence of a computationally unbounded adversary that controls less than a third of the parties. Our main result is a new upper bound of $O(n^{2+1/t})$ communication for testing singularity of a shared $n \times n$ matrix in constant round, for any constant t in both of these adversarial environments. We use this construction to design secure protocols for computing the rank of a shared matrix and solving a shared linear system of equations with similar efficiency.

We use different techniques from computer algebra, together with recent ideas from [Cramer, Kiltz, and Padró, CRYPTO 2007], to reduce the problem of securely deciding singularity to the problem of securely computing matrix product. We then design new and efficient protocols for secure matrix product in both adversarial settings. In the two-party setting, we combine cut-and-choose techniques on random additive decomposition of the input, with a careful use of the random strings of a homomorphic encryption scheme to achieve simulation-based security. Thus, our protocol avoids general zero-knowledge proofs and only makes a black-box use of a homomorphic encryption scheme.

1 Introduction

Solving a set of linear equations is one of the most basic algorithmic tasks, with numerous applications to various fields. In a distributed system, linear constraints may reflect sensitive information and thus parties who wish to solve a joint set of equations are interested in revealing as little information as possible on their input. Research in secure multiparty

* Department of Computer Science, UC Davis. pmohassel@ucdavis.edu.
** CWI, Amsterdam, The Netherlands. e.n.weinreb@cwi.nl.

computation (MPC) has been impressively successful in allowing secure computation of every function with efficiency that is proportional to its circuit complexity [BGW88,CCD88,Yao86,GMW87]. However, for various linear algebraic tasks, these general constructions fall well short of giving optimal protocols in terms of communication and round complexity. Starting with the work of Cramer and Damgård [CD01], the task of designing secure protocols for linear algebraic problems has been the focus of several recent works in secure computation [NW06,KMWF07,CKP07].

We focus on the problem of deciding the singularity of a shared matrix $M \in F^{n \times n}$, where $\mathbb{F}$ is a finite field. Many linear algebraic tasks, e.g. solving a joint set of linear equations and computing the rank of a shared matrix, are efficiently reducible to this task. When no honest majority is assumed, as in the classic two-party setting, our protocols are secure in the presence of a *covert* adversary [AL07], assuming the existence of public key homomorphic encryption schemes. Previous communication efficient secure protocols for linear algebra were known only in the honest but curious setting. In case there is a guarantee that the adversary controls less than one third (one half) of the parties, our protocols are secure against malicious (honest but curious) behaviour relying on no computational assumptions. Our protocols are constant round and achieve communication complexity of $O(n^{2+1/t})$ for every constant t. This is the first constant round construction for secure linear algebra with nearly linear communication complexity in the input size.

Applying General Results Fails. Unfortunately, general results in MPC do not yield efficient protocols for linear algebra. This is due to the fact that the communication complexity linearly depends on the Boolean circuit complexity of the function to be computed (or on the number of multiplication gates in case of information theoretic security). Alas, the circuit complexity of matrix singularity, as well as that of many other linear algebraic problem, is tightly related to that of matrix product [BCS97]. The best known upper bound for circuits for matrix product is $O(n^{\omega})$ [CW87] with $\omega \cong 2.38$, which is significantly larger than the input size. Moreover, in the information theoretic setting the round complexity of the protocols is related to the multiplicative depth of the corresponding circuit, preventing these general construction from yielding constant round protocols. This leads to the following approach: Design efficient protocols for matrix product and then use them to achieve protocols for other linear algebraic tasks.

Matrix Product in Realistic Adversary Models. Therefore, our first step is to design constant round secure protocols for matrix product in the covert setting and in the information theoretic malicious setting with communication complexity $O(n^2)$. In the honest but curious setting, simple solutions for this problems are achieved using homomorphic encryption in the computational setting and linear secret sharing schemes in the information theoretic setting. However, when arbitrary adversarial behavior is considered, this task becomes more involved. Since very few efficient secure protocols for interesting problems are known in the malicious setting when no honest majority is assumed, Aumann and Lindell [AL07] have defined an interesting compromise: protocols that are secure in the presence of a covert adversary. Roughly speaking, the security guarantee is that if the adversary deviates from the protocol in a way that allows him to "cheat", then the honest parties are guaranteed to detect this cheating with good probability (e.g. 1/2). The ability to tolerate cheating with some fixed probability turns out useful for secure computation of matrix product and, consequently, for other secure linear algebraic tasks.

Techniques for Matrix Product. The general idea behind our secure protocol for matrix product in the covert setting, is to compute a decomposition of the input matrices into additive shares and then use homomorphic encryption to perform the computation on these shares. We use the random strings of the encryption scheme to prove that the additive shares indeed sum up to the input matrix, avoiding expensive zero knowledge proofs. Towards this goal, we need the homomorphic encryption scheme to have a property that all the known schemes enjoy: when computing $C = E(m_1, r_1) + E(m_2, r_2)$, one can compute a string r such that $C = E(m_1 + m_2, r)$. We note that although not every homomorphic encryption scheme has this property, some well known encryption schemes, such as the one by Paillier [Pal99], are suitable for our purposes. After the computations take place, the parties reveal parts of their additive sharing of the input, catching cheating adversaries[1] with probability 1/2. Revealing parts of the decomposition of the input enables easy input extraction, which makes the simulation go through.

Constant Round Reduction From Singularity to Matrix Product. Unfortunately, in spite of the tight connection of matrix product and linear algebraic problems such as matrix singularity, efficient protocols for the former does not immediately imply efficient protocols for the latter. The reason is that the known reductions do not translate to protocols with constant round complexity. Therefore, we need to design a special purpose protocol for matrix singularity, equipped with our secure protocol

[1] the cheating probability can be reduced to $1/k$ paying a factor of k in the communication complexity.

for matrix product. We use ideas from [Wie86], [KP91], and [KS91] to reduce the problem of deciding the singularity of a general matrix M into deciding the singularity of a related *Toeplitz* matrix T. We then use a lemma by Leverrier [JáJ92] which reduces the problem into computing the traces of powers of T. Finally, we define the Toeplitz matrix of polynomials $(I - \lambda T)$ and use the Gohberg-Semencul formula for the inverse of a Toeplitz matrix, to compute the above traces efficiently. We rely on techniques for iterated matrix product [CKP07] (which, in turn, is based on techniques from [BIB89]), combined with some simple linear algebraic manipulations, to translate the above algorithmic ideas into a constant round secure protocol for matrix singularity with the above mentioned communication complexity.

1.1 Related Work

Information theoretic setting. Cramer and Damgård initiated the study of secure protocols for solving various linear algebra problems [CD01]. Their work was done in the information theoretic multi-party setting, with the main focus on achieving constant round complexity. The communication complexity[2] of their protocols is $\Omega(n^4)$ while the size of the inputs is just $O(n^2)$. Cramer et al. [CKP07] designed a constant round protocol for solving m equations in n variables with communication complexity $O(m^4 + n^2 m)$ which improves on [CD01] for small values of m. The only protocol applicable to the information theoretical setting with communication complexity of roughly $O(n^2)$ is that of Nissim and Weinreb [NW06]. However, this protocol has polynomial round complexity ($\Omega(n^{0.27})$) and is proved secure only in the honest but curious model. (The protocols of [CD01] and [CKP07] are secure in the malicious model, assuming the adversary controls less that a third of the parties.)

Computational Setting. As previously noted, using the general well-known garbled circuit method of Yao [Yao82], one can get a constant round protocol for various linear algebraic problems with communication complexity that is proportional to the Boolean circuit complexity of matrix multiplication, for which the best upper bound known is $O(n^\omega)$ [CW87] for $\omega \cong 2.38$. As discussed above, the protocol of [NW06] was the first to improve the communication complexity to roughly $O(n^2)$, in the price of large round complexity. Later, Kiltz et al. [KMWF07] improved on the

[2] The complexity of their protocols can be reduced to $O(n^3)$ using the matrix product protocol from this paper.

round complexity to get a protocol with $O(\log n)$ rounds and communication complexity of roughly $O(n^2)$. However, this protocol is secure only in the honest but curious setting.

Organization In section 2 we introduce the necessary definitions and primitives. In sections 3 and 4, we design secure and efficient protocols for matrix product in the covert setting and information theoretic setting respectively. In section 5, we reduce the problem of securely testing singularity of a matrix to a secure matrix product, put everything together, give our main theorem and explain how to extend our results to other linear algebra problems.

2 Preliminaries

Notation. Our protocols work with elements over a finite field, which we denote by $\mathbb{F}$. We guarantee the security with probability $1 - O(n^2)/|\mathbb{F}|$, where $|\mathbb{F}|$ is size of the field and n is the size of matrices we deal with in our linear algebra problems. Letting $|\mathbb{F}|$ be superpolynomial in n, we can achieve protocols with negligible error probability.[3] We count the communication complexity of our protocols in terms of number of field elements communicated. By default, a vector $v \in \mathbb{F}^n$ is regarded as a column vector. By $\mathbb{F}[x]$ we refer to the ring of polynomials over $\mathbb{F}$, and by $\mathbb{F}[[x]]$ to the field of rational functions over $\mathbb{F}$. Given a collection of matrices $C_1, \ldots, C_p$ over $\mathbb{F}$ where all the C_i's have the same number of rows, we denote by $|C_1|C_2| \ldots |C_p|$ the block matrix resulted by concatenating the $C_1, \ldots, C_p$.

2.1 Definitions For Security Against Covert Adversaries

Aumann and Lindell, [AL07], give a formal definition of security against covert adversaries in the *ideal/real simulation paradigm*. This notion of adversary lies somewhere between those of semi-honest and malicious adversaries. Loosely speaking, the definition provides the following guarantee: Let $0 \leq \epsilon \leq 1$ be a value (called the deterrence factor). Then any attempts to cheat by an adversary is detected by the honest parties with probability at least ϵ. Thus provided that ϵ is sufficiently large, an adversary that wishes not to get caught cheating will refrain from attempting to cheat, lest it be caught doing so. Furthermore, in the strongest version of security against covert adversaries introduced in [AL07], the adversary

[3] For smaller fields, one can pay a polylogarithmic factor in the communication complexity and work in an extension field.

will not learn any information about the honest parties' inputs if he gets caught. Please see [AL07] for the detailed definitions.

Homomorphic Encryption We use a semantically-secure public-key encryption scheme that allows for simple computations on encrypted data. In particular, we use encryption schemes where given two encryptions $E(m_1)$ and $E(m_2)$, we can efficiently compute a *random* encryption of m_1+m_2. We denote this by $E(m_1+m_2) = E(m_1)+_h E(m_2)$. Note that this implies that given an encryption $E(m)$ and $c \in \mathbb{F}$, we can efficiently compute a random encryption $E(cm)$; we denote this by $E(cm) = c \times_h E(m)$. For a matrix M We denote by $E(M)$ an entry-wise encryption of the matrix. Note that the above implies that two encrypted matrices can be added, and that we can compute a multiplication of an encrypted matrix by a matrix in the clear.

As an example for a homomorphic encryption scheme, we can use Pallier's [Pal99] cryptosystem. One minor issue is that the domain of Pallier's cryptosystem is the ring Z_n, where n is the product of two large and secret primes. Though Z_n is in fact not a field, any operation which separates it from a field leads to a factoring of n, and thus is unlikely to occur during a computation.

2.2 Definitions for the Information Theoretic Model

Linear Secret Sharing Schemes. To design our secure multiparty protocols, we use a linear secret sharing scheme (LSSS) to share values over a finite field $\mathbb{F}$. Each party receives a share that contains one or more field elements from the dealer. Each share is computed as a fixed linear function of the secret and some random field elements chosen by the dealer. The *size* of an LSSS is the total number of field elements distributed by the dealer. We denote by $[a]$ a secret sharing of $a \in \mathbb{F}$. For a vector $v \in \mathbb{F}^n$, we denote by $[v]$ a coordinate-wise sharing of the vector. Similarly, for a matrix $M \in \mathbb{F}^{n\times n}$ we denote by $[M]$ an entry-wise sharing of the matrix, and for a polynomial $p(x) \in \mathbb{F}[x]$ we denote by $[p]$ a coefficient-wise secret sharing of the polynomial.

Due to the linearity of the secret sharing, given secret shares of $[a]$ and $[b]$ and a third field element $c \in \mathbb{F}$, parties can compute secret shares of $[a+b]$ and $[ca]$ non-interactively. Furthermore, we require the LSSS to be *multiplicative*. Roughly speaking, this means that party P_i can use his shares of $[a]$ and $[b]$ to non-interactively compute a value c_i. The product ab can then be computed from c_i's using a fixed reconstruction vector

$(r_1, \ldots, r_k)$, where k is the number of parties[4]. In [CDM00], it is shown how to construct a multiplicative LSSS scheme from any LSSS scheme without sacrificing efficiency.

3 Matrix Product Secure Against Covert Adversaries

Given shares of two $n \times n$ matrices, we design a protocol for securely computing shares of the product matrix in the presence of a covert adversary in a small constant number of rounds and communication complexity $O(n^2)$. It is possible to compile a $O(n^2)$ communication matrix product protocol that is secure against semi-honest adversaries into a protocol that is secure against malicious adversaries using generic (or specially designed) zero knowledge proofs. However, we do not know how to do so without adding a significant overhead to the communication complexity of the original protocol.

The main idea is to break the input matrices into multiple additive shares and use a homomorphic encryption along with cut-and-choose techniques to perform the required computation on the shares. We further use the random strings of the encryption scheme to prove that the additive shares in fact add up to the original inputs. For this purpose, we need the homomorphic encryption to have an extra property: If one knows the random strings that correspond to the encryptions $E(m_1)$, and $E(m_2)$, one can efficiently compute the random string that corresponds to the encryption $E(m_1 + m_2) = E(m_1) +_h E(m_2)$.

We proceed to discuss the way in which a shared matrix is held in our protocols. In a sharing of a matrix $M \in \mathbb{F}^{n \times n}$, Alice holds $(A, E_b(B, r_b), r_a)$ and Bob holds $(B, E_a(A, r_a), r_b)$, where A and B are random matrices subject to the condition that $A + B = M$, the strings r_a and r_b are uniformly random strings, and $E_a(\cdot,\cdot)$ ($E_b(\cdot,\cdot)$) denotes encryption under Alice's (Bob's) public key. The remainder of this section is organized as follows: We start by formally defining the matrix product functionality in terms of the above secret sharing representation of the matrices. Then, we present our protocol for efficiently implementing this functionality.

Definition 1 (Matrix Multiplication Functionality). *The matrix multiplication functionality (MatMul) is defined as follows:*
Input to Alice: $A_1, A_2, r_{a_1}, r_{a_2}, E_b(B_1, r_{b_1}), E_b(B_2, r_{b_2}), E_b(C, r_c), r_{a'}$

[4] For ease of composition, we assume that each party holds a single field element as his share but note that our techniques automatically generalize to the case where the share sizes are larger.

Input to Bob: $B_1, B_2, C, r_{b_1}, r_{b_2}, r_c, E_a(A_1, r_{a_1}), E_a(A_2, r_{a_2})$
Output of Alice: $(A_1 + B_1)(A_2 + B_2) + C$
Output of Bob: $E_a((A_1 + B_1)(A_2 + B_2) + C, r_{a'})$

A few remarks on this definition are in place. The inputs to the players contain two shared matrices $(A_1 + B_1)$ and $(A_2 + B_2)$, together with the matrix C (which is also Bob's share of the output) and the string $r_{a'}$, which is the random string according to which Alice encrypts her output. That is, we choose not to introduce randomness into the definition and leave the choices of C and $r_{a'}$ to the protocols that use matrix product as a sub-protocol. This design choice was made to simplify the presentation of the protocol. But, it is not hard to construct on top of our protocol for this functionality, a protocol for a functionality with random C and $r_{a'}$ as outputs. Hence we assume that we have access to a secure coin-tossing protocol such as the one given in [Lin01].

To simplify the composition, we divide the protocol into several parts, where the parts will be performed sequentially one after the other.

Alice's Computation

1. **Alice writes her inputs as sums.** For each $i \in \{1, 2\}$, Alice chooses two random matrices $A_{i,1}$ and $A_{i,2}$ such that $A_{i,1} + A_{i,2} = A_i$, and generates two random strings $r_{i,1}$ and $r_{i,2}$. For each $j \in \{1, 2\}$, Alice sends $D_{i,j} \stackrel{\text{def}}{=} E_a(A_{i,j}, r_{i,j})$ to Bob.
2. **Alice proves her sums.** For each $i \in \{1, 2\}$, Alice computes a string $r_{0,i}$, such that
$$E_a(0, r_{0,i}) = E_a(A_i, r_{a_i}) -_h D_{i,1} -_h D_{i,2}.$$
Alice sends $r_{0,1}$ and $r_{0,2}$ to Bob.
3. **Alice sends output parts that depend only on her input.** For every $i_1, i_2 \in \{1, 2\}^2$, Alice generates a random element s_{i_1,i_2} and sends Bob $H_{i_1,i_2} \stackrel{\text{def}}{=} A_{1,i_1} \times_h D_{2,i_2} +_h E(0, s_{i_1,i_2})$.
4. **Bob's challenge.** Bob chooses a random number $c \in \{1, 2\}$ and sends it to Alice.
5. **Alice proves knowledge of encrypted data.** For every $i \in \{1, 2\}$ Alice sends Bob $A_{i,c}$, $r_{i,c}$. Moreover, Alice sends Bob $s_{c,1}$ and $s_{c,2}$ chosen at step 3.
6. **Bob verifies Alice's data**
   (a) **Alice's encryptions indeed sum to** $E_a(A_i, ra_i)$ For each $i \in \{1, 2\}$, Bob verifies that indeed $E(0, r_0) = E(A_i, r_{a_c}) -_h D_{i,1} -_h D_{i,2}$.
   (b) **Alice knows her encrypted data** For each $i \in \{1, 2\}$, Bob verifies that $D_{i,c} = E_a(A_{i,c}, r_{i,c})$.
   (c) **The computations that depend only on Alice were performed correctly.** Bob verifies that indeed $H_{c,j} = A_{1,c} \times_h D_{2,j} +_h E(0, s_{c,j})$ for $j \in \{1, 2\}$.

Bob's Computation

1. **Bob writes his inputs as sums.** For each $i \in \{1,2\}$, Bob randomly chooses two random matrices $B_{i,1}$ and $B_{i,2}$ such that $B_{i,1}+B_{i,2}=B_i$, and generates two random strings $q_{i,1}$ and $q_{i,2}$. For each $j \in \{1,2\}$, Bob sends $F_{i,j} \stackrel{\text{def}}{=} E_b(B_{i,j}, q_{i,j})$ to Alice. Similarly, Bob Chooses 12 random matrices C_{i_1,i_2}, C'_{i_1,i_2} and C''_{i_1,i_2} for every $i_1, i_2 \in \{1,2\}^2$, such that $\sum_{i_1,i_2} C_{i_1,i_2} + \sum_{i_1,i_2} C'_{i_1,i_2} = C$, and 12 random strings t_{i_1,i_2}, t'_{i_1,i_2}, and t''_{i_1,i_2}, and sends Alice $G_{i_1,i_2} = E_b(C_{i_1,i_2}, t_{i_1,i_2})$, $G'_{i_1,i_2} = E_b(C'_{i_1,i_2}, t'_{i_1,i_2})$, and $G''_{i_1,i_2} = E_b(C''_{i_1,i_2}, t''_{i_1,i_2})$, for every $i_1, i_2 \in \{1,2\}^2$.
2. **Bob proves his sums.** For each $i \in \{1,2\}$, Bob computes a string q_0, such that
$$E_b(0, q_0) = E(B_i, r_{b_i}) -_h F_{i,1} -_h F_{i,2}.$$
Bob sends q_0 to Alice. Similarly, Bob computes a string t_0 such that
$$E_b(0, t_0) = E(C, r_c) -_h \sum_{i_1,i_2} G_{i_1,i_2} -_h \sum_{i_1,i_2} G'_{i_1,i_2} -_h \sum_{i_1,i_2} G''_{i_1,i_2}.$$
Bob sends t_0 to Alice.
3. **Bob sends information that depends only on his input** For every $i_1, i_2 \in \{1,2\}^2$, Bob sends Alice $L_{i_1,i_2} = B_{1,i_1} B_{2,i_2} + C''_{i_1,i_2}$.
4. **Bob performs computations on Alice's inputs** For every $i_1, i_2 \in \{1,2\}$, computes $K_{i_1,i_2} = D_{1,i_1} B_{2,i_2} + C_{i_1,i_2}$ and $K'_{i_1,i_2} = D_{2,i_1} B_{1,i_2} + C'_{i_1,i_2}$. Bob sends K_{i_1,i_2} and K'_{i_1,i_2} to Alice for every $i_1, i_2 \in \{1,2\}^2$.
5. **Alice's challenge.** Alice chooses a two random numbers $d_1, d_2 \in \{1,2\}^2$ and sends them to Bob.
6. **Bob proves knowledge of encrypted data.** For every $i \in \{1,2\}$ Bob sends Alice B_{i,d_i} and q_{i,d_i}. Moreover for every $j \in \{1,2\}$ Bob sends Alice the matrices C_{j,d_j}, C'_{j,d_j}, and C''_{j,d_j} and the strings t_{j,d_j}, t'_{j,d_j}, and t''_{j,d_j}.
7. **Alice verifies Bob's data**
   (a) **Bob's encryptions indeed sum to $E_a(B_i, rb_i)$.** For every $i \in \{1,2\}$, Alice verifies that indeed $E_b(0, q_0) = E_b(B_i, r_{b_{d_i}}) - F_{i,1} - F_{i,2}$.
   (b) **Bob's encryptions indeed sum to $E_a(C, r_c)$.** Alice verifies that indeed $E_b(0, t_0) = E_b(C, r_c) - \sum_{i_1,i_2} G_{i_1,i_2} - \sum_{i_1,i_2} G'_{i_1,i_2}$.
   (c) **Bob knows his encrypted data.** For each $i \in \{1,2\}$, Alice verifies that $F_{i,d_i} = E_b(B_{i,d_i}, q_{i,d_i})$. Moreover, for every $j \in \{1,2\}$ Alice verifies that $G_{j,d_j} = E_b(C_{j,d_j}, t_{j,d_j})$ and that $G'_{j,d_j} = E_b(C'_{j,d_j}, t'_{j,d_j})$.
   (d) **The computations that depend only on Bob were computed correctly** Alice verifies that $L_{d_1,d_2} = B_{1,d_1} B_{2,d_2} + C''_{d_1,d_2}$
   (e) **Bob's homomorphic computations were correct** For every $j \in \{1,2\}$, Alice verifies that indeed $K_{j,d} = E_a(A_{1,j}) B_{2,d} + C_{j,d}$ and $K'_{j,d} = E_a(A_{2,j}) B_{1,d} + C'_{j,d}$.

Output Computation

1. **Output computation.** Alice decrypts all the values, sums everything up to get $(A_1+B_1)(A_2+B_2)+C$ and computes $O_A = E_a((A_1+B_1)(A_2+B_2)+C, r_{a'})$. Bob on his side, computes $O_B = E_a((A_1+B_1)(A_2+B_2)+C, r_{a''})$ using homomorphic operations on the shares that he holds. Bob performs these computation in a deterministic way, such that Alice knows $r_{a''}$.
2. **Output Delivery** Alice chooses two random matrix X_1 and X_2 such that $X_1 + X_2 = (A_1 + B_1)(A_2 + B_2) + C$. Alice chooses a random string x_1, and set the string x_2 to satisfy $E_a(X_1, x_1) +_h E_a(X_2, x_2) = O_A$. Alice sends $E_a(X_1, x_1)$ and $E_a(X_2, x_2)$ to Bob, together with a random string x_0 satisfying $O_B -_h O_A = E_a(0, x_0)$.
3. **Output Challenge** Bob chooses a random number $y \in \{0, 1\}$ and sends it to Alice.
4. **Output Response** Alice sends Bob X_y and x_y.
5. **Bob's output** Bob verifies the encryption $E_a(X_y, x_y)$ and that $O_B -_h E_a(X_1, x_1) -_h E_a(X_2, x_2) = E_a(0, x_0)$. If this is the case, he outputs $E_a(X_1, x_1) +_h E_a(X_2, x_2)$.
6. **Alice's output** Alice outputs $(A_1 + B_1)(A_2 + B_2) + C$.

The following theorem summarizes the results of this section.

Theorem 1. *Let n be a positive integer and consider matrices of dimension $n \times n$. Given a secure coin-tossing functionality, the above protocol securely realizes the matrix product functionality, in presence of a covert adversary with deterrence probability $\epsilon = 1/4$. The protocol is constant round and requires $O(n^2)$ communication.*

Due to lack of space, details of the simulators and proof of security appear in the full version. The above protocol in its current form is not secure under parallel composition due to the rewinding nature of the simulator in the proof. However, we can securely run many instances of the matrix product in parallel if (i) the same public key and encryption scheme is used for all instances and (2) the same challenges are used in all instances of the matrix product protocol. In fact, the description of the simulator in the proof for security of the parallel matrix product protocol will be almost identical to the simulator for the original protocol.

4 Secure Matrix Product in the Information Theoretic Model

Using standard techniques for multiplying two shared values leads to a simple protocol for matrix product in the information theoretic setting that requires $O(n^3)$ communication.

However, one can think of our element-wise secret sharing of the matrices as a secret sharing scheme over the matrix ring. In light of this observation, we can efficiently generalize the committed multiplication protocol of [CDM00] to a constant round committed matrix multiplication protocol that is secure against active adversaries, with $O(n^2)$ communication. A more detailed description of the protocols is deferred to the full version. The following theorem summarizes the result:

Theorem 2. *Given two shared matrices* $[A]$ *and* $[B]$ *where* $A, B \in \mathbb{F}^{n\times n}$*, there exists a multiparty protocol, secure against a malicious adversary that corrupts less than a third of the parties, for computing a secret sharing of the product* $[C] = [AB]$ *in a constant number of rounds and with* $O(n^2)$ *communication.*

As we will see in future sections, secure computation of other linear algebra problems is reduced to a secure matrix product protocol. The guaranteed security is in part due to the existing general composition theorems in the information-theoretic setting [KLR06].[5]

5 From Matrix Singularity to Matrix Product

In this section we design a secure protocol for deciding singularity of a shared matrix given an efficient implementation of a secure protocol for matrix product. Our techniques apply to both two-party protocols secure against covert adversaries and multiparty protocols secure against computationally unbounded malicious adversaries.

In Section 5.1, we design a constant round protocol for computing shares of the *linearly recurrent sequence* $v, Mv, \ldots, M^{2n-1}v$, given shares of a square matrix M and a vector v. Later, in Section 5.2, we apply this protocol to M and a random shared vector v, to reduce the problem of deciding the singularity of M into the problem of deciding the singularity of a related *Toeplitz* matrix T. In Section 5.3 we design a protocol for deciding singularity of a Toeplitz matrix, by applying the protocol for the linearly recurrent sequence on a different set of inputs. Finally, in Section 5.4, we connect all the above to get our main result: a secure constant round protocol for deciding shared matrix singularity with communication complexity $O(n^{2+1/t})$ for every positive integer t.

[5] The general composition theorem we use from [KLR06], requires the composed protocols to perform an initial synchronization step. This synchronization step can easily be added to all of our protocols without any asymptotic overhead.

5.1 Secure Computation of the Sequence $\{M^i v\}$

Given secret shares of a matrix $M \in \mathbb{F}^{n\times n}$ and a vector $v \in \mathbb{F}^n$, our goal is to design a secure protocol for computing shares of the vector sequence $\{M^i v\}_{1\le i\le 2n}$. This construction is an important building block for the design of our matrix singularity protocol. Using the methods of [CKP07] for computing powers of a matrix, one can securely compute the sequence $\{M^i v\}_{1\le i\le 2n}$ in constant round and with $O(n^4)$ communication. In this section, we design a constant round protocol for the same task with communication complexity of $O(n^{2+1/t})$ for any arbitrary constant t.

In [CKP07], the problem of computing sharings of $I, M, M^2, \ldots, M^d$ is reduced into (i) generating a sharing of $O(d)$ random matrices and their inverses, and (ii) executing $O(d)$ parallel[6] matrix multiplications. Using standard techniques that are available in both settings, both steps can be computed by performing $O(d)$ matrix multiplications of $n \times n$ matrices. Thus, using a constant round secure protocol for matrix product we get a secure protocol for computing these with $O(dn^2)$ communication which we refer to as $\mathrm{POWERS}_d(M)$. The following lemma summarizes this improvement.

Lemma 1 ($\mathrm{POWERS}_d(M)$)**.** *Given shares of a matrix $M \in \mathbb{F}^{n\times n}$, there exist a protocol for securely computing shares of $\{I, M, M^2, \ldots, M^d\}$ in constant round and with $O(dn^2)$ communication.*

We are now ready to describe our protocol for computing shares of the sequence $\{M^i v\}_{1\le i\le 2n}$. We introduce some notation. Denote $\ell \stackrel{\text{def}}{=} \left\lceil (2n)^{1/s} \right\rceil$, and for $0 \le i \le s-1$, denote by POW_M^i, the following set of $\ell+1$ powers of M: $\mathrm{POW}_M^i \stackrel{\text{def}}{=} \{I, M^{\ell^i}, M^{2\ell^i}, \ldots, M^{\ell^{i+1}}\}$. The following observation is easy to verify.

Observation 3 *For every $1 \le t \le 2n$, the matrix M^t can be represented as a product of s matrices $M^t = \prod_{0\le j\le s-1} M_{t,j}$, where $M_{t,j} \in POW_M^j$ for every $0 \le j \le s-1$.*

[6] Both in the covert setting and in the information theoretical setting, a secure protocol for computing d matrix products in parallel in constant round and communication complexity $O(dn^2)$ is a straightforward generalization of our matrix product protocols.

Protocol LIN-SEQ$_n(M, v)$

Input: Shares of a matrix $[M] \in \mathbb{F}^{n\times n}$, and vector $[v] \in \mathbb{F}^n$
Output: Shares of $2n$ vectors $[Mv], [M^2 v], \ldots, [M^{2n} v]$

1. Parties agree on a positive integer s, and let $\ell = \lceil (2n)^{1/s} \rceil$
2. Parties securely compute shares of the sets $POW_M^1, POW_M^2, \ldots, POW_M^s$ by sequentially running $\text{POWERS}_\ell(M), \text{POWERS}_\ell(M^\ell), \ldots, \text{POWERS}_\ell(M^{\ell^{s-1}})$ (See Lemma 1.)
3. Let $B_0 = v$
4. For $i = 0$ to $s - 1$:
   (a) For $j = 1, \ldots, \ell - 1$, parties compute $[C_j] = [M^{j\ell^i}][B_i]$ by performing the secure matrix product protocol of section 2.
   (b) As a result, parties hold secret shares of the $n \times \ell^{i+1}$ matrix $B_{i+1} = |C_{\ell-1}|C_{\ell-2}| \cdots |C_1|B_i|$.
5. Parties hold shares of the $n \times \ell^s$ matrix B_s which contains the sequence $M^i v$ as its columns, for $1 \le i \le l^s$.

Lemma 2. *Given a shared matrix $[M] \in \mathbb{F}^{n\times n}$ and a shared vector $[v] \in \mathbb{F}^n$, for any positive integer s, there exist a multiparty protocol for securely computing shares of the sequence $\{M^i v\}_{0\le i\le 2n}$ in $O(s)$ rounds and with $O(sn^{2+1/s})$ communication.*

Proof. In view of Claim 3, it is easy to verify that after s iterations of the for loop, the matrix B_s contains the $2n$ vectors in the sequence $\{M^i v\}_{0\le i\le 2n}$ as its columns. Based on Lemma 1, step 2 can be performed in $O(s)$ rounds and with $O(sn^{1/s}n^2)$ communication. In each iteration, step 4a requires ℓ multiplication of an $(n\times n)$ by an $(n \times \ell^i)$ matrix. Using an efficient matrix product protocol this requires only $O(n^2)$ communication, and leads to a total of $O(s\ell n^2) = O(sn^{1/s}n^2)$ communication for the s iterations of the loop.

5.2 From General Matrix Singularity to Toeplitz Matrix Singularity

In this section we reduce the problem of securely deciding the singularity of a general shared matrix M into the problem of securely deciding the singularity of a related shared matrix T.

Theorem 4. *For every integer s, there is an $O(s)$ rounds protocol that securely transforms a sharing of a matrix $M \in \mathbb{F}^{n\times n}$ into a sharing of a Toeplitz matrix $T \in \mathbb{F}^{n\times n}$, such that, with probability $1 - O(n^2)/|\mathbb{F}|$, matrix M is singular iff the matrix T is singular. The communication complexity of the protocol is $O(sn^{2+1/s})$.*

The proof of Theorem 4 relies on the following Lemmata. Due to lack of space, the full proof is defered to the final version of the paper.

Lemma 3 ([KS91]). *Consider the matrix $M \in \mathbb{F}^{n\times n}$ of rank r (unknown), and random non-singular matrices $V, W \in \mathbb{F}^{n\times n}$, let $M' = VMW$. Then, with probability greater than $1-n(n+1)/|\mathbb{F}|$, the upper-left $i \times i$ submatrices of M' are invertible for $1 \le i \le r$.*

Lemma 4 ([KS91]). *Let $B \in \mathbb{F}^{n\times n}$ be a matrix with invertible upper left $i \times i$ submatrices for $1 \le i \le r$, where $r < n$ is the rank of B. Let D be a randomly chosen diagonal matrix in $\mathbb{F}^{n\times n}$. Then, $r = deg(m_{DB}) - 1$ with probability greater than $1 - n^2/|\mathbb{F}|$.*

Lemma 5 ([Wie86]). *Let $A \in \mathbb{F}^{n\times n}$ and let m_A be the minimal polynomial of A. For $\mathbf{u}, \mathbf{v} \in \mathbb{F}^n$ chosen uniformly at random, consider the linearly recurrent sequence $\{a_i\} = \{u^t A^i v\}$. We have that the minimal polynomial of the sequence $\{a_i\}$ is equal to m_A with probability at least $1 - 2\deg(m_A)/|\mathbb{F}|$.*

For a linearly recurrent sequence $\{a_i\}_{i=0}^{\infty}$, and $\alpha > 0$, denote by T_α the following Toeplitz matrix:

$$T_\alpha = \begin{pmatrix} a_{\alpha-1} & a_{\alpha-2} & \cdots & a_1 & a_0 \\ a_\alpha & a_{\alpha-1} & \cdots & a_2 & a_1 \\ \vdots & a_\alpha & \ddots & \vdots & a_2 \\ & \vdots & & & \vdots \\ a_{2\alpha-3} & & & a_{\alpha-1} & \\ a_{2\alpha-2} & a_{2\alpha-3} & \cdots & a_\alpha & a_{\alpha-1} \end{pmatrix}$$

Lemma 6 ([KP91]). *Let $\{a_i\}$ be a linearly recurrent sequence, and let d be the degree of its minimum polynomial. For any $\alpha \ge 0$, let T_α be the Toeplitz matrix constructed as above. Then, $Det(T_d) \neq 0$ and for all $\alpha > d$, $Det(T_\alpha) = 0$.*

5.3 Deciding the Singularity of a Toeplitz Matrix

We test the singularity of a Toeplitz matrix by first computing its characteristic polynomial and then determining whether its constant coefficient is equal to zero or not. The well known lemma of Leverrier, connects the characteristic polynomial of a matrix to solution of a linear system that depends on the trace of powers of the matrix (see Appendix A).

Particularly, if we compute the traces of matrices $T^1, \ldots, T^n$, computing the characteristic polynomial reduces to inverting an invertible matrix for which there exist simple and efficient protocols in both settings. Hence, our main challenge remains to compute traces of powers of T in a round and communication efficient manner.

Denote by $X \in \mathbb{F}[\lambda]^{n\times n}$ the matrix $X = I + \lambda T + \lambda^2 T^2 + \ldots + \lambda^n T^n$. Note that entries of X are polynomials of degree at most n in $\mathbb{F}[\lambda]$. Therefore, the trace of the matrix X is also a polynomial of degree at most n in λ. It is easy to see that coefficients of this trace polynomial are in fact the traces of powers of T. Hence, our goal is to compute the trace of X. However, as the naïve representation of the matrix X consists of n^3 field elements, we need to compute the trace of X in a more clever way.

Consider the matrix $(I - \lambda T) \in \mathbb{F}[\lambda]^{n\times n}$. Since T is a Toeplitz matrix, the matrix $(I - \lambda T)$ is Toeplitz as well. One may compute the power series extension

$$(I - \lambda T)^{-1} = I + \lambda T + \lambda^2 T^2 + \ldots \in \mathbb{F}[[\lambda]]^{n\times n}$$

Note that $X \equiv (I - \lambda T)^{-1} (\mathrm{mod}\ \lambda^{n+1})$. Hence, the matrix X is equivalent modulo λ^{n+1} to the inverse of a Toeplitz matrix from $\mathbb{F}[[\lambda]]^{n\times n}$.

The following Lemma, based on the Gohberg-Semencul formula (e.g. see [GS72,FMKL79,BGY80]), shows that X can be represented using only its first and last columns:

Lemma 7. *For any Toeplitz matrix $T \in \mathbb{F}^{n\times n}$, it holds that*

$$\begin{aligned}
X &\stackrel{def}{=} (I - \lambda T)^{-1} \ mod\ \lambda^{n+1} = I + \lambda T + \lambda^2 T^2 + \ldots + \lambda^n T^n \\
&= \frac{1}{u_1}\Bigg(\begin{pmatrix} u_1 & & & & \\ u_2 & u_1 & & & \\ u_3 & u_2 & u_1 & & \\ \vdots & & \ddots & \ddots & \\ u_n & u_{n-1} & \ldots & u_2 & u_1 \end{pmatrix} \begin{pmatrix} v_1 & v_2 & & \cdots & v_n \\ & u_1 & v_2 & & v_{n-1} \\ & & \ddots & \ddots & \vdots \\ & & & v_1 & v_2 \\ & & & & v_1 \end{pmatrix} \\
&- \begin{pmatrix} 0 & & & & \\ v_n & 0 & & & \\ v_{n-1} & v_n & 0 & & \\ \vdots & & \ddots & \ddots & \\ v_2 & v_3 & \ldots & v_n & 0 \end{pmatrix} \begin{pmatrix} 0 & u_n & u_{n-1} & \ldots & u_2 \\ & 0 & u_n & & u_3 \\ & & \ddots & \ddots & \vdots \\ & & & 0 & v_n \\ & & & & 0 \end{pmatrix} \Bigg) \ mod\ \lambda^n
\end{aligned} \tag{1}$$

where $u = (u_1, \ldots, u_n)^t$ and $v = (v_1, \ldots, v_n)^t$ are the first and last columns of $(I - \lambda T)^{-1}$ respectively.

This brief representation of X allows for an efficient computation of the trace of X. Particularly, we get the following equation from (1):

$$\text{Trace}(X) \equiv \text{Trace}((I-\lambda T)^{-1}) \equiv \frac{1}{u_1}(nu_1v_1+(n-2)u_2v_2+\ldots+(-n+2)u_nv_n) \quad (2)$$

To compute Trace(X), it is sufficient to perform the following two steps **(i)** Compute the first and last columns of X: We apply Lemma 2 to compute the two sequences $\{T^i e_1\}_{1\leq i\leq n}$ and $\{T^i e_n\}1 \leq i \leq n$ where e_1 and e_n are column vectors of $\langle 1, 0, \ldots, 0\rangle$ and $\langle 0, \ldots, 0, 1\rangle$, respectively. This automatically gives us the first and last columns of X, and requires $O(s)$ rounds and $O(sn^{2+1/s})$ communication. **(ii)** Compute the trace of X using these two columns: This can be done based on Equation 2. Since every polynomial among $u_1 \ldots, u_n, v_1, \ldots, v_n$ is of degree at most n, multiplying two of these polynomials can be done in constant round and communication $O(n)$ in both settings (see [MF06]). Finally, dividing by u_1, which is invertible modulo λ^{n+1} (since its constant coefficient is 1), can be done in constant round and with communication complexity $O(n)$ (see [MF06]). Hence, we conclude that computation of traces of $T, T^2, \ldots, T^n$ can be done in $O(s)$ rounds and communication complexity $O(sn^{2+1/s})$.

5.4 The Protocol for Matrix Singularity

The following protocol for testing singularity of a matrix combines all the techniques discussed in this section:
Input: Shares of a matrix $[M] \in \mathbb{F}^{n\times n}$
Output: Shares of a bit $[b]$, where $b = 1$ if M is singular and 0 otherwise.

1. Compute shares of the Toeplitz matrix $[T]$, as described in Section 5.2.
2. Execute LIN-SEQ$_n(T, (1, 0, \ldots, 0)^t)$ and LIN-SEQ$_n(T, (0, 0, \ldots, 0, 1)^t)$. The output of these two executions yield shares of the first and last columns ($[u]$ and $[v]$ respectively) of the matrix X.
3. Given shares of $[u]$ and $[v]$, compute shares of $TRACE(X)$, as described in Section 5.3.
4. Construct the matrix S from equation 3 without any interaction and solve the linear system of equation 3 using the protocol for inverting an invertible matrix (see e.g. [BIB89])[7]. As a result, they hold shares of coefficients of characteristic polynomial of T.
5. Securely test whether the constant coefficient of the characteristic polynomial of T is 0 using an equality testing protocol (e.g. [DFKNT06,Yao86]).

[7] Entries of S are traces of T^is which are computed in step 3.

Theorem 5. *Let n and $s < n$ be a positive integers and consider matrices of dimension $n \times n$. There exist secure protocols that realizes the Matrix Singularity functionality in the covert adversarial model, and the information-theoretic model in $O(s)$ rounds and with $O(sn^{2+1/s})$ communication.*

Other Linear algebra problems Using existing techniques, one can efficiently and in constant round, reduce the task of secure computation of rank of a matrix and solving a linear system of equation to testing singularity of a matrix. It is easy to verify that the reductions will apply to both settings we considered in this paper. See [KMWF07] or the full version of this paper for more detail.

Acknowledgment We would like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[AL07] Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. In *TCC*, pages 137–156, 2007.

[BCS97] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*. Springer-Verlag, Berlin, 1997.

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for noncryptographic fault-tolerant distributed computations. In *STOC*, pages 1–10, 1988.

[BGY80] RP. Brent, FG Gustavson, and DYY Yun. Fast solution of Toeplitz systems of equations and computation of pade approximants. In *J. Algorithms*, pages 259–295, 1980.

[BIB89] J. Bar-Ilan and D. Beaver. Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. In *PODC*, pages 201–209, New York, NY, USA, 1989. ACM Press.

[CCD88] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *STOC*, pages 11–19, 1988.

[CD01] R. Cramer and I. Damgaard. Secure distributed linear algebra in a constant number of rounds. In *CRYPTO*, pages 119–136. Springer-Verlag, 2001.

[CDM00] R. Cramer, I. Damgård, and U. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT*, pages 316–334. Springer-Verlag, 2000.

[CKP07] R. Cramer, E. Kiltz, and C. Padró. A note on secure computation of the Moore-Penrose pseudo-inverse and its application to secure linear algebra. In *CRYPTO*, volume 4622, page 613, 2007.

[CW87] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. In *STOC*, pages 1–6. ACM Press, 1987.

[FMKL79] B. Friedlander, M. Morf, T. Kailath, and L. Ljung. New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices. In *Linear Algebra and Appl*, volume 27, pages 31–60, 1979.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC*, pages 218–229, 1987.

[GS72] I. Gohberg and A. Semencul. On the inversion of finite Toeplitz matrices and their continuous analogs. In *Math Issl*, pages 201–233, 1972.

[JáJ92] J. JáJá. *An introduction to parallel algorithms.* Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1992.

[KLR06] E. Kushilevitz, Y. Lindell, and T. Rabin. Information-theoretically secure protocols and security under composition. *STOC*, pages 109–118, 2006.

[KMWF07] E. Kiltz, P. Mohassel, E. Weinreb, and M. Franklin. Secure linear algebra using linearly recurrent sequences. In *TCC* pages 291–310. Springer, 2007.

[KP91] E. Kaltofen and V. Pan. Processor efficient parallel solution of linear systems over an abstract field. In *SPAA*, pages 180–191. ACM Press, 1991.

[KS91] E. Kaltofen and D. Saunders. On Wiedemann's method of solving sparse linear systems. In *AAECC-9*, pages 29–38, London, UK, 1991.

[DFKNT06] I. Damgaard and Matthias Fitzi and Eike Kiltz and Jesper Buus Nielsen and Tomas Toft. Unconditionally Secure Constant-Rounds Multi-Party Computation for Equality, Comparison, Bits and Exponentiation. Proceedings of the third Theory of Cryptography Conference, TCC, 2006, pages = 285–304.

[Lin01] Y. Lindell. Parallel Coin-Tossing and Constant-Round Secure Two-Party Computation. *Crypto*, pages 171–189, 2001.

[MF06] P. Mohassel and M. Franklin. Efficient Polynomial Operations in the Shared-Coefficient Setting, Proc. of Public Key Cryptography coference, PKC, pages 44–57, 2006.

[NW06] K. Nissim and E. Weinreb. Communication efficient secure linear algebra. In *TCC*, pages 522–541, 2006.

[Pal99] P. Pallier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.

[Wie86] D. H. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theor.*, 32(1):54–62, 1986.

[Yao82] A. C. Yao. Protocols for secure computations. In *FOCS*, pages 160–164, 1982.

[Yao86] A. C. Yao. How to generate and exchange secrets. In *FOCS*, pages 162–167, 1986.

A Leverrier's Lemma

Lemma 8 (Leverrier's Lemma). *The coefficients $c_1, \ldots, c_n$ of the characteristic polynomial of an $n \times n$ matrix T satisfies the following system of equations:*

$$S.\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_n \end{pmatrix} where, S = \begin{pmatrix} 1 & 0 & 0 & ... & 0 \\ s_1 & 2 & 0 & ... & 0 \\ s_2 & s_3 & 3 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{n-1} & s_{n-2} & s_{n-3} & ... & n \end{pmatrix} \tag{3}$$

and $s_i = tr(T^i)$ for $1 \leq i \leq n$