# Multiparty Computation with Minimal Overhead without Circuit Transformation

Aditya Hegde[*†] Phuoc Van Long Pham[*‡] Mingyuan Wang[*§]

**Abstract**

A central goal in secure multiparty computation (MPC) is to minimize the gap between the total computation performed by all parties and the cost of insecure circuit evaluation. In the honest majority setting, the ideal overhead of $\tilde{O}(1)$ is known to be achievable. However, all prior protocols over prime fields attaining this goal for general circuits relied on the so-called *share packing* techniques and, as a consequence, required prohibitively expensive circuit transformation steps to impose repetitive structure on the circuit.

The only exception is the recent work of Garg, Jain, Mukherjee, and Wang (CRYPTO'24), which introduced a *share unpacking* technique and obtained the first such protocol without circuit transformation. However, their protocol only applies when the underlying field size is extraordinarily large, in particular, exponential in the *square* of the number of parties.

In this work, we present the first MPC protocol with $\tilde{O}(1)$ overhead for any prime field $\mathbb{F}_p$ (more generally, any integer ring $\mathbb{Z}_p$) whose size is only required to be $n^{\Omega(n)}$, where $n$ is the number of parties (as opposed to $n^{\Omega(n^2)}$). Technically, we extend the unpacking approach of Garg et al. by developing a new approach to *linear randomness extraction over integers*, a tool that may be of independent interest.

[*]Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing.

[†]Johns Hopkins University, ahegde@cs.jhu.edu

[‡]Brown University, phuoc_pham_van_long@brown.edu

[§]New York University, Shanghai mingyuan.wang@nyu.edu

# Contents


**1 Introduction** **3**
- 1.1 Our Contributions . . . 3

**2 Technical Overview** **5**

**3 Preliminaries** **14**
- 3.1 Secure Multiparty Computation . . . 15
- 3.2 Chinese Remainder Theorem-based Secret Sharing . . . 16
- 3.3 Useful Facts . . . 16

**4 Linear Mask Extractors** **18**
- 4.1 Definition . . . 18
- 4.2 Template for Constructing Linear Mask Extractors . . . 19

**5 Toeplitz Linear Mask Extractor** **21**
- 5.1 Proof . . . 21
- 5.2 Efficiency of Extraction . . . 33
- 5.3 Concrete Parameters . . . 34

**6 Honest-Majority MPC** **36**
- 6.1 Proof . . . 38

**References** **48**

# 1 Introduction

Secure multiparty computation (MPC) [Yao86, GMW87, BGW88, CCD88] is a fundamental cryptographic primitive that enables mutually distrustful parties to jointly evaluate a function on their private inputs while revealing nothing beyond the prescribed output. Since its inception, MPC has grown into a rich area of research with wide-ranging applications in both theory and practice.

A central objective in MPC is to minimize the overhead of secure computation compared to insecure circuit evaluation. In the *honest-majority setting*, protocols achieving the ideal $\tilde{O}(1)$ overhead[1] are known [DIK10]. Such protocols are referred to as *scalable MPC* as their efficiency does not degrade as the number of parties grows. Beyond their theoretical importance, scalable MPC protocols are especially appealing for multiple reasons. First, honest-majority MPC relies on the assumption that an adversary cannot corrupt a majority of the parties. As the number of parties grows, this assumption becomes increasingly reliable, thereby strengthening security without sacrificing efficiency. Second, in certain applications such as federated machine learning, the number of participants can naturally be large; when the total cost is independent of the number of parties, the *workload per participant* decreases as the system scales. Third, in applications such as zero-knowledge proofs from MPC-in-the-Head [IKOS07], it is often desirable for the number of parties to be on the order of the security parameter to avoid parallel repetition for soundness amplification. Scalable MPC thus provides a natural first step towards constructing proof systems with $\tilde{O}(1)$ prover overhead.

On the technical side, scalable MPC is well understood for structured circuits. In particular, [FY92, BGJK21] show how to achieve strict constant overhead for a broad class of *highly repetitive circuits*, e.g., SIMD circuits. For general circuits (over a prime field) that may not possess such repetitive structures, however, all known approaches [DIK10, GJMW24][2] achieve $\tilde{O}(1)$ overhead only after a heavy circuit transformation step.[3] Such preprocessing is undesirable because (i) it typically increases both round and communication complexity by polylogarithmic factors in the circuit size, and (ii) it is often concretely very expensive. Fundamentally, these circuit transformations are required because all prior approaches achieving scalable MPC rely on *share packing* [FY92], a technique that is only applicable when the circuit has certain repetitive structures.

The sole exception for general circuits without circuit transformation is the recent work of Garg, Jain, Mukherjee, and Wang [GJMW24], which departs from the share-packing paradigm and introduces a novel *share unpacking* technique. Their protocol is the first to achieve $\tilde{O}(1)$ overhead for arbitrary circuits without circuit transformation. However, their result applies only when the underlying ring $\mathbb{Z}_p$[4] is extraordinarily large, requiring $\log p = \Theta(n^2 \log n)$, where $n$ denotes the number of parties. This strong restriction stems from a technical barrier in their approach, despite the fact that the underlying ideas could potentially extend to moduli as small as $\log p = \Theta(n \log n)$. Moreover, their technique further requires that $p$ be coprime with $2, 3, \ldots, n-1$, which rules out many natural settings, such as the widely studied arithmetic circuits over $\mathbb{Z}_{2^n}$ [CDE+18, CKL21, CRX21, EXY22, BBMHS22, LED+24].

**This work.** Building on the share-unpacking framework of Garg et al. [GJMW24], we present the first MPC protocol achieving $\tilde{O}(1)$ overhead over arbitrary integer rings $\mathbb{Z}_p$ whose size is exponential in the number of parties. Our result matches the natural lower bound inherent in the technical approach of [GJMW24].

Our main technical contribution is a new method for performing *linear randomness extraction over the integers*, which allows us to eliminate both the modulus-size and coprimality restrictions on $p$ required in [GJMW24]. Concretely, we show that *a random Toeplitz matrix is a good linear mask extractor over the integers* (refer to Definition 5) with high probability. We believe this technique is of independent interest beyond the MPC setting.

## 1.1 Our Contributions

**MPC with Minimal Overhead in the Honest-Majority Setting.** In this work, we obtain the first scalable multiparty computation (MPC) protocols without circuit transformation over arbitrary integer rings $\mathbb{Z}_p$, where the mod-

[1] Throughout this work, $\tilde{O}(\cdot)$ suppresses polylogarithmic factors.

[2] [GJMW24] sketched an approach for achieving scalable MPC over an exponentially large prime field. Namely, one can translate an arithmetic circuit into a Boolean circuit that has the *repetitive structure* required for scalable MPC.

[3] Over extension fields, the work of [CDdH07] presents a scalable MPC over $\mathbb{F}_{p^m}$ with linear extension degree (i.e., $m = \Omega(n)$), where each party performs computation over $\mathbb{F}_p$.

[4] We write $\mathbb{Z}_p$ instead of $\mathbb{F}_p$ to highlight that $p$ is not necessarily a prime.

ulus satisfies $\log p = \Theta(n \log n)$. In particular, we show that in the (super) honest-majority setting,[5] there exists an $n$-party protocol for evaluating arithmetic circuits over $\mathbb{Z}_p$ that achieves unconditional security against an *adaptive* semi-honest adversary. Formally, we obtain the following theorem.

**(Informal) Theorem 1.** *Let $\kappa$ denote the statistical security parameter. For all constant $\delta \in (0, 1/2)$ and positive integers $n, p \in \mathbb{N}$ such that $n \in \Omega(\kappa)$ and $\log p \in \Omega(n \log n)$, there exists an $n$-party MPC protocol for evaluating circuits over $\mathbb{Z}_p$ with unconditional security against an adaptive semi-honest adversary corrupting at most $(1/2 - \delta) \cdot n$ parties. The total communication is $O(|\mathsf{C}| \cdot \log p)$ bits and the total computation is $\widetilde{O}(|\mathsf{C}| \cdot \log p)$ bit operations.*

Before proceeding, we make the following remark on the necessity of achieving adaptive security for scalable MPC.

**Remark 1** (Necessity of Adaptive Security)**.** *Achieving MPC with minimal overhead against a* static *adversary is straightforward using a committee-based approach. Specifically, the parties can elect a small committee of size, say, $\log^2 n$, and delegate the entire computation to this committee. Since the committee size is only polylogarithmic in $n$, running MPC naïvely within the committee already achieves $\widetilde{O}(1)$ overhead. This approach, however, completely breaks down in the presence of* adaptive *adversaries.*

As mentioned above, one of the main differences between our results and prior work [GJMW24] lies in the required modulus size. Specifically, we require only $\log p = \Theta(n \log n)$, whereas their protocol requires $\log p = \Theta(n^2 \log n)$. We further note that reducing the modulus $p$ substantially below this regime is currently unlikely, for the following reason. In any protocol where all parties communicate for each gate evaluation, the total communication (and hence computation) necessarily grows as $|C| \cdot n$. If such a protocol achieves $\widetilde{O}(1)$ overhead, this implies that

$$|C| \cdot n = \widetilde{O}(|C| \cdot \log p),$$

since $\widetilde{O}(|C| \cdot \log p)$ is the bit complexity necessary for the *insecure* circuit evaluation. Consequently, the modulus $p$ must grow exponentially in $n$. As noted earlier, protocols that avoid per-gate interaction [FY92, DIK10, BGJK21] by evaluating $\Theta(n)$ gates in a batch are only known either for highly repetitive circuits or at the cost of circuit transformation. Therefore, given the current state-of-the-art, our results achieve near-optimality with respect to the required modulus size.

Beyond being an exciting theoretical improvement, we argue that this difference is also relevant in practice. For instance, [GJMW24] motivates MPC over large field setting by observing that it arises naturally in the collaborative generation of *cryptographic functionalities*. In particular, a recent active line of work in this direction is collaborative zkSNARKs [OB22, GGJ+23, CLMZ23, GGJ+25], where the underlying fields are inherently large, typically with $\log p = \Theta(\lambda)$, the security parameter. In such settings, our protocol supports nearly $\lambda$ parties, while their protocol is limited to about $\sqrt{\lambda}$ parties. For practical choices such as $\lambda = 128$, the value of $\sqrt{\lambda}$ is very small, making it unclear whether their protocol achieves any meaningful scalability advantage.

Moreover, as we discussed earlier, there are other natural scenarios, such as MPC in the head, where the number of parties itself is desired to be on the order of the security parameter. In these settings, our improvement on $m$ is especially relevant. We view the potential application of our protocol to proof generation with $\widetilde{O}(1)$ overhead for cryptographic functionalities via MPCitH as an exciting avenue for future works.

**Linear Mask Extraction over Integers.** As we mentioned earlier, the core technical ingredient underlying our results is a new *mask extraction*[6] result over the integers. At a high level, we obtain a seeded *linear* mask extractor for block-fixing sources (BFS). We summarize this result below.

**(Informal) Theorem 2.** *Let* BFS *be the family of $t$-block-fixing distributions over $\mathbb{Z}^n$. That is, if $\mathcal{D} \in$ BFS, then $t$-out-of-$n$ coordinates are fixed, but the rest are uniformly random over $\{0, \ldots, B_{\mathsf{rand}} - 1\}$. There exists a sampling procedure*

[5] It is known [DLN19] that a super honest majority (i.e., allowing a $(1/2 - \delta)$ fraction of corruptions, as opposed to a strictly honest majority) is necessary for achieving scalable MPC. We refer the reader to [GJMW24] for further discussion.

[6] We use the term *mask extraction*, rather than the more standard *randomness extraction*, since uniform randomness is not well defined over the integers. Instead, the extracted output is "sufficiently random" in the sense that it can be used as a one-time pad to statistically mask arbitrary integer secrets.

*for an integer matrix $E \in \mathbb{Z}^{\ell \times n}$, where $\ell = \Omega(n - t)$, such that for any secret $x \in \mathbb{Z}^{\ell}$, it holds that*

$$\mathrm{SD}\Big(\big(E,\ E \cdot \mathcal{D}\big),\ \big(E,\ E \cdot \mathcal{D} + x\big)\Big) \leqslant \exp(-\Omega(n)) + \frac{\log n \cdot n^{O(n)} \cdot \|x\|_\infty}{B_{\mathsf{rand}}}.$$

*Moreover, $E \cdot \mathcal{D}$ can be computed in* quasilinear time*, i.e., $O(n \cdot \mathrm{polylog}\, n)$.*

From the perspective of a standard *seeded randomness extractor*, the matrix $E$ serves as the seed, while the product $E \cdot \mathcal{D}$ can be viewed as the extracted randomness from the source $\mathcal{D}$. Our theorem shows that this extraction is sufficient to statistically mask any integer secret $x$, provided that

$$B_{\mathsf{rand}} \gg n^{O(n)} \cdot \|x\|_\infty.$$

Although [GJMW24] does not explicitly formulate its result in extractor terminology, a direct comparison, however, reveals that their extraction guarantee only holds when

$$B_{\mathsf{rand}} \gg n^{\Theta(n^2)} \cdot \|x\|_\infty,$$

and moreover requires the additional constraint that the secret $x$ be coprime with each of $2, 3, \ldots, n-1$.

Finally, our extractor is *adaptively secure*. Specifically, even if the seed $E$ is sampled first and an (unbounded) adversary subsequently chooses the source $\mathcal{D} \in \mathsf{BFS}$ adaptively as a function of $E$, the conclusion of our theorem continues to hold. Equivalently, one may interpret our result as a form of *seedless* randomness extraction for BFS sources: with overwhelming probability over the choice of $E$, the resulting mapping yields a seedless linear extractor. It remains an intriguing open question whether this construction can be derandomized.

## 2 Technical Overview

This section presents the main technical ideas underlying our results. We consider an arithmetic circuit $C$ defined over the ring $\mathbb{Z}_{p_0}$, where we emphasize that $p_0$ is not necessarily prime. Throughout, we use $n$ to denote the number of parties and $t$ to denote the number of (semi-honestly) corrupted parties. We focus on an information-theoretic MPC protocol in the honest-majority setting, namely, $t = (1/2 - \delta) \cdot n$ for some constant $\delta \in (0, 1/2)$.

**Secret Sharing and Scalable MPC.** All existing information-theoretic MPC protocols in the honest-majority setting are based on secret sharing. Parties first secret-share their inputs and then collectively emulate the circuit gate by gate: for each gate whose input wires are secret-shared, the parties invoke appropriate subprotocols (e.g., local evaluation followed by degree reduction) to obtain shares of the output wire.

When combined with *linear* secret sharing, however, this paradigm fundamentally conflicts with scalability. In a linear secret-sharing scheme, all shares reside in the same ring $\mathbb{Z}_{p_0}$ as the secret[7] and thus emulating a *single* arithmetic operation forces the parties collectively to perform $n$ arithmetic operations. As a result, the protocol inevitably incurs a multiplicative overhead of $n$. Notably, this overhead applies even to addition gates, where no communication is required but each party must still perform the local computation.

To circumvent this limitation, prior work developed the *share packing* technique [FY92]. In this approach, a single sharing instance encodes not one but $\Theta(n)$ secrets; consequently, a single local operation can simultaneously emulate $\Theta(n)$ gates, eliminating the per-gate factor-$n$ overhead in an *amortized* sense. The drawback, however, is that the protocol must evaluate entire batches of $\Theta(n)$ gates at once, which enforces strong repetitive structure on the circuit. As a result, scalable MPC without circuit transformation using share packing has only been achieved for *highly repetitive circuits*, such as SIMD circuits [FY92, BGJK21].

For arbitrary circuits that lack such structure and are therefore not packing-friendly, the only known approach is to apply *circuit transformation*, which restructures the original circuit into a packing-friendly form [DIK10, GJMW24]. Although this is theoretically possible, circuit transformation is generally undesirable in practice: it incurs significant concrete inefficiencies and, from a theoretical standpoint, introduces polylogarithmic overheads in communication, computation, and round complexity. For similar reasons, for example, while it is in principle possible to garble an

[7] In some constructions, shares may instead live in a ring extension of $\mathbb{Z}_{p_0}$ (e.g., [CRX21]).

arithmetic circuit by first converting it into a Boolean circuit and then applying Boolean garbling [Yao86], arithmetic garbling [AIK11] is still strongly preferred in practice.

**Unpacked Secret Sharing via CRT.** Recently, Garg et al. [GJMW24] introduced an alternative approach, termed *unpacked secret sharing*, to eliminate the overhead described above. To explain their idea, we first recall the secret-sharing scheme based on the Chinese Remainder Theorem (CRT) [Mig83, AB83, GRS99] and the MPC protocols built upon it.

CRT-based secret sharing is parameterized by public moduli $p_1, p_2, \ldots, p_n$, which are all pairwise coprime and also coprime with $p_0$. To share a secret $s \in \mathbb{Z}_{p_0}$, the dealer samples a random integer

$$S = s + u \cdot p_0,$$

where $u$ is chosen uniformly at random from $\{1, 2, \ldots, L\}$ for a suitably chosen parameter $L$. Party $i$'s share is then defined as $s_i = S \bmod p_i$.

Correctness requires that $L$ be small enough so that any set of shares above the reconstruction threshold uniquely determines $S$ (and hence $s = S \bmod p_0$) via the CRT. Privacy, on the other hand, requires that $L$ be large enough so that any set of shares below the privacy threshold is statistically close to uniform. Prior work [GRS99, GJM+23] shows that both requirements can be satisfied simultaneously. Informally, to achieve privacy against $t$ corruptions, the integer $S$ must be sampled from a range whose size is at least the product of any $t$ moduli.

We use the notation $[\![S_t]\!]$ to denote such a sharing of $s$, where the subscript $t$ serves as a reminder of the magnitude of the underlying integer. A formal description of the scheme appears in Section 3.2.

Furthermore, [GJM+23] observed that, much like Shamir's secret sharing, CRT-based secret sharing enjoys a local homomorphism property that makes it well suited for MPC: locally adding (resp., multiplying) shares yields a sharing of the sum (resp., product) of the underlying secrets.[8] The main complication is that the magnitude of the underlying integer grows with each operation. For example, if $x$ and $y$ are shared as $[\![X_t]\!]$ and $[\![Y_t]\!]$, then locally multiplying the shares corresponds to the integer $X_t \cdot Y_t$. Correctness requires that $X_t \cdot Y_t < \prod_{i=1}^{n} p_i$.

To keep the integers bounded, the parties invoke an *integer reduction* protocol after each multiplication. This procedure is analogous to degree reduction in Shamir-based MPC, except that here one reduces the integer magnitude rather than the polynomial degree. Concretely, the parties pre-generate a *double sharing* $[\![R_t]\!]$ and $[\![R_{2t}]\!]$ of a random value $r \in \mathbb{Z}_{p_0}$ and of $-r$ (i.e., of $p_0 - r$), respectively. After performing a multiplication, they reconstruct

$$[\![X_t]\!] \cdot [\![Y_t]\!] + [\![R_{2t}]\!]$$

to obtain a public value $r'$ (reduced modulo $p_0$), and then recover a bounded sharing of the result as $r' + [\![R_t]\!]$, which is a sharing of $xy - r + r = xy$ modulo $p_0$.

The key observation of [GJMW24] is that, in CRT-based secret sharing, the size of each share need not match the size of the secret. In particular, when

$$\log p_0 = \Omega(n \log n),$$ [9]

one can choose pairwise coprime moduli $p_1, \ldots, p_n$ such that

$$\log p_0 = O(\log p_1 + \log p_2 + \cdots + \log p_n).$$

In this regime, the total length of all shares is comparable to the length of the secret itself, and the total work performed by all parties to emulate a single $\mathbb{Z}_{p_0}$ gate is only $O(\log p_0)$ bit operations. Consequently, the protocol achieves an effective $O(1)$ overhead.

Crucially, unlike share packing, which achieves an *amortized* $O(1)$ overhead by evaluating batches of $\Omega(n)$ gates at once, CRT-based secret sharing instead *unpacks* a *long* secret into many *small* shares. This yields a *non-amortized* $O(1)$ overhead and completely eliminates the need to batch gates, and hence any structural requirements on the underlying circuit. Indeed, given sufficiently many (namely, $|C|$) pairs of masks $\{[\![R_t]\!], [\![R_{2t}]\!]\}$, this approach already yields a scalable MPC protocol for any arithmetic circuit $C$ over $\mathbb{Z}_{p_0}$, provided that $\log p_0 = \Omega(n \log n)$.

[8] Each party performs its local computation modulo its own modulus $p_i$.

[9] The extra $\log n$ factor arises because the moduli $p_1, \ldots, p_n$ must be distinct, which forces their sizes to grow by a factor of $\Theta(\log n)$.

**Key Technical Barrier.** The remaining challenge is how to efficiently generate the mask pairs $\{[\![R_t]\!], [\![R_{2t}]\!]\}$. Recall that one such pair is consumed per multiplication gate. Thus, in order to maintain $O(1)$ overhead, the total computational cost across all parties for generating each pair of masks must be only $O(\log p_0)$.

On the one hand, if a single party samples and distributes the masks, then the computational cost is acceptable—namely, $O(\sum_{i=1}^{n} \log p_i) = O(\log p_0)$. However, this approach is insecure, since the entire mask would be known to a single party. On the other hand, if each party independently samples its own masks

$$\left\{[\![R_t^{(i)}]\!], [\![R_{2t}^{(i)}]\!]\right\}_{i\in[n]}$$

and the parties jointly use

$$\left\{\sum_i [\![R_t^{(i)}]\!],\ \sum_i [\![R_{2t}^{(i)}]\!]\right\}$$

as the mask pair, then security is restored. However, the total cost now becomes $n \cdot \Omega(\log p_1 + \cdots + \log p_n)$, which reintroduces the undesired factor-$n$ overhead.

This technical barrier is not unique to CRT-based secret sharing. In Shamir-based MPC over finite fields, random masks are also required for degree reduction. To eliminate the factor-$n$ overhead, Damgård and Nielsen [DN07] introduced the now-standard technique of randomness extraction via *super-invertible matrices*. Specifically, they showed that if $E$ is an $(n-t)\times n$ super-invertible matrix (i.e., every $(n-t)\times(n-t)$ submatrix is invertible), then

$$E \cdot \begin{pmatrix} [\![R_t^{(1)}]\!] \\ [\![R_t^{(2)}]\!] \\ \vdots \\ [\![R_t^{(n)}]\!] \end{pmatrix} \quad \text{and} \quad E \cdot \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n)}]\!] \end{pmatrix}$$

yield $n-t$ pairs of uniformly random masks.

Moreover, super-invertible matrices are easy to construct over finite fields. For example, the following Vandermonde matrix

$$E = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^{n-t-1} & \cdots & n^{n-t-1} \end{pmatrix}$$

is super-invertible over $\mathbb{F}_p$ for any $p > n$. This construction eliminates the factor-$n$ overhead for two reasons:

1. The matrix-vector multiplication can be performed in $O(n \log n)$ computational complexity via FFT.
2. As a result, the amortized cost[10] of generating a single pair of masks is comparable to that of sampling a single instance of secret sharing.

Switching back to CRT secret sharing, the outputs we want to mask,

$$\left\{[\![X_t^{(i)}]\!] \cdot [\![Y_t^{(i)}]\!]\right\}_{i\in[|C|]},$$

are *integers*. Thus, we must extract *random integers*.[11] If we had a super-invertible matrix *over integers*, we could derive a similar result. Assume the first $n-t$ parties are honest and let $E_{n-t}$ denote the corresponding square

[10] Here, achieving $\tilde{O}(1)$ overhead via amortization does not impose any structure on the circuits; it merely requires the circuit size to be at least comparable to $n$, which is almost always the case (e.g., when each party provides one input).

[11] By random integer, we simply mean a distribution that is sufficiently random for masking purposes.

submatrix. Then:

$$E_{n-t} \cdot \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix} + \begin{pmatrix} [\![X_t^{(1)}]\!] \cdot [\![Y_t^{(1)}]\!] \\ [\![X_t^{(2)}]\!] \cdot [\![Y_t^{(2)}]\!] \\ \vdots \\ [\![X_t^{(n-t)}]\!] \cdot [\![Y_t^{(n-t)}]\!] \end{pmatrix} = E_{n-t} \cdot \left( \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix} + (E_{n-t})^{-1} \cdot \begin{pmatrix} [\![X_t^{(1)}]\!] \cdot [\![Y_t^{(1)}]\!] \\ [\![X_t^{(2)}]\!] \cdot [\![Y_t^{(2)}]\!] \\ \vdots \\ [\![X_t^{(n-t)}]\!] \cdot [\![Y_t^{(n-t)}]\!] \end{pmatrix} \right). \tag{1}$$

While there is no one-time pad over integers, the idea is that if each $R_{2t}^{(i)}$ is chosen uniformly from $\{1, 2, \ldots, L'\}$ for a sufficiently large $L'$, it can smudge the contribution of the $(E_{n-t})^{-1} \cdot (\cdots)$ term (this is commonly known as the smudging lemma, refer to Lemma 4), making the resulting distribution simulatable without the secrets. Namely,

$$E_{n-t} \cdot \left( \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix} + (E_{n-t})^{-1} \cdot \begin{pmatrix} [\![X_t^{(1)}]\!] \cdot [\![Y_t^{(1)}]\!] \\ [\![X_t^{(2)}]\!] \cdot [\![Y_t^{(2)}]\!] \\ \vdots \\ [\![X_t^{(n-t)}]\!] \cdot [\![Y_t^{(n-t)}]\!] \end{pmatrix} \right) \approx E_{n-t} \cdot \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix}.$$

However, this assumes the existence of an explicit super-invertible matrix construction over $\mathbb{Z}$, which we currently do not have. At this point, we conclude that, to achieve scalable MPC, it suffices to solve the following main technical challenge.

> **Key Question:** *Given $n$ pairs of CRT sharings $\left\{[\![R_t^{(i)}]\!], [\![R_{2t}^{(i)}]\!]\right\}_{i\in[n]}$, $t = O(n)$ of which may be corrupted, how can we extract $\Omega(n)$ pairs that are random enough to mask secret integers?*

**Limitations of Prior Solution [GJMW24].** Before we present our solution, it is helpful to recall how [GJMW24] approached this problem. They observe that any square matrix $E_{n-t}$ has an *adjugate matrix* $\mathrm{adj}(E_{n-t})$ (see Section 3.3), which satisfies

$$E_{n-t} \cdot \mathrm{adj}(E_{n-t}) = \det(E_{n-t}) \cdot I_{n-t}$$

over $\mathbb{Z}$. Then, assuming $\det(E_{n-t}) \neq 0$, *even if $E_{n-t}$ is not invertible*, we can still adapt Equation 1 as

$$E_{n-t} \cdot \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix} + \begin{pmatrix} [\![X_t^{(1)}]\!] \cdot [\![Y_t^{(1)}]\!] \\ [\![X_t^{(2)}]\!] \cdot [\![Y_t^{(2)}]\!] \\ \vdots \\ [\![X_t^{(n-t)}]\!] \cdot [\![Y_t^{(n-t)}]\!] \end{pmatrix} = E_{n-t} \cdot \left( \begin{pmatrix} [\![R_{2t}^{(1)}]\!] \\ [\![R_{2t}^{(2)}]\!] \\ \vdots \\ [\![R_{2t}^{(n-t)}]\!] \end{pmatrix} + \mathrm{adj}(E_{n-t}) \cdot \begin{pmatrix} [\![X_t^{(1)}]\!] \cdot [\![Y_t^{(1)}]\!] / \det(E_{n-t}) \\ [\![X_t^{(2)}]\!] \cdot [\![Y_t^{(2)}]\!] / \det(E_{n-t}) \\ \vdots \\ [\![X_t^{(n-t)}]\!] \cdot [\![Y_t^{(n-t)}]\!] / \det(E_{n-t}) \end{pmatrix} \right).$$

Therefore, as long as the secrets are divisible by $\det(E_{n-t})$,[12] the extracted randomness can still be used to smudge the secrets (assuming $R_{2t}^{(i)}$ are sampled from a sufficiently large range). This is referred to as the *high-dimensional smudging lemma* [GJMW24, Lemma 1]. This argument requires that no square submatrix of $E$ has determinant zero (equivalently, $E$ must be super-invertible over $\mathbb{Q}$).[13] In their construction, [GJMW24] simply use the Vandermonde matrix as $E$. While this works, it introduces two problems.

[12] Since we can only smudge *whole numbers*, it is necessary that the secrets be divisible by $\det(E_{n-t})$.

[13] This also implicitly requires $E_{n-t}$ to be super-invertible over $\mathbb{Z}_{p_0}$ (i.e., $\det(E_{n-t})$ and $p_0$ are coprime), as otherwise it is impossible to ensure that the secret integer $S$ being masked simultaneously satisfies (i) $S = s \bmod p_0$ (for arbitrary $s \in \mathbb{Z}_{p_0}$) and (ii) $S$ is divisible by $\det(E_{n-t})$.

1. **Extraordinarily large $p_0$.** The entries of $\mathrm{adj}(E_{n-t})$ can be as large as $\det(E_{n-t})$. For the Vandermonde matrix $E$, this can be as large as $n^{\Omega(n^2)}$. Consequently, for smudging purposes, each $R_{2t}^{(i)}$ must be sampled from a range much larger than $n^{\Omega(n^2)}$. By CRT correctness, this implies

$$\log p_1 + \cdots + \log p_n \;\geqslant\; \log R_{2t}^{(i)} \;=\; \Omega\big(n^2 \log n\big).$$

   Thus, to maintain $O(1)$ overhead, we must also have $\log p_0$ comparable to $\sum_i \log p_i$, i.e., $\log p_0 = \Omega\big(n^2 \log n\big)$. This creates an unreasonable requirement on the ring size relative to the number of parties, and is the key technical barrier that prevents [GJMW24] from obtaining scalable MPC for the optimal ring size $\log p_0 = O(n \log n)$.

2. **Reformatting the secrets.** As noted, the high-dimensional smudging lemma can only mask secrets divisible by $\det(E_{n-t})$. An arbitrary secret $s \in \mathbb{Z}_{p_0}$ may not satisfy this condition. Therefore, [GJMW24] enforce a *secret reformatting* step: before sharing $s$ via CRT, they first compute $\bar{s}$ such that

$$\bar{s} = \begin{cases} 0 & (\mathrm{mod}\ \det(E_{n-t})), \\ s & (\mathrm{mod}\ p_0). \end{cases}$$

   This is unsatisfying for two reasons. First, from the randomness-extraction perspective, the extracted integer masks cannot smudge arbitrary integers, but only those in a restricted subset. Second, and more importantly, since we do not know which subset of $t$ parties is corrupted, we must reformat the secret such that $\bar{s}$ is divisible by the determinant of *every square submatrix* of $E$.

   Fortunately, when $E$ is Vandermonde, the LCM of the determinants of all such submatrices can be computed explicitly as $\prod_{1 \leqslant i < j \leqslant n} (j - i)$. Two observations are important here: (i) this LCM is efficiently computable, and (ii) while large, it is still $n^{\Theta(n^2)}$, which does not qualitatively change the final asymptotic bounds in [GJMW24]. For any other choices of the super-invertible matrices $E$, it is unclear whether these two points remain true.

**A Summary Thus Far.** It is helpful to pause and collect the properties that the matrix $E$ must satisfy in order for our approach to yield a scalable MPC protocol.

1. **Small determinants.** For smudging to work efficiently, the range requirement on $R_{2t}$ depends on the determinants of square submatrices of $E$. We want these determinants to be bounded, ideally no larger than $n^{O(n)}$.

2. **Reformatting-friendly.** Either (i) we can efficiently compute the least common multiple of the determinants of all square submatrices (enabling secret reformatting),[14] or (ii) there is an alternative mechanism that eliminates the need for secret reformatting altogether.

3. **Matrix-vector multiplication friendly.** To ensure low computational overhead, the matrix-vector multiplication should be computable in near-linear time, analogous to the FFT-based efficiency of the Vandermonde matrix.

4. **Potential relaxation.** Prior work extracts exactly $(n - t)$ masks, but in fact it suffices to extract $\Omega(n - t)$ masks. This relaxation still preserves the $\tilde{O}(1)$ overhead.

**Starting Point: Randomized Matrix with Small Entries.** We begin with a natural attempt to satisfy the first property. Our goal is to construct a super-invertible matrix $E$ (over both $Q$ and $\mathbb{Z}_{p_0}$) with integer entries such that every square submatrix has determinant at most $n^{O(n)}$. A simple observation is that if each entry of $E$ is bounded by a small range (e.g., $\{1, 2, \ldots, 10n\}$), then the determinant of any submatrix is trivially bounded by $n^{O(n)}$.[15]

Unfortunately, we do not know any explicit construction of super-invertible matrices with $O(n)$-sized entries. This motivates a natural randomized approach: sample $E$ as an integer matrix with each entry chosen uniformly from, say, $\{1, 2, \ldots, 10n\}$. The hope is that, with high probability, every square submatrix is full-rank. However, this guarantee does not hold with overwhelming probability. Consider a single square submatrix and ask whether its

[14] As with the small-determinant requirement, this LCM should remain bounded by $n^{O(n)}$.

[15] This follows since the determinant is the sum of $n!$ terms, each a product of $n$ entries bounded by $O(n)$.

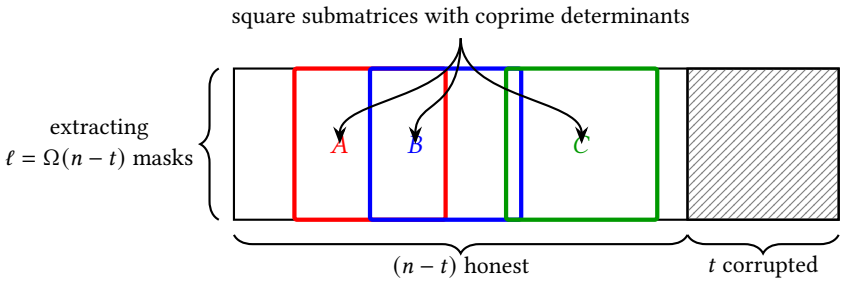


Figure 1: A Pictorial Summary of Our Strategy for Performing Linear Mask Extraction over Integers.

determinant is zero modulo $p_0$. By the Schwartz–Zippel lemma (assuming $p_0$ is prime for simplicity), the failure probability that the determinant is zero over $\mathbb{Z}_{p_0}$ will be approximately $n/(10n) = 1/10$.[16] This is far from the negligible failure probability that we desire.

At this point, our earlier observation on potential relaxation becomes useful. Suppose instead of extracting all $(n-t)$ masks, we aim to extract only $(n-t)/2$ masks. After discarding the contributions from corrupted parties, we are left with a $\frac{(n-t)}{2} \times (n-t)$ submatrix, which only needs to have rank $\frac{(n-t)}{2}$. This weaker requirement is much easier to satisfy. Indeed, one can show that, with overwhelming probability, an $\frac{(n-t)}{2} \times (n-t)$ random matrix with small entries will be full-rank. We defer a proof sketch to later (as we will prove a much stronger statement that subsumes this), but carry on with intuitive claims for now.

**Avoid Reformatting: Leveraging Bézout's Identity.** Now we have a candidate matrix $E \in \mathbb{Z}^{\frac{(n-t)}{2} \times n}$, where every $\frac{(n-t)}{2} \times (n-t)$ submatrix is full-rank with high probability. We next address the secret reformatting issue. As far as we know, there is no efficient algorithm for computing the LCM of the determinants of all square submatrices, so we have to resort to a different approach.

Here, the relaxation again comes to our aid. By working with a $\frac{(n-t)}{2} \times (n-t)$ matrix, not only is it full-rank with high probability, but we can in fact expect the existence of *multiple* full-rank submatrices. Figure 1 illustrates this situation, where the matrix contains three full-rank submatrices $A$, $B$, and $C$. By the high-dimensional smudging lemma, the matrix $A$ can absorb any shift divisible by $\det(A)$ into the randomness of the extracted mask. Formally,

$$(\cdots, A, \cdots) \cdot \begin{pmatrix} \vdots \\ R^{(A)}_{2t} \\ \vdots \end{pmatrix} + \begin{pmatrix} \Delta_1 \\ \Delta_2 \\ \vdots \\ \Delta_{\frac{n-t}{2}} \end{pmatrix} = (\cdots, A, \cdots) \cdot \begin{pmatrix} \vdots \\ R^{(A)}_{2t} + M \\ \vdots \end{pmatrix},$$

where

$$M = \mathrm{adj}(A) \cdot \begin{pmatrix} \Delta_1/\det(A) \\ \Delta_2/\det(A) \\ \vdots \\ \Delta_{\frac{n-t}{2}}/\det(A) \end{pmatrix}.$$

Analogous statements hold for $B$ and $C$. Thus, any shift divisible by $\det(A)$, $\det(B)$, or $\det(C)$ is unnoticeable.

At this point it is natural to leverage *Bézout's identity* (refer to Lemma 5) to express any secret $\Delta$ as

$$\Delta = \alpha \cdot \det(A) + \beta \cdot \det(B) + \gamma \cdot \det(C).$$

For any $\Delta$ that can be expressed as above, we can repeatedly apply the high-dimensional smudging lemma to argue that a shift by $\Delta$ is also unnoticeable. By Bézout's identity, this is possible whenever $\Delta$ is divisible by the GCD of

[16] This is informal since Schwartz–Zippel only gives an upper bound, but not a lower bound. Nonetheless, it serves as a useful sanity check.

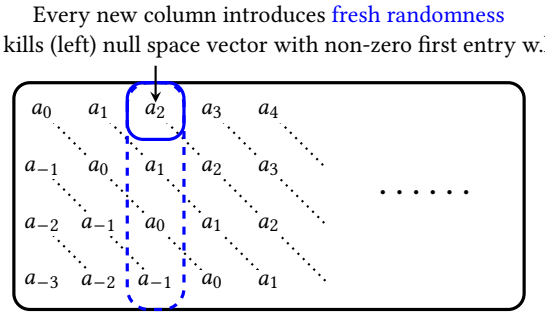


Figure 2: A Pictorial Representation of Toeplitz Matrix and Our Proof Strategy.

$\det(A)$, $\det(B)$, and $\det(C)$. Moreover, if $\det(A)$, $\det(B)$, and $\det(C)$ are coprime, then the extracted mask can hide *any* integer secret. This completely lifts the need for reformatting the secret.

The range requirement on $R_{2t}$ in the above argument depends on the magnitudes of the Bézout coefficients $(\alpha, \beta, \gamma)$ as well as on the sizes of the adjugate matrices of $A$, $B$, and $C$. Ideally, all of these quantities should be bounded by $n^{O(n)}$. Accordingly, it suffices to establish the existence of a small number of square submatrices that satisfy the following properties: (i) they are full rank, (ii) their determinants are collectively coprime, and (iii) they admit Bézout coefficients of sufficiently small magnitude.

In general, if $x_1, x_2, \ldots, x_m$ are coprime integers, Bézout's identity guarantees the existence of coefficients $\{\alpha_i\}$ satisfying

$$\sum_{i=1}^{m} \alpha_i x_i = 1 \qquad \text{with} \qquad |\alpha_i| \leqslant \prod_{j \neq i} x_j.$$

However, as we will see later, this generic bound is *not* sufficient for our purposes. We therefore highlight this issue here and return to it later.

**Fast Matrix-Vector Multiplication via Toeplitz Matrix.** We now turn to the final requirement: fast matrix-vector multiplication. If the matrix $E$ is completely unstructured, then achieving a fast algorithm is clearly impossible. Thus, $E$ must be sampled from a suitably structured distribution. In this work, we sample $E$ as a random *Toeplitz matrix* (see Figures 2 and 4). A key advantage of Toeplitz matrices is that they admit fast matrix-vector multiplication algorithms [FK23, Section 4], thereby satisfying our complexity requirement. However, this choice also introduces a new technical challenge. Specifically, all of our earlier claims must now hold for a *random Toeplitz matrix*, rather than for a fully random matrix.

**Formalizing the Proof.** Based on the discussion so far, the following statement suffices for our purposes.[17] To establish this result, we introduce Claim 1. We first explain why this claim is sufficient for our purposes, and then outline how it can be proved.

Let $E$ be a random Toeplitz matrix of dimension $\Omega(m) \times m$, where each nonzero entry is sampled from a bounded range of size $O(m)$. Then there exist square submatrices $E_1, E_2, \ldots, E_t$ of $E$ and integers $\alpha_1, \ldots, \alpha_t$ such that

$$\sum_{i=1}^{t} \alpha_i \cdot \det(E_i) = 1,$$

with

$$|\alpha_i| \leqslant m^{O(m)} \qquad \text{for all } i \in [t].$$

[17] To present this result as a self-contained statement independent of the MPC context, we formulate it in terms of a parameter $m$. In our MPC application, one has $m = \Theta(n)$.

**Claim 1.** *Fix any prime $q$. For a random Toeplitz matrix $E$ of dimension $\Omega(m) \times m$, where each nonzero entry is sampled from a bounded $O(m)$ range, there exists (with overwhelming probability) a square submatrix $E_0$ that is full-rank over $\mathbb{Z}_q$.*

Step I: Why is this claim sufficient? Suppose we sample two independent matrices $E'$ and $E''$ of the prescribed dimension and let $E = [E', E'']$. We argue that $E$ then satisfies the desired property. It involves the following steps.

- **$E'$ contains a submatrix with nonzero determinant over $\mathbb{Z}$.** This follows directly from the claim: if $E'$ contains a full-rank submatrix $E_0$ over some $\mathbb{Z}_q$, then $\det(E_0)$ must be nonzero over $\mathbb{Z}$.
- **For each prime factor of $\det(E_0)$, $E''$ contains a full-rank submatrix over that modulus.** Since $\det(E_0) = m^{O(m)}$, it has at most $m \log m$ distinct prime factors $q_1, \ldots, q_t$, where $t \leqslant m \log m$. For each $q_i$, we invoke the claim on $E''$ to obtain a full-rank submatrix $E_i$ over $\mathbb{Z}_{q_i}$. By a union bound, this holds simultaneously for all $i \in [t]$ with overwhelming probability.
- **Bézout coefficients are not too large.** We now have $t + 1$ submatrices: $E_0$ from $E'$ and $E_1, \ldots, E_t$ from $E''$, such that for each prime factor $q_i$ of $\det(E_0)$, the determinant $\det(E_i)$ is not divisible by $q_i$. Thus,

$$\det(E_0),\ \det(E_1),\ \ldots,\ \det(E_t) \quad \text{are coprime.}$$

  To complete the proof, it suffices to find $\alpha_i$ such that

$$\sum_{i=0}^{t} \alpha_i \cdot \det(E_i) = 1$$

  with $|\alpha_i| = m^{O(m)}$ for every $i \in \{0, 1, \ldots, t\}$. Note that a generic Bézout identity would upper-bound the coefficients by $\prod_{i=0}^{t} |\det(E_i)| \leqslant (m^m)^t \leqslant m^{m^2 \log m}$, which is far too large.

  Fortunately, we can argue the desired bound as follows. Write $\det(E_0) = \prod_{i=1}^{t} q_i^{d_i}$. Since every $\det(E_i)$ is not divisible by $q_i$ for each $i \in \{1, \ldots, t\}$, we can find $\alpha_i$ with $|\alpha_i| \leqslant \det(E_0)$ (via CRT) such that

$$\alpha_i \cdot \det(E_i) \equiv \begin{cases} 1 & (\bmod\ q_i^{d_i}), \\ 0 & (\bmod\ q_j^{d_j}) \quad (\forall j \neq i). \end{cases}$$

  Now, summing over $i$ gives

$$\sum_{i=1}^{t} \alpha_i \cdot \det(E_i) \equiv 1 \pmod{\det(E_0)},$$

  which implies the existence of $\alpha_0 = m^{O(m)}$ such that

$$\sum_{i=0}^{t} \alpha_i \cdot \det(E_i) = 1 \qquad \text{over } \mathbb{Z}.$$

Step II: How to prove this claim? We now outline how to prove Claim 1. In this part, we will use $\ell = \Theta(m)$ to denote the number of rows of the matrix $E$. Our key ideas are as follows (see also Figure 2). We view the sampling of a random Toeplitz matrix as an $m$-step process, where in each step a new column is sampled from left to right. The crucial observation is that each newly sampled column introduces fresh independent randomness, which we exploit to argue that the rank of the matrix increases with high probability.

More concretely, suppose that after sampling several columns, the current partial matrix $E$ is not yet full rank. Let $u$ be a nonzero vector in its left null space. Consider the next sampled column $c$. The vector $u$ will cease to lie in the null space provided that

$$\langle u, c \rangle \neq 0.$$

By the Toeplitz structure, the first coordinate of $c$ is freshly and independently sampled. Consequently, by the Schwartz-Zippel lemma, the event $\langle u, c\rangle = 0$ occurs with probability at most $O(1/m)$. It follows that each newly sampled column increases the rank with probability $\geqslant 1-O(1/m)$. After sampling $m$ additional columns, a Chernoff bound implies that, with high probability, the resulting matrix will reach rank-$\ell$, completing the proof.

This provides the high-level intuition for the proof. We now comment on two technical subtleties that arise in formalizing this intuition.

- *What if the first entry of $u$ is zero?* In the above Schwartz-Zippel argument, we implicitly assumed that the first coordinate of the null-space vector $u$ is nonzero, which need not hold in general. To address this issue, our formal proof decomposes the bad event (i.e., that $E$ is not full rank) into $\ell$ disjoint events $\mathsf{Bad}_1, \mathsf{Bad}_2, \ldots, \mathsf{Bad}_\ell$. Here, the event $\mathsf{Bad}_i$ occurs if there exists a nonzero vector $u$ in the left null space of $E$ such that

$$u = (\underbrace{0, \ldots, 0}_{i-1 \text{ entries}}, \neq 0, \ldots).$$

  Our earlier argument directly implies that $\mathsf{Bad}_1$ occurs only with negligible probability. By applying the same reasoning iteratively, we can similarly bound the probabilities of $\mathsf{Bad}_2, \ldots, \mathsf{Bad}_\ell$. Finally, a union bound over all $\ell$ events completes the argument.

- *Why does the Chernoff bound apply?* The Chernoff bound requires independence, whereas in our argument the events that successive columns increase the rank appear highly correlated. For instance, once the matrix becomes full rank, subsequent columns can no longer increase the rank. Thus, it is not immediately clear how to justify the use of a Chernoff bound.

  One can address this issue by a standard *decoupling argument*. For instance, suppose we are trying to bound the failure probability of event $\mathsf{Bad}_1$. For each column index $i$, let us define $b_i$ to be an independent Bernoulli random variable with bias $1 - O(1/m)$. We now define an alternative procedure for sampling the $i$-th column as follows:

  1. If the current left null space contains no vector whose first coordinate is nonzero, we ignore $b_i$ and sample the new column according to the original Toeplitz distribution.
  2. Otherwise, let $u$ be a fixed vector in the null space (in particular, we may take the lexicographically first such vector) whose first entry is nonzero. We then sample the new column $c$ according to the original distribution, but condition on the randomness variable $b_i$ as follows.[18]

$$\langle u, c\rangle = \begin{cases} 0, & \text{if } b_i = 0, \\ \neq 0, & \text{if } b_i = 1. \end{cases}$$

  We make the following observations. (i) First, this alternative sampling procedure preserves the original distribution of each column $c$. (ii) Second, the random variables $\{b_i\}$ are independent, and hence the Chernoff bound applies to their sum. (iii) Finally, if $\sum_i b_i \geqslant \ell$, then the event $\mathsf{Bad}_1$ cannot occur. Together, these points justify the application of a Chernoff bound in our argument.

  This completes the high-level idea of proving Claim 1.

**Achieving Adaptive Security.** Finally, an astute reader may observe that the argument so far only establishes *static* security (as discussed in Remark 1, achieving adaptive security is the non-trivial part). Indeed, Claim 1 suffices to show that a random $\ell \times (n - t)$ submatrix is full rank with high probability. For adaptive security, however, we require a significantly stronger guarantee: after sampling a random Toeplitz matrix $E$ of dimension $\ell \times n$, *every*

[18] The description below is slightly informal. The Schwartz-Zippel lemma only guarantees that the equation $\langle u, c\rangle = 0$ has *at most* one root as we pick the fresh randomness for the first entry of $c$. Thus, over the bounded range from which we sample the randomness, if such a root exists, excluding it suffices. However, it is also possible that no root exists at all, that is, $\langle u, c\rangle$ is never zero over the entire range. In this case, the new column never annihilates the fixed null-space vector $u$, and we simply ignore the conditioning on $b_i$ and sample $c$ according to the original distribution.

$\ell \times (n-t)$ *submatrix* of $E$ must be full rank. Equivalently, the full-rank property must hold simultaneously for all possible choices of the corrupted set of $t$ parties.

A natural approach is to apply a union bound over all choices of $(n-t)$ columns. The number of such submatrices is $\binom{n}{t} \approx 2^n$. Consequently, the failure probability in our earlier analysis must be sufficiently small to absorb this additional factor of $2^n$.

Interestingly, with careful bookkeeping of constants and probability bounds, our earlier argument in fact tolerates the union bound for all primes $q$ *except* the small primes $q \leqslant 16$.[19] For these small primes, we therefore give a separate argument that explicitly exploits the small alphabet size.

*The small-q argument.* Fix a nonzero null vector $u \in \mathbb{F}_q^\ell$. A column $c$ is orthogonal to $u$ with probability (approximately) $1/q$. Hence, among the $n$ columns, the number of columns orthogonal to $u$ follows a binomial distribution with mean $n/q$.

On the other hand, the protocol *fails* to achieve adaptive security only if there *exists a nonzero vector u such that fewer than $\ell$ columns are* non-*orthogonal to u*. Equivalently, the number of columns orthogonal to $u$ exceeds $n-\ell$.

By a Chernoff bound, for any fixed nonzero $u$, the probability of such a large deviation is exponentially small in $n$. We then apply a union bound over all nonzero null vectors. Since there are at most $q^\ell$ such vectors, the total failure probability is bounded by

$$q^\ell \cdot \exp(-\Omega(n)).$$

For $q \leqslant 16$, we can choose $\ell = c \cdot n$ for a suitably small constant $c > 0$ so that this expression remains exponentially small in $n$. The main intuition behind this argument is that we have effectively replaced the earlier union bound over $2^n$ possible choices of corrupted subsets with a union bound over $q^\ell$, where $\ell = \Theta(n)$ and the constant in the exponent is under our control.

We emphasize that this argument fundamentally relies on $q$ being a small constant. If $q$ were large (which could be as large as $n^n$), then $q^\ell$ would be too large for this argument to work.

Taken together, our general-prime argument and the tailor-made small-$q$ argument enable us to establish the adaptive security of our randomness extraction scheme. We defer the full details to Section 4.

# 3 Preliminaries

We begin by introducing the notation we use in this work and then review other relevant preliminaries.

*General Notation.* We use $\kappa$ to denote the statistical security parameter. For any $x, y \in \mathbb{Z}$, where $x \leqslant y$, we use $[x, y] = \{x, \ldots, y\}$ to denote consecutive integers from $x$ to $y$. For $y \geqslant 1$, we use $[y]$ as shorthand to denote $[1, y]$. We use the convention that a product with no terms equals 1, i.e., $\prod_{i \in \emptyset} x_i = 1$. For an integer $p \geqslant 2$, whenever we regard an element of $\mathbb{Z}_p$ as an integer, we use its canonical representative in $[0, p-1]$. For any $x, y \in \mathbb{R}$, where $x \leqslant y$, we use $(x, y)$ to denote the open interval $\{z \mid z \in \mathbb{R} \ \wedge \ x < z < y\}$. For $x > 1$, we use $\log_x y$ to denote the logarithm of $y$ base $x$. We omit the base and use $\log y$ to denote logarithms base 2 and use $\ln y$ for the natural logarithm.

*Linear Algebra.* Let $\mathcal{R}$ be a commutative ring. We use bold lowercase letters $\mathbf{x} \in \mathcal{R}^\ell$ for vectors and bold uppercase letters $\mathbf{M} \in \mathcal{R}^{\ell \times n}$ for matrices. For $i \in [\ell]$, we use $\mathbf{x}_i$ to denote the $i$-th element of $\mathbf{x}$, and for an index set $\mathcal{I} \subseteq [\ell]$, we use $\mathbf{x}_\mathcal{I} = (\mathbf{x}_i)_{i \in \mathcal{I}}$ to denote the subvector corresponding to $\mathcal{I}$, with the elements ordered by increasing index.

All vectors are column vectors by default. We use $\mathbf{x}^\top$ to denote the transpose of a vector $\mathbf{x}$ and $\mathbf{M}^\top$ to denote the transpose of a matrix $\mathbf{M}$. For a matrix $\mathbf{M} \in \mathcal{R}^{\ell \times n}$ and integers $i \in [\ell]$ and $j \in [n]$, $\mathbf{M}_{i,j} \in \mathcal{R}$ denotes the element in the $i$-th row and $j$-th column, $\mathbf{M}_{i,*} \in \mathcal{R}^{1 \times n}$ denotes the $i$-th row, and $\mathbf{M}_{*,j} \in \mathcal{R}^{\ell \times 1}$ denotes the $j$-th column. We extend the notation to index sets: for $\mathcal{I} \subseteq [\ell]$ and $\mathcal{J} \subseteq [n]$, $\mathbf{M}_{\mathcal{I},\mathcal{J}}$ denotes the $|\mathcal{I}| \times |\mathcal{J}|$ submatrix of $\mathbf{M}$ formed by rows indexed by $\mathcal{I}$ and columns indexed by $\mathcal{J}$. Similarly, $\mathbf{M}_{\mathcal{I},*}$ denotes the submatrix consisting of the rows indexed by $\mathcal{I}$ and $\mathbf{M}_{*,\mathcal{J}}$ denotes the submatrix consisting of the columns indexed by $\mathcal{J}$. For $i \in [\ell]$ and $j \in [n]$, we use $\mathbf{M}_{-i,-j}$ to denote the $(\ell-1) \times (n-1)$ matrix obtained by deleting the $i$-th row and $j$-th column of $\mathbf{M}$. For matrices $\mathbf{M}^{(1)} \in \mathcal{R}^{\ell \times n_1}, \ldots, \mathbf{M}^{(k)} \in \mathcal{R}^{\ell \times n_k}$, we use $\left[\mathbf{M}^{(1)} \mid \cdots \mid \mathbf{M}^{(k)}\right] \in \mathcal{R}^{\ell \times (n_1 + \cdots + n_k)}$ to denote column-wise concatenation

[19] One intuition for why the argument breaks down in this regime is the following. When $q$ is very small, the independent failure probability is no longer governed by $1/m$, but rather by $1/q$. As the bias of the i.i.d. Bernoulli variables approaches $1/2$, the Chernoff bound no longer yields a sufficiently strong exponential tail bound to absorb the $2^n$ union bound. See Lemma 6.

of the matrices. We use $\mathrm{rank}_{\mathcal{R}}(\mathbf{M})$ to denote the rank of $\mathbf{M}$ over $\mathcal{R}$, and when $\mathbf{M}$ is a square matrix, we use $\det(\mathbf{M})$ to denote its determinant. For a matrix $\mathbf{M}$ over $\mathbb{Z}$ and an integer $p \geqslant 1$, we use $\mathbf{M} \bmod p$ to denote the entry-wise reduction of $\mathbf{M}$ modulo $p$.

For a vector $\mathbf{x} \in \mathbb{R}^{\ell}$ and a matrix $\mathbf{M} \in \mathbb{R}^{\ell \times n}$, we define $\|\mathbf{x}\|_{\infty} \coloneqq \max_{i \in [\ell]} |\mathbf{x}_i|$ and $\|\mathbf{M}\|_{\infty} \coloneqq \max_{i \in [\ell],\, j \in [n]} |\mathbf{M}_{i,j}|$. We call $\mathbf{M}$ $B$-bounded if $\|\mathbf{M}\|_{\infty} \leqslant B$.

*Probability.* For a distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes sampling $x$ according to $\mathcal{D}$. We extend the notation and use $x \leftarrow S$ to denote sampling $x$ uniformly at random from a finite set $S$. For $\ell \in \mathbb{N}$, we use $\mathbf{x} \leftarrow \mathcal{D}^{\ell}$ to denote $\ell$ independent samples from $\mathcal{D}$. We use $x \leftarrow \mathcal{A}$ to denote assignment from a randomized algorithm $\mathcal{A}$ and $x \coloneqq y$ to denote initialization of $x$ to the value of $y$ (which may be the output of a deterministic algorithm).

Let $\Pr[E : A]$ denote the probability of an event $E$ in an experiment defined by executing $A$. For an event $E$, we use $\mathbb{1}[E]$ to denote its indicator, which equals 1 if $E$ occurs and 0 otherwise. For two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over the same support $S$, their *statistical distance* is defined as $\mathrm{SD}(\mathcal{D}_1, \mathcal{D}_2) = \frac{1}{2} \sum_{s \in S} |\Pr[\mathcal{D}_1 = s] - \Pr[\mathcal{D}_2 = s]|$. We slightly abuse notation and use $\mathrm{SD}(f(x), g(y))$ to denote the statistical distance between the distributions of $f(x)$ and $g(y)$, where $x$ and $y$ are sampled from distributions that are clear from context. For two probability ensembles $\{A_\kappa\}_\kappa$ and $\{B_\kappa\}_\kappa$, we write $\{A_\kappa\}_\kappa \equiv \{B_\kappa\}_\kappa$ to denote that the ensembles are identical and $\{A_\kappa\}_\kappa \stackrel{\mathrm{s}}{\approx} \{B_\kappa\}_\kappa$ to denote that the ensembles are statistically indistinguishable i.e., $\mathrm{SD}(A_\kappa, B_\kappa)$ is negligible in $\kappa$.

*Algorithms.* We measure running times in the bit-cost RAM model. In particular, arithmetic on variable-length integers is charged according to its bit complexity.

## 3.1 Secure Multiparty Computation

We use the standard $n$-party model for MPC [Gol04]. Each party $\mathsf{P}_i$ holds an input $x_i$, participates in the computation, and receives an output. We assume private point-to-point channels and a synchronous network.

We consider an adaptive, semi-honest adversary that corrupts at most $t$ parties in total. Corrupt parties follow the protocol as prescribed, but reveal their inputs, internal randomness, and all messages received so far to the adversary. The adversary may choose which parties to corrupt adaptively, i.e., as a function of the views it has observed so far. For simplicity, we restrict attention to deterministic functionalities with a common output.

**Real and ideal experiments.** Let $\mathcal{A}$ be a (computationally unbounded) interactive adversary. In the real experiment $\mathsf{Real}^{\pi}_{\mathcal{A}}(1^{\kappa}, \mathbf{x})$, the protocol $\pi$ is executed on input vector $\mathbf{x}$ and security parameter $\kappa$. At any point before, during, or after the execution, $\mathcal{A}$ may issue a corruption request for a party $\mathsf{P}_i$ with $i \in [n]$, subject to the constraint that the set $\mathcal{I}$ of corrupted parties never exceeds size $t$. Upon corrupting $\mathsf{P}_i$, the adversary receives the current view of $\mathsf{P}_i$, consisting of $x_i$, the internal randomness of $\mathsf{P}_i$, and all messages $\mathsf{P}_i$ has received so far; thereafter, $\mathcal{A}$ also receives every message $\mathsf{P}_i$ receives in later rounds. The output of the experiment is the output of $\mathcal{A}$, which without loss of generality is its entire view, including the sequence of corruption requests and the rounds at which they were made.

In the ideal experiment $\mathsf{Ideal}^{f}_{\mathsf{Sim},\mathcal{A}}(1^{\kappa}, \mathbf{x})$, a simulator $\mathsf{Sim}$ interacts with $\mathcal{A}$. At the outset, $\mathsf{Sim}$ receives $1^{\kappa}$ and $f(\mathbf{x})$. Whenever $\mathcal{A}$ issues a corruption request for $\mathsf{P}_i$, $\mathsf{Sim}$ learns $x_i$ and must return a simulated view of $\mathsf{P}_i$; it may continue to send simulated later-round messages for $\mathsf{P}_i$ to $\mathcal{A}$. The output of the experiment is again the output of $\mathcal{A}$. Since corrupt parties behave honestly, the execution of $\pi$ does not depend on $\mathcal{A}$; adaptivity affects only which views are revealed to $\mathcal{A}$ and when.

**Definition 1** (Adaptive Semi-Honest Multiparty Computation)**.** *Let $f : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}$ be a deterministic functionality and let $\varepsilon \geqslant 0$. A protocol $\pi$ realizes $f$ $\varepsilon$-securely against adaptive semi-honest adversaries with corruption threshold $t$ if it satisfies the following properties.*

- ***Perfect correctness:*** *For every $\mathbf{x} \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$,*

$$\Pr\left[\forall i \in [n] : \mathsf{Output}^{\pi}_i(1^{\kappa}, \mathbf{x}) = f(\mathbf{x})\right] = 1.$$

- *$\varepsilon$-**Statistical security:** There exists a (possibly computationally unbounded) simulator $\mathsf{Sim}$ such that, for every adaptive semi-honest adversary $\mathcal{A}$ corrupting at most $t$ parties and every $\mathbf{x} \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$,*

$$\mathrm{SD}\Big(\mathsf{Ideal}^{f}_{\mathsf{Sim},\mathcal{A}}(1^{\kappa}, \mathbf{x}), \mathsf{Real}^{\pi}_{\mathcal{A}}(1^{\kappa}, \mathbf{x})\Big) \leqslant \varepsilon.$$

## 3.2 Chinese Remainder Theorem-based Secret Sharing

We first recall the definition of a ramp threshold secret-sharing scheme and then describe the Chinese Remainder Theorem (CRT)-based scheme used in this work.

**Definition 2** (Ramp Secret Sharing Scheme). *Let $n \in \mathbb{N}$ denote the number of parties, $t_{\mathsf{rec}} \in \mathbb{N}$ denote the reconstruction threshold, and $t_{\mathsf{priv}} \in \mathbb{Z}_{\geqslant 0}$ denote the privacy threshold such that $0 \leqslant t_{\mathsf{priv}} < t_{\mathsf{rec}} \leqslant n$. Let $\mathcal{R}_0, \ldots, \mathcal{R}_n$ be finite commutative rings. An $(n, t_{\mathsf{priv}}, t_{\mathsf{rec}})$-*ramp secret-sharing scheme *with secret space $\mathcal{R}_0$ and share space $\mathcal{R}_1 \times \ldots \times \mathcal{R}_n$ consists of a tuple of algorithms* $(\mathsf{Share}, \mathsf{Recon})$ *with the following properties.*

- ***Syntax:***
  - $\mathsf{Share}(s) \to [\![s]\!]$ *is a randomized algorithm that takes a secret $s \in \mathcal{R}_0$ as input and outputs a tuple of shares $[\![s]\!] \in \mathcal{R}_1 \times \cdots \times \mathcal{R}_n$.*
  - $\mathsf{Recon}(\mathcal{I}, [\![s]\!]_{\mathcal{I}}) =: s$ *is a deterministic algorithm that takes an index set $\mathcal{I} \subseteq [n]$ and the corresponding shares $[\![s]\!]_{\mathcal{I}}$ as input and outputs a value $s \in \mathcal{R}_0$.*
- ***Perfect correctness:*** *For all $s \in \mathcal{R}_0$ and $\mathcal{I} \subseteq [n]$ such that $|\mathcal{I}| \geqslant t_{\mathsf{rec}}$,*

$$\Pr\left[\mathsf{Recon}(\mathcal{I}, [\![s]\!]_{\mathcal{I}}) = s : [\![s]\!] \leftarrow \mathsf{Share}(s)\right] = 1.$$

- *$\varepsilon$-**Statistical privacy:*** *For all $s, s' \in \mathcal{R}_0$ and $\mathcal{I} \subseteq [n]$ such that $|\mathcal{I}| \leqslant t_{\mathsf{priv}}$,*

$$\mathsf{SD}\left([\![s]\!]_{\mathcal{I}}, [\![s']\!]_{\mathcal{I}}\right) \leqslant \varepsilon,$$

  *where $[\![s]\!] \leftarrow \mathsf{Share}(s)$ and $[\![s']\!] \leftarrow \mathsf{Share}(s')$.*

**CRT-based sharing.** We use the CRT-based ramp secret-sharing scheme [Mig83, AB83, GRS99, GJM+23, GJMW24], which we describe in Figure 3. The scheme is parameterized by pairwise coprime positive integers $p_0, p_1, \ldots, p_n$ and a positive integer $L$. The secret space is $\mathbb{Z}_{p_0}$ and for each $i \in [n]$, party $\mathsf{P}_i$ receives a share in $\mathbb{Z}_{p_i}$. Note that $p_0$ is not required to be prime. Let

$$P_{\mathsf{all}} \coloneqq \prod_{i \in [n]} p_i, \qquad P_{\mathsf{min}} \coloneqq \min_{\substack{\mathcal{I} \subseteq [n] \\ |\mathcal{I}| \geqslant t_{\mathsf{rec}}}} \prod_{i \in \mathcal{I}} p_i, \qquad P_{\mathsf{max}} \coloneqq \max_{\substack{\mathcal{I} \subseteq [n] \\ |\mathcal{I}| \leqslant t_{\mathsf{priv}}}} \prod_{i \in \mathcal{I}} p_i,$$

where the empty product is 1.

The following lemma captures the correctness and privacy of the construction.

**Theorem 1** (CRT Secret Sharing [GRS99, GJM+23]). *Fix positive integers $n, L$, thresholds $0 \leqslant t_{\mathsf{priv}} < t_{\mathsf{rec}} \leqslant n$, and pairwise coprime positive integers $p_0, p_1, \ldots, p_n$. The scheme described in Figure 3 is perfectly correct if $Lp_0 \leqslant P_{\mathsf{min}}$ and is $\varepsilon$-statistically private for every $\varepsilon \geqslant P_{\mathsf{max}}/L$.*

**Remark 2** (Efficiency of CRT-based Secret Sharing). *As discussed in [GJMW24], naively computing each share $S \bmod p_i$ independently incurs an additional factor of $n$ in the running time. This factor can be avoided by arranging the reductions in a balanced remainder tree [Ber04]. The resulting running time of* Share *is $O(\log P_{\mathsf{all}} \cdot \log n)$, where we suppress factors of order $O(\log \log P_{\mathsf{all}})$. Applying the same technique to reconstruction yields $O(\log P_{\mathsf{all}} \cdot \log n)$ running time for* Recon. *We refer the reader to [GJMW24] for more details.*

## 3.3 Useful Facts

We next recall a number of facts and lemmas that will be used in the technical sections.

**Determinants of bounded matrices.** The dimension of an integer matrix and the magnitudes of its entries yield a simple bound on its determinant.

**CRT-based Secret Sharing**

**Public parameters:** The number of parties $n$, pairwise coprime positive integers $p_0, p_1, \ldots, p_n$, and a positive integer $L$.

- $\underline{\mathsf{Share}(s) \to [\![s]\!]}$
  1: Sample $u \leftarrow [0, L-1]$ uniformly at random.
  2: Set $S \coloneqq s + up_0$.
  3: For every $i \in [n]$, compute $s_i \coloneqq S \bmod p_i$ and output $[\![s]\!] = (s_1, \ldots, s_n)$.
- $\underline{\mathsf{Recon}(\mathcal{I}, [\![s]\!]_{\mathcal{I}}) \to s}$
  1: Use CRT to compute the unique integer $S$ satisfying $0 \leqslant S < \prod_{i \in \mathcal{I}} p_i$ and $S \equiv [\![s]\!]_i \pmod{p_i}$ for every $i \in \mathcal{I}$.
  2: Output $S \bmod p_0$.

Figure 3: The CRT-based $(n, t_{\mathsf{priv}}, t_{\mathsf{rec}})$-ramp secret-sharing scheme.

**Fact 2.** *Let $\mathbf{M} \in \mathbb{Z}^{\ell \times \ell}$ and $B \in \mathbb{N}$. If $\|\mathbf{M}\|_\infty \leqslant B$, then $|\det(\mathbf{M})| \leqslant \ell!\, B^\ell \leqslant (\ell B)^\ell$.*

**Adjugates of bounded matrices.** Given an integer-valued matrix $\mathbf{M}$ with non-zero determinant, it is not always possible to find an integer-valued inverse of $\mathbf{M}$. The adjugate $\mathrm{adj}(\mathbf{M})$ serves as a substitute for the inverse of an integer matrix. For every $\mathbf{M} \in \mathbb{Z}^{\ell \times \ell}$, its adjugate satisfies

$$\mathrm{adj}(\mathbf{M}) \cdot \mathbf{M} = \mathbf{M} \cdot \mathrm{adj}(\mathbf{M}) = \det(\mathbf{M}) \cdot I_\ell.$$

Moreover, each entry of the adjugate is a signed determinant of the corresponding minor, i.e.,

$$\mathrm{adj}(\mathbf{M})_{i,j} = (-1)^{i+j} \det\left(\mathbf{M}_{-j,-i}\right),$$

where $\mathbf{M}_{-j,-i}$ denotes the submatrix of $\mathbf{M}$ without its $j$-th row and $i$-th column. Fact 2 then yields a bound on each entry of the adjugate.

**Fact 3.** *Let $\mathbf{M} \in \mathbb{Z}^{\ell \times \ell}$ and $B \in \mathbb{N}$. If $\|\mathbf{M}\|_\infty \leqslant B$, then for all $i, j \in [\ell]$ $\left|\mathrm{adj}(\mathbf{M})_{i,j}\right| \leqslant (\ell-1)!\, B^{\ell-1} \leqslant (\ell B)^\ell$.*

**Smudging lemma.** Informally, a uniformly random integer $s$ can be used to statistical mask a secret $x$ when its sampling interval is sufficiently larger than $|x|$.

**Lemma 4** (Smudging Lemma [AJL+12])**.** *Let $B_{\mathsf{rand}} \in \mathbb{N}$ and $x \in \mathbb{Z}$. If $s \leftarrow [0, B_{\mathsf{rand}} - 1]$, then*

$$\mathrm{SD}(s, s+x) \leqslant \frac{|x|}{B_{\mathsf{rand}}}.$$

**Bézout's identity.** Bézout's identity states that integers whose greatest common divisor is 1 admit an integer linear combination equal to 1. Moreover, the coefficients can be chosen with magnitudes bounded in terms of the input integers.

**Lemma 5** (Bézout's Identity)**.** *Let $x_0, \ldots, x_d \in \mathbb{Z}$ be non-zero integers such that $\gcd(x_0, \ldots, x_d) = 1$. There exist $\eta_0, \ldots, \eta_d \in \mathbb{Z}$ such that*

$$\sum_{i=0}^{d} \eta_i \cdot x_i = 1, \qquad \text{and} \qquad \forall i \in [0, d] \ : \ |\eta_i| \leqslant \prod_{\substack{j=0 \\ j \neq i}}^{d} |x_j|.$$

**Chernoff–Hoeffding bound.** We also use the following standard Chernoff–Hoeffding bound for the sum of independent Bernoulli random variables.

**Lemma 6** (Chernoff–Hoeffding Bound)**.** *Let $X_1, \ldots, X_n$ be independent Bernoulli random variables, let $X = \sum_{i=1}^{n} X_i$, and let $\mu = \mathbb{E}[X]$. For every $t > 0$,*

$$\Pr[X \leqslant \mu - t] \leqslant \exp\left(-\frac{2t^2}{n}\right) \qquad \text{and} \qquad \Pr[X \geqslant \mu + t] \leqslant \exp\left(-\frac{2t^2}{n}\right).$$

# 4 Linear Mask Extractors

In this section, we define and construct linear mask extractors. We first define linear mask extractors in Section 4.1 and provide a general template for constructing them in Section 4.2. In Section 5, we provide an explicit instantiation of the template using Toeplitz matrices. In particular, this instantiation has the additional property that extraction takes quasilinear time in the output, which will be useful for our MPC protocol.

## 4.1 Definition

A linear mask extractor is similar to a seeded linear randomness extractor over the *integers*. While constructions of linear randomness extractors are known over finite fields, the notion is not well-defined over integers since there is no uniform distribution over the entire domain. Instead, we require the extraction output to suffice for *masking*: using it to mask two different bounded secrets should yield statistically close distributions.

Thus, a linear mask extractor lets us use a high-entropy source to mask values, where both the source and the values being masked consist of bounded integers. In our MPC protocol, the source consists of a random contribution from each party and we use linear mask extractors to extract masks for masking intermediate values of the computation. A minority of these contributions come from corrupt parties and are therefore fixed by the adversary's view while the honest parties' contributions remain uniformly random and independent even conditioned on this view. Since our protocol uses CRT-based secret sharing, these contributions and the secrets being masked are bounded integers. Moreover, because extraction is linear, it can be performed directly on a secret-shared source.

**Matrix generator.** We begin by describing a *matrix generator*, which specifies a distribution over integer matrices with $\ell$ rows and $n$ columns. We also consider bounded matrix generators, where the magnitude of each entry of the generated matrix is bounded.

**Definition 3** (Matrix Generator)**.** *Let $\ell \in \mathbb{N}$ denote the number of rows and $n \in \mathbb{N}$ denote the number of columns. An $(\ell, n)$-*matrix generator *is a randomized algorithm* $\mathsf{MGen}$ *that outputs a matrix $\mathbf{M} \in \mathbb{Z}^{\ell \times n}$. For $B \in \mathbb{N}$, we say* $\mathsf{MGen}$ *is $B$-*bounded *if its support only consists of $B$-bounded matrices.*

**Block-fixing source.** A block-fixing source models a high-entropy source in which at most $t$ out of $n$ coordinates are fixed and the remaining coordinates are independent and uniformly random over an alphabet of size $B_{\mathsf{rand}}$. We use the term *corruption fraction* to refer to the maximum fraction of fixed coordinates $\tau = t/n$.

**Definition 4** (Block-Fixing Source)**.** *Let $n \in \mathbb{N}$ denote the source length, $B_{\mathsf{rand}} \in \mathbb{N}$ denote the source alphabet size, and $t \in \mathbb{Z}_{\geqslant 0}$ denote the threshold on the number of fixed coordinates, such that $t \leqslant n$. A distribution $\mathcal{S}$ over $[0, B_{\mathsf{rand}} - 1]^n$ is called an $(n, t, B_{\mathsf{rand}})$-*block-fixing source *if there exists a subset $\mathcal{I} \subseteq [n]$ with $|\mathcal{I}| \leqslant t$ such that for $\mathbf{s} \leftarrow \mathcal{S}$, the subvector $\mathbf{s}_{\mathcal{I}}$ is fixed and the subvector $\mathbf{s}_{\mathcal{H}}$ is uniformly distributed over $[0, B_{\mathsf{rand}} - 1]^{|\mathcal{H}|}$, where $\mathcal{H} = [n] \setminus \mathcal{I}$. The family of all such sources is denoted by* $\mathsf{BFS}(n, t, B_{\mathsf{rand}})$.

**Linear mask extractor.** Given an $n$-length block-fixing source over an alphabet of size $B_{\mathsf{rand}}$, a linear mask extractor can be used to mask $\ell$ secrets, each of magnitude at most $B_{\mathsf{sec}}$. As discussed earlier, by "masking" we mean that the statistical distance between the masked values for any two tuples of secrets is small. Formally, we capture this by requiring the distance between a masking of any such tuple and a masking of the all-zero vector to be small.

The linear mask extractor is additionally parameterized by $\delta_{\mathsf{sd}}$ and $\delta_{\mathsf{lme}}$, which capture two types of error. For a good sampled matrix, $\delta_{\mathsf{sd}}$ bounds the statistical distance between the shifted and unshifted distributions. The parameter $\delta_{\mathsf{lme}}$ bounds the probability of sampling a matrix that fails this guarantee for some admissible source and bounded shift. Thus, except with probability $\delta_{\mathsf{lme}}$, a single sampled matrix satisfies the guarantee simultaneously for every admissible source and shift, even if they are selected after the matrix is sampled.

**Definition 5** (Linear Mask Extractor)**.** *Let* MGen *be a* $(\ell, n)$*-matrix generator (Definition 3), let* $B_{\mathsf{rand}} \in \mathbb{N}$ *and* $0 \leqslant t \leqslant n$ *denote the parameters for the block-fixing source (Definition 4), let* $B_{\mathsf{sec}} \in \mathbb{Z}_{\geqslant 0}$ *denote the shift radius, let* $0 \leqslant \delta_{\mathsf{sd}} \leqslant 1$ *denote the masking error and let* $0 \leqslant \delta_{\mathsf{lme}} \leqslant 1$ *denote the failure probability.* MGen *is said to be a* $(t, B_{\mathsf{rand}}, B_{\mathsf{sec}}, \delta_{\mathsf{sd}}, \delta_{\mathsf{lme}})$-linear mask extractor *if*

$$\Pr\left[\begin{array}{c} \exists \mathcal{S} \in \mathsf{BFS}(n, t, B_{\mathsf{rand}}),\ \exists \mathbf{x} \in \mathbb{Z}^{\ell} \text{ such that } \|\mathbf{x}\|_{\infty} \leqslant B_{\mathsf{sec}} : \\ \mathsf{SD}(\mathbf{E} \cdot \mathbf{s} + \mathbf{x},\ \mathbf{E} \cdot \mathbf{s}) > \delta_{\mathsf{sd}} \end{array}\right] \leqslant \delta_{\mathsf{lme}},$$

*where the probability is over* $\mathbf{E} \leftarrow$ MGen *and the statistical distance is over* $\mathbf{s} \leftarrow \mathcal{S}$.

Observe that the number of columns $n$ of the linear mask extractor equals the source length and the number of rows $\ell$ corresponds to the number of extracted masks. We use *extraction rate* to refer to the ratio of the number of rows to the number of columns $\rho = \ell/n$.

## 4.2 Template for Constructing Linear Mask Extractors

We next show that matrix generators that satisfy a "co-prime submatrices" property are linear mask extractors. Informally, this property requires that the matrix obtained after removing any set of at most $t$ columns contains square submatrices whose determinants are collectively coprime.

In more detail, consider a $\ell \times n$ matrix $\mathbf{E}$ sampled by a matrix generator. Recall that we extract masks from a block-fixing source $\mathbf{s}$, with at most $t$ fixed coordinates, by computing $\mathbf{Es}$. Let $\mathcal{H}$ denote the indices of the uniformly random coordinates, and write $\mathbf{M} = \mathbf{E}_{*,\mathcal{H}}$. We require $\mathbf{M}$ to contain $\ell \times \ell$ submatrices whose determinants have greatest common divisor 1.

This property lets us combine the adjugates of these submatrices using Bézout's identity to obtain an integer right inverse: a matrix $\mathbf{Q}$ over the integers such that $\mathbf{MQ} = I_{\ell}$. Looking ahead, while the super-invertible matrices in [GJMW24] have $\ell = n - t$, our construction will have $\ell < n - t$. This leaves room for multiple square submatrices with $\ell$ rows, whose determinants can be collectively coprime.

The integer right inverse lets us bound the statistical distance between the masked values and the masks as

$$\mathsf{SD}(\mathbf{Es} + \mathbf{x}, \mathbf{Es}) = \mathsf{SD}(\mathbf{Ms}_{\mathcal{H}} + \mathbf{x}, \mathbf{Ms}_{\mathcal{H}}) = \mathsf{SD}(\mathbf{M}(\mathbf{s}_{\mathcal{H}} + \mathbf{Qx}), \mathbf{Ms}_{\mathcal{H}}) \leqslant \mathsf{SD}(\mathbf{s}_{\mathcal{H}} + \mathbf{Qx}, \mathbf{s}_{\mathcal{H}}).$$

The last distance is small by the smudging lemma (Lemma 4), provided the random coordinates are sampled from a sufficiently large range.

For comparison, the argument in [GJMW24] uses a square matrix $\mathbf{M}$ with non-zero determinant and yields

$$\mathsf{SD}(\mathbf{Ms}_{\mathcal{H}} + \mathbf{x}, \mathbf{Ms}_{\mathcal{H}}) \leqslant \mathsf{SD}\left(\mathbf{s}_{\mathcal{H}} + \mathrm{adj}(\mathbf{M})\frac{\mathbf{x}}{\det(\mathbf{M})}, \mathbf{s}_{\mathcal{H}}\right).$$

To apply integer smudging, they reformat the secrets to be divisible by $\det(\mathbf{M})$. In our case, $\mathbf{MQ} = I_{\ell}$, so expressing the statistical distance in terms of a shift on the source coordinates no longer introduces a $1/\det(\mathbf{M})$ factor, and the secrets need not be reformatted.

The theorem below uses bounds on the matrix entries and the Bézout coefficients to give an explicit bound on the masking error. The probability that the coprime submatrix property fails for some $\mathcal{H}$ corresponds to the linear mask extractor failure probability $\delta_{\mathsf{lme}}$.

**Theorem 7.** *Let* MGen *be a* $B$*-bounded* $(\ell, n)$*-matrix generator (Definition 3), let* $t \in \mathbb{Z}_{\geqslant 0}$ *satisfy* $t \leqslant n - \ell$*, let* $\Lambda \in \mathbb{N}$*, and let* $0 \leqslant \delta_{\mathsf{lme}} \leqslant 1$*. Suppose that for* $\mathbf{E} \leftarrow$ MGen *the following property holds, except with probability at most* $\delta_{\mathsf{lme}}$*:*

*for all $\mathcal{H} \subseteq [n]$ satisfying $|\mathcal{H}| \geqslant n - t$, there exist $k \in \mathbb{N}$, $\ell$-sized column sets $C_1, \ldots, C_k \subseteq \mathcal{H}$, and integers $\eta_1, \ldots, \eta_k$ such that*

$$\sum_{i=1}^{k} \eta_i \cdot \det\left(\mathbf{E}_{*,C_i}\right) = 1 \qquad \text{and} \qquad \sum_{i=1}^{k} |\eta_i| \leqslant \Lambda. \tag{2}$$

*Then, for all $B_{\mathsf{rand}} \in \mathbb{N}$ and $B_{\mathsf{sec}} \in \mathbb{Z}_{\geqslant 0}$, $\mathsf{MGen}$ is a $(t, B_{\mathsf{rand}}, B_{\mathsf{sec}}, \delta_{\mathsf{sd}}, \delta_{\mathsf{lme}})$-linear mask extractor (Definition 5), where*

$$\delta_{\mathsf{sd}} = \min\left\{1, \frac{\ell^2(\ell-1)!\; B^{\ell-1} B_{\mathsf{sec}}\, \Lambda}{B_{\mathsf{rand}}}\right\}.$$

*Proof.* Fix a matrix $\mathbf{E}$ satisfying the property in the theorem statement. Fix an arbitrary source $\mathcal{S} \in \mathsf{BFS}(n, t, B_{\mathsf{rand}})$ and an arbitrary vector $\mathbf{x} \in \mathbb{Z}^{\ell}$ such that $\|\mathbf{x}\|_{\infty} \leqslant B_{\mathsf{sec}}$. By Definition 5, it suffices to show that $\mathsf{SD}(\mathbf{Es} + \mathbf{x}, \mathbf{Es}) \leqslant \delta_{\mathsf{sd}}$, where $\mathbf{s} \leftarrow \mathcal{S}$.

Let $\mathcal{I} \subseteq [n]$ be the set of fixed coordinates of $\mathcal{S}$, and let $\mathcal{H} = [n] \setminus \mathcal{I}$. Setting $h \coloneqq |\mathcal{H}|$, we have $h \geqslant n - t$ and, for $\mathbf{s} \leftarrow \mathcal{S}$, the vector $\mathbf{s}_{\mathcal{H}}$ is uniformly distributed over $[0, B_{\mathsf{rand}} - 1]^h$. Let

$$\mathbf{M} \coloneqq \mathbf{E}_{*,\mathcal{H}}.$$

We next show that $\mathbf{M}$ admits an integer matrix $\mathbf{Q}$ as its right inverse. By the hypothesis, choose $k \in \mathbb{N}$, column sets $C_1, \ldots, C_k \subseteq \mathcal{H}$, and integers $\eta_1, \ldots, \eta_k$ satisfying Equation (2). For every $i \in [k]$, define $\mathbf{M}_i \coloneqq \mathbf{E}_{*,C_i}$ and let $\mathbf{P}_i \in \{0,1\}^{h \times \ell}$ be the column-selection matrix that selects from $\mathbf{M}$ the columns forming $\mathbf{M}_i$. In particular,

$$\mathbf{M}\mathbf{P}_i = \mathbf{M}_i.$$

We construct

$$\mathbf{Q} \coloneqq \sum_{i=1}^{k} \eta_i \cdot \mathbf{P}_i \cdot \mathrm{adj}(\mathbf{M}_i) \in \mathbb{Z}^{h \times \ell}.$$

Its entries are integers because each $\eta_i$ is an integer and each $\mathbf{P}_i$ and $\mathrm{adj}(\mathbf{M}_i)$ has integer entries. Moreover,

$$\begin{aligned}
\mathbf{MQ} &= \sum_{i=1}^{k} \eta_i \cdot \mathbf{M}\mathbf{P}_i\, \mathrm{adj}(\mathbf{M}_i) && \triangleright \text{ Definition of } \mathbf{Q} \\
&= \sum_{i=1}^{k} \eta_i \cdot \mathbf{M}_i\, \mathrm{adj}(\mathbf{M}_i) && \triangleright \text{ Definition of } \mathbf{P}_i \\
&= \left(\sum_{i=1}^{k} \eta_i \cdot \det(\mathbf{M}_i)\right) I_{\ell} && \triangleright \text{ Adjugate identity} \\
&= I_{\ell}. && \triangleright \text{ Equation (2)}
\end{aligned}$$

We next use $\mathbf{Q}$ to transfer the output shift to the honest source coordinates. In more detail, we have

$$\begin{aligned}
\mathsf{SD}(\mathbf{Es} + \mathbf{x}, \mathbf{Es}) &= \mathsf{SD}(\mathbf{M}\mathbf{s}_{\mathcal{H}} + \mathbf{x}, \mathbf{M}\mathbf{s}_{\mathcal{H}}) && \triangleright \text{ Translation invariance} \\
&= \mathsf{SD}(\mathbf{M}\,(\mathbf{s}_{\mathcal{H}} + \mathbf{Qx}), \mathbf{M}\mathbf{s}_{\mathcal{H}}) && \triangleright\ \mathbf{MQ} = I_{\ell} \\
&\leqslant \mathsf{SD}(\mathbf{s}_{\mathcal{H}} + \mathbf{Qx}, \mathbf{s}_{\mathcal{H}})\,. && \triangleright \text{ Data processing}
\end{aligned}$$

It therefore remains to bound the effect of the input shift $\mathbf{Qx}$. Since $\mathbf{Qx} \in \mathbb{Z}^h$ and the $h$ honest coordinates are independent and uniform over $[0, B_{\mathsf{rand}} - 1]$, applying the Smudging Lemma (Lemma 4) coordinate-wise and summing the resulting bounds gives

$$\mathsf{SD}(\mathbf{s}_{\mathcal{H}} + \mathbf{Qx}, \mathbf{s}_{\mathcal{H}}) \leqslant \min\left\{1, \frac{\|\mathbf{Qx}\|_1}{B_{\mathsf{rand}}}\right\}.$$

We are left to bound $\|\mathbf{Qx}\|_1$. We first bound each $\|\mathbf{P}_i \operatorname{adj}(\mathbf{M}_i)\mathbf{x}\|_1$. Since $\mathbf{P}_i$ selects distinct columns, each column of $\mathbf{P}_i$ contains exactly one 1, and each row contains at most one 1. Thus, left multiplication by $\mathbf{P}_i$ places each input coordinate in a distinct output coordinate and fills the remaining coordinates with zeros. Consequently, $\mathbf{P}_i$ preserves the $\ell_1$-norm and, for each $i \in [k]$, we have

$$\begin{aligned}\|\mathbf{P}_i \operatorname{adj}(\mathbf{M}_i)\mathbf{x}\|_1 &= \|\operatorname{adj}(\mathbf{M}_i)\mathbf{x}\|_1 \\ &\leqslant \ell^2 \cdot \|\operatorname{adj}(\mathbf{M}_i)\|_\infty \cdot \|\mathbf{x}\|_\infty \\ &\leqslant \ell^2(\ell-1)!\, B^{\ell-1} B_{\mathsf{sec}}. \qquad \triangleright \text{ Fact 3}\end{aligned}$$

This implies that

$$\|\mathbf{Qx}\|_1 \quad \leqslant \quad \sum_{i=1}^{k} |\eta_i| \cdot \|\mathbf{P}_i \operatorname{adj}(\mathbf{M}_i)\mathbf{x}\|_1 \quad \leqslant \quad \ell^2(\ell-1)!\, B^{\ell-1} B_{\mathsf{sec}}\, \Lambda.$$

Substituting this into the statistical distance bound yields

$$\mathrm{SD}(\mathbf{Es}+\mathbf{x}, \mathbf{Es}) \leqslant \min\left\{1, \frac{\ell^2(\ell-1)!\, B^{\ell-1} B_{\mathsf{sec}} \Lambda}{B_{\mathsf{rand}}}\right\} = \delta_{\mathsf{sd}}.$$

Since $\mathcal{S}$ and $\mathbf{x}$ were arbitrary, every matrix $\mathbf{E}$ satisfying the theorem's property meets the statistical-distance requirement in Definition 5. Such a matrix is sampled with probability at least $1 - \delta_{\mathsf{lme}}$, which completes the proof. ■

# 5 Toeplitz Linear Mask Extractor

In this section, we instantiate the template from Section 4.2 using Toeplitz matrices.

**Toeplitz matrix generator.** We describe the Toeplitz matrix generator in Figure 4. It is parameterized by the number of rows $\ell$ and the bound $B$, and takes the number of columns $n$ as input. For each diagonal, it independently samples a value uniformly from $[0, B-1]$ and places that value in every entry along the diagonal. Our indexing assigns the entry in row $a$ and column $j$ to $z_{j-a}$. Thus, $z_0$ fills the main diagonal, while positive indices refer to diagonals above it and negative indices to diagonals below it.

$\mathsf{GenToeplitz}_{\ell,B}(1^n)$

1 : $(z_{1-\ell}, \ldots, z_{n-1}) \leftarrow [0, B-1]^{\ell+n-1}$
2 : **for** $a \in [\ell]$
3 : **for** $j \in [n]$
4 : $\mathbf{E}_{a,j} \coloneqq z_{j-a}$
5 : **return** $\mathbf{E}$

Figure 4: Toeplitz Matrix Generator

We prove that the Toeplitz matrix generator is a linear mask extractor in Section 5.1, discuss the efficiency of extraction in Section 5.2, and conclude with concrete parameters in Section 5.3.

## 5.1 Proof

We prove that the Toeplitz matrix generator is a linear mask extractor by showing that it satisfies the conditions of Theorem 7. We begin in Section 5.1.1 by bounding the probability that a fixed submatrix fails to have full row rank modulo a prime. For small primes, this bound is too weak to handle all choices of submatrices at once, so we give a separate bound in Section 5.1.2. We then establish the required coprime submatrix property in Section 5.1.3 and apply the template to obtain the Toeplitz linear mask extractor in Section 5.1.4.

### 5.1.1 Full Rank Modulo Primes

Consider the $\ell \times n$ Toeplitz matrix $\mathbf{E}$ sampled using $\mathsf{GenToeplitz}_{\ell,B}(1^n)$. We bound the probability that, for a fixed prime $p$ and column set $J \subseteq [n]$ with $|J| \geqslant \ell$, the submatrix $\mathbf{E}_{*,J}$ fails to have full row rank over $\mathbb{Z}_p$. Note that the probability is for a fixed column set $J$, which will correspond to the uniformly random source coordinates when $\mathbf{E}$ is used as a linear mask extractor. To handle an adaptive choice of $J$, we will later apply a union bound over each possible column set.

As discussed in Section 2, the proof considers the columns one by one, from left to right, and lower-bounds the probability that adding a column increases the rank. In more detail, let $\mathbf{C}^{(k)}$ denote the submatrix consisting of the first $k$ columns indexed by $J$, and let $\mathcal{N}_k$ be the left nullspace of $\mathbf{C}^{(k)}$. By definition, a vector $\mathbf{u} \in \mathcal{N}_k$ satisfies $\mathbf{u}^\top \mathbf{C}^{(k)} = \mathbf{0}$. Now consider appending the $(k+1)$-th column in $J$, denoted by $\mathbf{c}^{(k+1)}$, to $\mathbf{C}^{(k)}$. Importantly, the first entry of this column hasn't appeared in any of the previous columns and is thus uniformly random in $[0, B-1]$ conditioned on $\mathbf{C}^{(k)}$ and even the remaining entries of this column. Assuming the first entry of $\mathbf{u}$ is non-zero, this column satisfies $\mathbf{u}^\top \mathbf{c}^{(k+1)} \neq 0$ with probability at least $1 - \alpha_p$, where $\alpha_p = \lceil B/p \rceil / B$. When this occurs, the new column eliminates $\mathbf{u}$ from the nullspace and consequently increases the rank by 1.

The final bound comes from computing an upper bound on the probability that the rank is less than $\ell$ after processing all the columns indexed by $J$. Specifically, let $Y_{k+1}$ denote the random variable that is 1 when $\mathbf{c}^{(k+1)}$ increases the rank, and is 0 otherwise. We would like to bound the probability that $\sum_{j=1}^{|J|} Y_j < \ell$. However, computing this bound directly is not straightforward since the rank increase events, and hence $\{Y_j\}_j$, might be correlated. To obtain an explicit bound, we use a decoupling argument that lets us replace each $Y_j$ with an independent Bernoulli random variable for the purpose of upper-bounding the failure probability. This allows us to express the final bound in terms of the tail of a binomial distribution.

So far, we have assumed that the first entry of the left-null vector is non-zero, since we rely on the randomness of the first entry of the column. To handle the other cases, we repeat the argument for each possible position of the first non-zero coordinate. In particular, if this position is $i$, dropping the first $i-1$ coordinates gives a left-null vector, with a non-zero first entry, for the submatrix $\mathbf{E}_{[i,\ell],J}$. When processing columns of this submatrix one by one, the first entry is again uniformly random conditioned on the remaining entries of the column and the previous columns of the submatrix. This allows us to make essentially the same argument for each $i \in [\ell]$ and then take a union bound over all indices.

We actually prove a slightly stronger statement below: we first fix the columns indexed by a set $I$ and then bound the probability that $\mathbf{E}_{*,J}$ fails to have full row rank over $\mathbb{Z}_p$, where every column in $J$ is to the right of every column in $I$. Looking ahead, this will be useful for identifying square submatrices with collectively coprime determinants. For each prime factor $p$ of the determinant of a square submatrix indexed by $I$, we will look for a square submatrix among the columns indexed by $J$ whose determinant is not divisible by $p$.

Importantly, when proving this stronger statement, we have to be careful when relying on the randomness of the first entry of a column since we condition on the columns indexed by $I$. In particular, this may also fix the first entries of the first $i-1$ columns of $\mathbf{E}_{[i,\ell],J}$. However, this does not pose an issue: we can skip the first $i-1$ columns of $J$, after which the first entry of each new column is uniformly random even under the conditioning. We lose $i-1$ columns, but the submatrix also has $i-1$ fewer rows, and we only need the rank to be $\ell - i + 1$, so we can recover the same bound as before.

**Lemma 8.** *Let $n, \ell, B \in \mathbb{N}$, let $p$ be a prime, and let $I, J \subseteq [n]$ satisfy $|J| = s \geqslant \ell$, with every index in $I$ smaller than every index in $J$. For $\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)$ (Figure 4), let $\mathbf{A}$ be any matrix satisfying $\Pr[\mathbf{E}_{*,I} = \mathbf{A}] > 0$. Then*

$$\Pr\left[\mathrm{rank}_{\mathbb{Z}_p}(\mathbf{E}_{*,J}) < \ell \;\middle|\; \mathbf{E}_{*,I} = \mathbf{A}\right] \leqslant \ell \cdot \Pr\left[\mathrm{Bin}(s, 1-\alpha_p) \leqslant \ell - 1\right],$$

*where $\alpha_p \coloneqq \lceil B/p \rceil / B$ and $\mathrm{Bin}(s, 1-\alpha_p)$ denotes a binomial random variable with $s$ trials and success probability $1-\alpha_p$.*

*Proof.* Let $J = \{j_1, \ldots, j_s\}$, where $j_1 < \cdots < j_s$. Let $z_{1-\ell}, \ldots, z_{n-1}$ be the entries sampled to generate $\mathbf{E}$, so that $\mathbf{E}_{a,j} = z_{j-a}$ for every $a \in [\ell]$ and $j \in [n]$. Fix a matrix $\mathbf{A}$ satisfying $\Pr[\mathbf{E}_{*,I} = \mathbf{A}] > 0$, and condition on $\mathbf{E}_{*,I} = \mathbf{A}$ for the remainder of the proof. If $I$ is empty, then $\mathbf{A}$ is the empty matrix and this conditioning is vacuous. The conditioning fixes those entries in $z_{1-\ell}, \ldots, z_{n-1}$ that appear in the columns indexed by $I$. All other entries remain mutually independent and uniformly distributed over $[0, B-1]$.

For every $i \in [\ell]$, let $\mathrm{Bad}_i$ be the event that the left nullspace of $\mathbf{E}_{[i,\ell],J}$ over $\mathbb{Z}_p$ contains a vector whose first coordinate is non-zero. The following claim states that $\mathbf{E}_{*,J}$ is rank deficient exactly when $\mathrm{Bad}_i$ occurs for some $i \in [\ell]$.

**Claim 8.1.** *We have*

$$\left\{\mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{E}_{*,J}\right) < \ell\right\} = \bigcup_{i=1}^{\ell} \mathrm{Bad}_i.$$

*Proof of the claim.* By definition, $\mathbf{E}_{*,J}$ is rank deficient if there exists a non-zero vector in its left nullspace. Such a vector causes $\mathrm{Bad}_i$ to occur when $i$ is the index of its first non-zero coordinate. Conversely, if $\mathrm{Bad}_i$ occurs, prepending $i-1$ zeros to the corresponding vector gives a non-zero left-null vector for $\mathbf{E}_{*,J}$. □

Therefore, it suffices to bound the probability that each $\mathrm{Bad}_i$ occurs. Fix $i \in [\ell]$ and let $d \coloneqq \ell - i + 1$. For every $k \in [s]$, let

$$\mathbf{c}^{(k)} \coloneqq \begin{pmatrix} z_{j_k - i} \\ z_{j_k - i - 1} \\ \vdots \\ z_{j_k - \ell} \end{pmatrix} \in \mathbb{Z}_p^d$$

be the column of $\mathbf{E}_{[i,\ell],J}$ corresponding to $j_k$.[20] Let $\mathbf{C}^{(0)}$ be the empty matrix and, for $k \in [s]$, let

$$\mathbf{C}^{(k)} \coloneqq \left[\mathbf{c}^{(1)} \mid \cdots \mid \mathbf{c}^{(k)}\right]$$

be the first $k$ columns of $\mathbf{E}_{[i,\ell],J}$. By definition, $\mathrm{Bad}_i$ occurs when $\mathbf{C}^{(s)}$ has a left-null vector whose first coordinate is non-zero.

We bound the probability that the addition of each new column $\mathbf{c}^{(k)}$ increases the rank. For every $i \leqslant k \leqslant s$, define

$$Y_k \coloneqq \begin{cases} 1, & \text{if } \mathbf{C}^{(k)} \text{ has no left-null vector} \\ & \text{whose first coordinate is non-zero,} \\ \mathbb{1}\left[\mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{C}^{(k)}\right) > \mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{C}^{(k-1)}\right)\right], & \text{otherwise.} \end{cases}$$

The following claim bounds the probability that $Y_k = 1$ for each $k \geqslant i$. Note that the claim need not extend to $k < i$ because it relies on the first entry of $\mathbf{c}^{(k)}$ being random, whereas conditioning on $\mathbf{E}_{*,I}$ may fix this entry. However, looking ahead, this does not pose an issue to bound $\mathrm{Bad}_i$.

**Claim 8.2.** *For every $k \in [i,s]$, we have*

$$\Pr\left[Y_k = 1 \,\middle|\, \mathbf{C}^{(k-1)}\right] \geqslant 1 - \alpha_p,$$

*where $\alpha_p$ is defined in the lemma statement.*

*Proof of the claim.* Fix $k \in [i,s]$ and let

$$\mathcal{N}_{k-1} \coloneqq \left\{\mathbf{u} \in \mathbb{Z}_p^d : \mathbf{u}^\top \mathbf{C}^{(k-1)} = 0 \text{ and } \mathbf{u}_1 \neq 0\right\}.$$

If $\mathcal{N}_{k-1}$ is empty, then $\mathbf{C}^{(k)}$ also has no left-null vector whose first coordinate is non-zero, since every left-null vector for $\mathbf{C}^{(k)}$ is also a left-null vector for $\mathbf{C}^{(k-1)}$. Thus, $Y_k = 1$ and the claim follows.

Suppose that $\mathcal{N}_{k-1}$ is nonempty. Fix an arbitrary ordering of $\mathbb{Z}_p^d$, and let $\mathbf{u}^{(k)}$ be the first vector in $\mathcal{N}_{k-1}$ under this ordering. In particular, $\mathbf{u}^{(k)}$ is determined by the preceding columns $\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(k-1)}$.

We next bound the probability that the $k$-th column increases the rank. Every entry among $z_{1-\ell}, \ldots, z_{n-1}$ that appears in the columns indexed by $I$ has index at most $\max(I) - 1$. Since $\max(I) < j_1$ and the column indices $j_1, \ldots, j_s$ are strictly increasing, we have $j_k \geqslant \max(I) + k$. Thus, for every $k \geqslant i$, the leading entry of $\mathbf{c}^{(k)}$ satisfies $j_k - i \geqslant \max(I)$ and has not appeared in the columns indexed by $I$. This also holds when $I$ is empty, since no

[20] The vectors $\mathbf{c}^{(k)}$ and matrices $\mathbf{C}^{(k)}$ depend on the fixed index $i$; we suppress this dependence from the notation.

entries are fixed by the conditioning. An entry from $\mathbf{c}^{(r)}$ for $r < k$ has index at most $j_r - i < j_k - i$, so $z_{j_k - i}$ has also not appeared in $\mathbf{C}^{(k-1)}$. Moreover, every other entry of $\mathbf{c}^{(k)}$ has a smaller index. Consequently, $z_{j_k - i}$ remains independent and uniform after conditioning on $\mathbf{C}^{(k-1)}$ and on all other entries of $\mathbf{c}^{(k)}$.

Next observe that the equation $\left(\mathbf{u}^{(k)}\right)^\top \mathbf{c}^{(k)} = 0$ fixes $z_{j_k - i}$ modulo $p$, since $\left(\mathbf{u}^{(k)}\right)_1 \neq 0$ and is thus invertible in $\mathbb{Z}_p$. Since $z_{j_k - i}$ is uniformly random in $[0, B-1]$, we have

$$\Pr\left[\left(\mathbf{u}^{(k)}\right)^\top \mathbf{c}^{(k)} \neq 0 \;\middle|\; \mathbf{C}^{(k-1)}\right] \geqslant 1 - \alpha_p.$$

Whenever $\left(\mathbf{u}^{(k)}\right)^\top \mathbf{c}^{(k)} \neq 0$, the rank of $\mathbf{C}^{(k)}$ is one more than the rank of $\mathbf{C}^{(k-1)}$. Therefore, $Y_k = 1$ with probability at least $1 - \alpha_p$, which proves the claim. □

The following claim replaces the possibly correlated indicators $Y_i, \ldots, Y_s$ with independent Bernoulli variables.

**Claim 8.3.** *There exist mutually independent Bernoulli random variables $X_i, \ldots, X_s$, each with mean $1 - \alpha_p$, such that $X_k \leqslant Y_k$ for every $k \in [i, s]$.*

*Proof of the claim.* If $\alpha_p = 1$, take $X_k = 0$ for every $i \leqslant k \leqslant s$. Otherwise, let

$$\gamma_k \coloneqq \Pr\left[Y_k = 1 \;\middle|\; \mathbf{C}^{(k-1)}\right],$$

so that $\gamma_k \geqslant 1 - \alpha_p > 0$ by Claim 8.2. Sample mutually independent auxiliary random variables $U_i, \ldots, U_s$ uniformly over the real unit interval and independently of the Toeplitz matrix, and set

$$X_k \coloneqq Y_k \cdot \mathbb{1}\left[U_k \leqslant \frac{1 - \alpha_p}{\gamma_k}\right].$$

Then $X_k \leqslant Y_k$. Condition on the preceding columns $\mathbf{c}^{(1)}, \ldots, \mathbf{c}^{(k-1)}$ and on all auxiliary values sampled before step $k$. Since the auxiliary variables are independent of the Toeplitz matrix, $Y_k$ still equals 1 with probability $\gamma_k$. Therefore,

$$\Pr\left[X_k = 1 \;\middle|\; \mathbf{C}^{(k-1)}, U_i, \ldots, U_{k-1}\right] = \gamma_k \cdot \frac{1 - \alpha_p}{\gamma_k} = 1 - \alpha_p.$$

This yields the same probability $1 - \alpha_p$ for every possible value of the preceding columns and auxiliary variables. Since $X_k$ is binary, it follows that $X_k$ is independent of $\left(\mathbf{C}^{(k-1)}, U_i, \ldots, U_{k-1}\right)$, and we have $\Pr[X_k = 1] = 1 - \alpha_p$. Thus, $X_k$ is independent of $(X_i, \ldots, X_{k-1})$ since the latter are determined by $\left(\mathbf{C}^{(k-1)}, U_i, \ldots, U_{k-1}\right)$. Induction on $k$ then shows that $X_i, \ldots, X_s$ are mutually independent Bernoulli random variables with mean $1 - \alpha_p$, as required. □

We next use the above claims to bound the probability that $\mathrm{Bad}_i$ occurs.

**Claim 8.4.** *We have*

$$\Pr\left[\mathrm{Bad}_i \;\middle|\; \mathbf{E}_{*,I} = \mathbf{A}\right] \leqslant \Pr\left[\mathrm{Bin}(s, 1 - \alpha_p) \leqslant \ell - 1\right].$$

*Proof of the claim.* When $\mathrm{Bad}_i$ occurs, a vector witnessing $\mathrm{Bad}_i$ is a left-null vector for $\mathbf{C}^{(s)}$ and hence for every $\mathbf{C}^{(k)}$. Thus, the first condition in the definition of $Y_k$ does not occur, and every index $i \leqslant k \leqslant s$ for which $Y_k = 1$ contributes to an increase in the rank. Moreover, $\mathbf{C}^{(s)}$ has rank at most $d - 1$ since it has a non-zero left-null vector, where $d = \ell - i + 1$ denotes the number of rows of $\mathbf{C}^{(s)}$.

Fix random variables $X_i, \ldots, X_s$ as in Claim 8.3. It follows that when $\mathrm{Bad}_i$ occurs, we have

$$\sum_{k=i}^{s} X_k \leqslant \sum_{k=i}^{s} Y_k \leqslant \mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{C}^{(s)}\right) \leqslant d - 1 = \ell - i.$$

Therefore,

$$\mathsf{Bad}_i \subseteq \left\{ \sum_{k=i}^{s} X_k \leqslant \ell - i \right\},$$

which implies that

$$\begin{aligned} \Pr\left[\mathsf{Bad}_i \;\middle|\; \mathbf{E}_{*,I} = \mathbf{A}\right] &\leqslant \Pr\left[\mathrm{Bin}(s - i + 1, 1 - \alpha_p) \leqslant \ell - i\right] \\ &\leqslant \Pr\left[\mathrm{Bin}(s, 1 - \alpha_p) \leqslant \ell - 1\right]. \end{aligned}$$

which proves the claim. Here, the second inequality follows from the fact that adding $i - 1$ additional Bernoulli trials contributes at most $i - 1$ successes. □

Finally, combining Claims 8.1 and 8.4 with a union bound gives

$$\begin{aligned} \Pr\left[\mathrm{rank}_{\mathbb{Z}_p}(\mathbf{E}_{*,J}) < \ell \;\middle|\; \mathbf{E}_{*,I} = \mathbf{A}\right] &\leqslant \sum_{i=1}^{\ell} \Pr\left[\mathsf{Bad}_i \;\middle|\; \mathbf{E}_{*,I} = \mathbf{A}\right] \\ &\leqslant \ell \cdot \Pr\left[\mathrm{Bin}(s, 1 - \alpha_p) \leqslant \ell - 1\right]. \end{aligned}$$

This proves the lemma. ■

### 5.1.2 Full Rank Modulo Small Primes

Lemma 8 provides an upper bound for the probability that, for a fixed $J \subseteq [n]$, the submatrix of $\mathbf{E}$ consisting of columns indexed by $J$ fails to have full row rank over $\mathbb{Z}_p$, for some fixed prime $p$. To allow for an adaptive choice of $J$, a natural approach is to apply a union bound over all possible column sets. However, for small primes $p$, the bound from Lemma 8 can be too weak to absorb this union bound. For example, when $p = 2$, the bound for a fixed set of $s$ columns is at least $2^{-s}$, which can be outweighed by the $\binom{n}{s}$ possible choices of $J$.

The following lemma provides an upper bound on the probability that there exists $J$ satisfying $|J| \geqslant s$ such that $\mathbf{E}_{*,J}$ does not have full row rank over $\mathbb{Z}_p$. As discussed in Section 2, the proof fixes a non-zero vector $\mathbf{u} \in \mathbb{Z}_p^{\ell}$ and then upper-bounds the probability that $s$ or more columns of $\mathbf{E}$ are orthogonal to $\mathbf{u}$. If this occurs, these columns define a submatrix that does not have full row rank. Crucially, counting orthogonal columns across the whole matrix accounts for all possible choices of $J$ at once, even if $J$ is chosen after seeing the matrix.

To bound the probability that a fixed vector $\mathbf{u}$ is orthogonal to a particular column of $\mathbf{E}$, we use a similar approach as in Lemma 8. Specifically, let $i$ be the index of the first non-zero coordinate of $\mathbf{u}$. Since the preceding coordinates are zero, we can restrict attention to rows $i, \ldots, \ell$. As before, the first entry of each column in this restricted matrix is uniformly random conditioned on the previous columns and the remaining entries of the current column. Since the corresponding coordinate of $\mathbf{u}$ is non-zero, the column is orthogonal to $\mathbf{u}$ over $\mathbb{Z}_p$ with probability at most $\alpha_p = \lceil B/p \rceil / B$.

Let $Z_j$ denote the random variable that is 1 when the $j$-th column is orthogonal to $\mathbf{u}$ and is 0 otherwise. We would like to bound the probability that $\sum_{j=1}^{n} Z_j \geqslant s$. As before, these variables might be correlated, so we use a decoupling argument that lets us replace each $Z_j$ with an independent Bernoulli random variable. This allows us to express the bound in terms of the upper tail of a binomial distribution.

**Lemma 9.** *Let $n, \ell, B, s \in \mathbb{N}$ satisfy $\ell \leqslant s \leqslant n$, and let $p$ be a prime. Then*

$$\Pr_{\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)} \left[\exists J \subseteq [n],\ |J| \geqslant s : \mathrm{rank}_{\mathbb{Z}_p}(\mathbf{E}_{*,J}) < \ell\right] \leqslant \frac{p^{\ell} - 1}{p - 1} \cdot \Pr\left[\mathrm{Bin}(n, \alpha_p) \geqslant s\right],$$

*where $\alpha_p \coloneqq \lceil B/p \rceil / B$ and $\mathrm{Bin}(n, \alpha_p)$ denotes a binomial random variable with $n$ trials and success probability $\alpha_p$ and $\mathsf{GenToeplitz}_{\ell,B}(1^n)$ is described in Figure 4.*

*Proof.* Let $z_{1-\ell}, \dots, z_{n-1}$ be the entries sampled to generate $\mathbf{E}$, so that $\mathbf{E}_{a,j} = z_{j-a}$ for every $a \in [\ell]$ and $j \in [n]$. For every non-zero vector $\mathbf{u} \in \mathbb{Z}_p^\ell$, define the number of columns orthogonal to $\mathbf{u}$ by

$$N(\mathbf{u}) \coloneqq \sum_{j=1}^{n} \mathbb{1}\left[\mathbf{u}^\top \mathbf{E}_{*,j} = 0\right].$$

The following claim expresses the failure event in terms of these counts.

**Claim 9.1.** *We have*

$$\left\{\exists J \subseteq [n],\ |J| \geqslant s : \operatorname{rank}_{\mathbb{Z}_p}\left(\mathbf{E}_{*,J}\right) < \ell\right\} = \bigcup_{\mathbf{u} \in \mathbb{Z}_p^\ell \setminus \{\mathbf{0}\}} \{N(\mathbf{u}) \geqslant s\}.$$

*Proof of the claim.* If $\mathbf{E}_{*,J}$ is rank deficient for some $J$ with $|J| \geqslant s$, then a non-zero left-null vector for this matrix is orthogonal to at least $s$ columns of $\mathbf{E}$. Conversely, if $N(\mathbf{u}) \geqslant s$ for some non-zero $\mathbf{u}$, choosing any $s$ columns orthogonal to $\mathbf{u}$ gives a rank-deficient submatrix. □

Fix a non-zero vector $\mathbf{u} \in \mathbb{Z}_p^\ell$, and let $i$ be the index of its first non-zero coordinate. Set $d \coloneqq \ell - i + 1$ and $\mathbf{w} \coloneqq \mathbf{u}_{[i,\ell]}$, so that $\mathbf{w} \in \mathbb{Z}_p^d$ and $\mathbf{w}_1 \neq 0$. For every $j \in [n]$, let

$$\mathbf{c}^{(j)} \coloneqq \begin{pmatrix} z_{j-i} \\ z_{j-i-1} \\ \vdots \\ z_{j-\ell} \end{pmatrix} \in \mathbb{Z}_p^d$$

be the $j$-th column of $\mathbf{E}$ restricted to rows $i, \dots, \ell$. Since the first $i-1$ coordinates of $\mathbf{u}$ are zero, we have

$$\mathbf{u}^\top \mathbf{E}_{*,j} = \mathbf{w}^\top \mathbf{c}^{(j)}.$$

Let $\mathbf{C}^{(0)}$ be the empty matrix and, for every $j \in [n]$, let $\mathbf{C}^{(j)} \coloneqq \left[\mathbf{c}^{(1)} \mid \cdots \mid \mathbf{c}^{(j)}\right]$. Define

$$Z_j \coloneqq \mathbb{1}\left[\mathbf{w}^\top \mathbf{c}^{(j)} = 0\right],$$

so that $N(\mathbf{u}) = \sum_{j=1}^{n} Z_j$. We next bound the probability that $Z_j = 1$ conditioned on $\mathbf{C}^{(j-1)}$.

**Claim 9.2.** *For every $j \in [n]$, we have*

$$\Pr\left[Z_j = 1 \,\middle|\, \mathbf{C}^{(j-1)}\right] \leqslant \alpha_p.$$

*Proof of the claim.* Fix $j \in [n]$. The leading entry $z_{j-i}$ of $\mathbf{c}^{(j)}$ has not appeared in $\mathbf{C}^{(j-1)}$. Moreover, since the sampled entries are mutually independent, $z_{j-i}$ remains independent after conditioning on $\mathbf{C}^{(j-1)}$ and on all other entries of $\mathbf{c}^{(j)}$. Since $\mathbf{w}_1 \neq 0$, the equation

$$\mathbf{w}^\top \mathbf{c}^{(j)} = 0$$

fixes $z_{j-i}$ modulo $p$. Since $z_{j-i}$ is sampled uniformly at random from $[0, B-1]$, the probability that $Z_j = 1$, conditioned on $\mathbf{C}^{(j-1)}$ and the other entries of $\mathbf{c}^{(j)}$, is at most $\alpha_p$. □

The following claim replaces the possibly correlated indicators $Z_1, \dots, Z_n$ with independent Bernoulli variables.

**Claim 9.3.** *There exist mutually independent Bernoulli random variables $X_1, \dots, X_n$, each with mean $\alpha_p$, such that $Z_j \leqslant X_j$ for every $j \in [n]$.*

*Proof of the claim.* If $\alpha_p = 1$, take $X_j = 1$ for every $j$. Otherwise, let

$$\gamma_j \coloneqq \Pr\left[Z_j = 1 \,\middle|\, \mathbf{C}^{(j-1)}\right],$$

so that $0 \leqslant \gamma_j \leqslant \alpha_p < 1$ by Claim 9.2. Sample mutually independent auxiliary random variables $U_1, \dots, U_n$

uniformly over the real unit interval and independently of the Toeplitz matrix, and set

$$X_j \coloneqq Z_j + (1 - Z_j) \cdot \mathbb{1}\left[U_j \leqslant \frac{\alpha_p - \gamma_j}{1 - \gamma_j}\right].$$

Then $Z_j \leqslant X_j$. Condition on $\mathbf{C}^{(j-1)}$ and on $U_1, \ldots, U_{j-1}$. Since the auxiliary variables are independent of the Toeplitz matrix, $Z_j$ still equals 1 with probability $\gamma_j$. Therefore,

$$\begin{aligned}\Pr\left[X_j = 1 \,\middle|\, \mathbf{C}^{(j-1)}, U_1, \ldots, U_{j-1}\right] &= \gamma_j + (1 - \gamma_j)\frac{\alpha_p - \gamma_j}{1 - \gamma_j} \\ &= \alpha_p.\end{aligned}$$

Importantly, this yields the same probability $\alpha_p$ for every possible value of the preceding columns and auxiliary variables. Since $X_j$ is binary, it follows that $X_j$ is independent of $\left(\mathbf{C}^{(j-1)}, U_1, \ldots, U_{j-1}\right)$, and we have $\Pr\left[X_j = 1\right] = \alpha_p$. It follows that $X_j$ is independent of $(X_1, \ldots, X_{j-1})$ since the latter are determined by $\left(\mathbf{C}^{(j-1)}, U_1, \ldots, U_{j-1}\right)$. Induction on $j$ then shows that $X_1, \ldots, X_n$ are mutually independent Bernoulli random variables with mean $\alpha_p$, which proves the claim. □

Fix random variables $X_1, \ldots, X_n$ as in Claim 9.3. Since $Z_j \leqslant X_j$ for every $j$, we have

$$\{N(\mathbf{u}) \geqslant s\} = \left\{\sum_{j=1}^{n} Z_j \geqslant s\right\} \subseteq \left\{\sum_{j=1}^{n} X_j \geqslant s\right\}.$$

Consequently,

$$\Pr[N(\mathbf{u}) \geqslant s] \leqslant \Pr\left[\mathrm{Bin}(n, \alpha_p) \geqslant s\right].$$

Finally, if $\mathbf{u}'$ is a non-zero scalar multiple of $\mathbf{u}$, then $N(\mathbf{u}') = N(\mathbf{u})$. Every equivalence class of non-zero vectors under multiplication by a non-zero scalar in $\mathbb{Z}_p$ contains $p - 1$ vectors. Hence, the union in Claim 9.1 contains at most $(p^\ell - 1)/(p - 1)$ distinct events. Combining this observation with the preceding bound and applying a union bound gives

$$\Pr_{\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)}\left[\exists J \subseteq [n],\ |J| \geqslant s : \mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{E}_{*,J}\right) < \ell\right] \leqslant \frac{p^\ell - 1}{p - 1} \cdot \Pr\left[\mathrm{Bin}(n, \alpha_p) \geqslant s\right],$$

which proves the lemma. ■

### 5.1.3 Coprime Submatrices

We now compute an upper bound on the probability that the $\ell \times n$ matrix $\mathbf{E}$ generated using $\mathsf{GenToeplitz}_{\ell,B}$ does not satisfy the requirements of Theorem 7. We first bound the probability that there exists a set $\mathcal{H}$ of $n - t$ columns, corresponding to uniformly random source coordinates, for which $\mathbf{E}_{*,\mathcal{H}}$ does not contain square submatrices whose determinants are jointly coprime.

To do this, we use Lemmas 8 and 9, which bound the probability that a submatrix of $\mathbf{E}$ does not have full row rank modulo a prime $p$. In particular, observe that if a matrix with $\ell$ rows has full row rank over $\mathbb{Z}_p$, then it contains $\ell$ linearly independent columns and hence an invertible $\ell \times \ell$ square submatrix. This means that the determinant of this square submatrix is non-zero modulo $p$, which in turn implies that its determinant computed over the integers is not divisible by $p$.

Using this observation, we identify square submatrices in $\mathbf{E}_{*,\mathcal{H}}$ for a fixed $\mathcal{H}$ as follows. We first consider the left half of $\mathbf{E}_{*,\mathcal{H}}$ and use Lemma 8 to upper-bound the probability that it does not have full row rank over $\mathbb{Z}_p$ for some fixed prime $p$. If the left half does have full row rank modulo $p$, then it contains a square submatrix whose determinant is not divisible by $p$. We call this the *anchor submatrix*. Let $p_1, \ldots, p_d$ denote the distinct prime factors of its determinant. For each prime factor $p_i$, we use Lemma 8 to upper-bound the probability that the right half of $\mathbf{E}_{*,\mathcal{H}}$ fails to have full row rank modulo $p_i$. Since the entries of the matrix are bounded, we can bound the magnitude of

the determinant and hence the number of prime factors. If it has full row rank, it contains a square submatrix whose determinant is not divisible by $p_i$. Note that the primes $p_i$ depend on the anchor submatrix, so they may be correlated with the right half. We handle this by conditioning on the entries in the left half, which fixes the anchor submatrix and its prime factors. Lemma 8 then bounds the failure probability for the right half even under this conditioning.

Consequently, for any fixed set $\mathcal{H}$ of $n - t$ columns, we can bound the probability that $\mathbf{E}_{*,\mathcal{H}}$ does not contain square submatrices with coprime determinants. A union bound over the choices of $\mathcal{H}$ then provides a bound for all such column sets. However, recall from our discussion in Section 5.1.2 that for small prime factors, say $p_i = 2$, the bound from Lemma 8 might be too weak to absorb this union bound. Therefore, we fix a prime $q$ as the cutoff and use $p = q$ when choosing the anchor submatrix in the left half. We then apply the above approach only to prime factors $p_i > q$ of the anchor determinant. This bounds the probability that, for some such column set $\mathcal{H}$, a prime $p \geqslant q$ divides the determinants of all $\ell \times \ell$ submatrices of $\mathbf{E}_{*,\mathcal{H}}$.

To eliminate small prime factors, we use Lemma 9 for each prime $p < q$ to bound the probability that $\mathbf{E}_{*,\mathcal{H}}$ fails to have full row rank over $\mathbb{Z}_p$. We then take a union bound over all such primes. In particular, recall that Lemma 9 already provides a bound for all possible column sets $\mathcal{H}$, so a union bound over the choices of $\mathcal{H}$ is not required in this case.

Combining the small-prime and large-prime analyses with a union bound gives an upper bound on the probability that there exists a set $\mathcal{H}$ of $n - t$ columns for which $\mathbf{E}_{*,\mathcal{H}}$ does not contain square submatrices whose determinants are jointly coprime. In summary, the small-prime analysis uses Lemma 9 and takes a union bound over every prime less than $q$. The large-prime analysis first identifies the anchor submatrix in the left half and then takes a union bound over the prime factors of its determinant that are greater than $q$. The bound on the anchor determinant allows controlling the number of primes in the union bound.

The remaining step is to bound the Bézout coefficients of these determinants, as required by Theorem 7. As discussed in Section 2, the generic bound grows as a product of the magnitudes of the determinants, making the resulting statistical-distance bound too large. Instead, we exploit our process for identifying square submatrices: for each prime factor of the anchor determinant, we have another determinant that is not divisible by that prime. This lets us use CRT to obtain a coefficient bound that grows only linearly with the determinant bound and the number of prime factors. We first prove the Bézout coefficient bound in Lemma 10 and then prove the coprime submatrix property in Lemma 11.

**Lemma 10.** *Let $x_0, \ldots, x_d \in \mathbb{Z}$ be non-zero integers with $|x_0| > 1$ such that the prime factorization of $|x_0|$ is*

$$|x_0| = \prod_{i=1}^{d} p_i^{e_i}, \qquad \text{and} \qquad p_i \nmid x_i,\ \forall i \in [d].$$

*Then $\gcd(x_0, \ldots, x_d) = 1$ and there exist $\eta_0, \ldots, \eta_d \in \mathbb{Z}$ such that*

$$\sum_{i=0}^{d} \eta_i \cdot x_i = 1, \qquad |\eta_0| \leqslant d \cdot \max_{1 \leqslant i \leqslant d} |x_i|, \qquad \text{and} \qquad |\eta_i| < |x_0|, \forall i \in [d].$$

*Proof.* We describe how to construct integers $(\eta_0, \ldots, \eta_d)$ satisfying the required bounds such that $\sum_i \eta_i x_i = 1$. Note that this implies $\gcd(x_0, \ldots, x_d) = 1$. For each $i \in [d]$, since $x_i \not\equiv 0 \bmod p_i$, $x_i$ is invertible modulo $p_i^{e_i}$. Thus, from CRT, the following system of congruences

$$\eta_i \cdot x_i \equiv 1 \bmod p_i^{e_i} \qquad \text{and} \qquad \eta_i \equiv 0 \bmod p_j^{e_j},\ \forall j \in [d] \setminus \{i\}$$

has a unique solution $\eta_i \in [0, |x_0| - 1]$. Let $S = \sum_{i=1}^{d} \eta_i \cdot x_i$. Since $S \equiv 1 \bmod p_i^{e_i}$ for each $i \in [d]$, we have $S \equiv 1 \bmod |x_0|$ from CRT. This implies that $S = q \cdot x_0 + 1$, for some integer $q$. Setting $\eta_0 = -q$, we have

$$\eta_0 \cdot x_0 + \sum_{i=1}^{d} \eta_i \cdot x_i = 1.$$

We are left to bound $|\eta_0|$. Observe that

$$\begin{aligned}
|\eta_0| &= \left|\frac{S-1}{x_0}\right| \\
&\leqslant \frac{1+\sum_{i=1}^{d}|\eta_i|\cdot|x_i|}{|x_0|} && \triangleright \text{ Triangle inequality} \\
&\leqslant \frac{1+(|x_0|-1)\sum_{i=1}^{d}|x_i|}{|x_0|} && \triangleright\ |\eta_i| < |x_0| \\
&\leqslant \sum_{i=1}^{d}|x_i| && \triangleright\ \sum_{i=1}^{d}|x_i| \geqslant 1 \\
&\leqslant d \max_{1\leqslant i\leqslant d}|x_i|
\end{aligned}$$

This concludes the proof. ■

**Lemma 11.** *Let $n, \ell, B \in \mathbb{N}$ satisfy $B \geqslant 2$, let $t \in \mathbb{Z}_{\geqslant 0}$ satisfy $n - t \geqslant 2\ell$, and let $q$ be a prime. Then, for $\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)$ (Figure 4), the following holds except with probability at most $\min(1, \delta_{\mathsf{lme}})$: for all $\mathcal{H} \subseteq [n]$ satisfying $|\mathcal{H}| \geqslant n - t$, there exist $k \in \mathbb{N}$, $\ell$-sized column sets $C_1, \ldots, C_k \subseteq \mathcal{H}$, and integers $\eta_1, \ldots, \eta_k$ such that*

$$\sum_{i=1}^{k}\eta_i \cdot \det\left(\mathbf{E}_{*,C_i}\right) = 1 \qquad \text{and} \qquad \sum_{i=1}^{k}|\eta_i| \leqslant 2\ell!\, B^{\ell}\left\lfloor \log_2\left(\ell!\, B^{\ell}\right)\right\rfloor,$$

*where*

$$\delta_{\mathsf{lme}} = \sum_{\substack{p<q \\ p \text{ prime}}} \frac{p^{\ell}-1}{p-1}\Pr\left[\mathrm{Bin}(n,\alpha_p) \geqslant h\right] + \binom{n}{t}\left\lceil \log_q\left(\ell!\, B^{\ell}\right)\right\rceil \ell \cdot \Pr\left[\mathrm{Bin}\left(\lfloor h/2\rfloor, 1-\alpha_q\right) \leqslant \ell - 1\right],$$

$$h \coloneqq n - t, \qquad \text{and} \qquad \alpha_p \coloneqq \frac{\lceil B/p\rceil}{B}.$$

*Proof.* Let $h_{\mathsf{left}} \coloneqq \lfloor h/2\rfloor$ and $h_{\mathsf{right}} \coloneqq \lceil h/2\rceil$. For every set $\mathcal{H} \subseteq [n]$ of size $h$, let $\mathcal{H}_{\mathsf{left}}$ denote the first $h_{\mathsf{left}}$ indices in $\mathcal{H}$ and let $\mathcal{H}_{\mathsf{right}}$ denote the remaining indices. In other words, we partition $\mathbf{E}_{*,\mathcal{H}}$ into a left submatrix $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ with $h_{\mathsf{left}}$ columns and a right submatrix $\mathbf{E}_{*,\mathcal{H}_{\mathsf{right}}}$ with $h_{\mathsf{right}}$ columns. For every $h$-sized set $\mathcal{H}$ for which $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ has rank $\ell$ over $\mathbb{Z}_q$, let $C^{\star}(\mathcal{H}) \subseteq \mathcal{H}_{\mathsf{left}}$ be the lexicographically first $\ell$-sized column set such that

$$x^{\star}(\mathcal{H}) \coloneqq \det\left(\mathbf{E}_{*,C^{\star}(\mathcal{H})}\right) \not\equiv 0 \pmod{q}.$$

Such a set exists because a matrix of full row rank contains $\ell$ linearly independent columns.

We will show that, except with probability at most $\delta_{\mathsf{lme}}$, for every admissible $\mathcal{H}$, the matrix $\mathbf{E}_{*,\mathcal{H}}$ contains $\ell \times \ell$-order submatrices whose determinants satisfy the conditions of Lemma 10. Applying Lemma 10 to these determinants then concludes the proof. To bound the probability that such submatrices do not exist, we define the following large-prime and small-prime bad events.

- For every $h$-sized set $\mathcal{H}$, let $\mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H})$ denote the event that $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ is rank deficient over $\mathbb{Z}_q$, or that it has full row rank over $\mathbb{Z}_q$ but $\mathbf{E}_{*,\mathcal{H}_{\mathsf{right}}}$ is rank deficient over $\mathbb{Z}_p$ for some prime $p > q$ that divides $x^{\star}(\mathcal{H})$. Define

$$\mathsf{Bad}_{\mathsf{lrg}} \coloneqq \bigcup_{\substack{\mathcal{H}\subseteq[n] \\ |\mathcal{H}|=h}} \mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H}).$$

- Let $\mathsf{Bad}_{\mathsf{sml}}$ be the event that there exist a prime $p < q$ and a set $J \subseteq [n]$ satisfying $|J| \geqslant h$ such that $\mathbf{E}_{*,J}$ is rank deficient over $\mathbb{Z}_p$.

If neither bad event occurs, then for every $h$-sized set $\mathcal{H}$, the left submatrix provides an anchor determinant not divisible by $q$, while for every prime divisor $p$ of this determinant, either $\mathbf{E}_{*,\mathcal{H}}$ is full rank over $\mathbb{Z}_p$ when $p < q$, or $\mathbf{E}_{*,\mathcal{H}_{\mathsf{right}}}$ is full rank over $\mathbb{Z}_p$ when $p > q$. In either case, the full-rank matrix contains an $\ell \times \ell$ submatrix whose determinant is not divisible by $p$, which gives the determinants required by Lemma 10. We note that the proof only requires $C^\star(\mathcal{H})$ to be chosen deterministically from $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ so that $x^\star(\mathcal{H})$ is not divisible by $q$. This is because, when bounding $\mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H})$, we condition on $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$, after which $C^\star(\mathcal{H})$, $x^\star(\mathcal{H})$, and its prime divisors must be fixed.

The following claim bounds the probability of $\mathsf{Bad}_{\mathsf{lrg}}$.

**Claim 11.1.** *We have*

$$\Pr\left[\mathsf{Bad}_{\mathsf{lrg}}\right] \leqslant \binom{n}{t}\left\lceil \log_q\left(\ell!\, B^\ell\right)\right\rceil \ell \cdot \Pr\left[\mathrm{Bin}\left(h_{\mathsf{left}}, 1-\alpha_q\right) \leqslant \ell - 1\right].$$

*Proof of the claim.* Let $\delta_q = \ell \cdot \Pr\left[\mathrm{Bin}\left(h_{\mathsf{left}}, 1-\alpha_q\right) \leqslant \ell - 1\right]$. Fix an $h$-sized set $\mathcal{H}$. Apply Lemma 8 with $I = \emptyset$, $J = \mathcal{H}_{\mathsf{left}}$, and $p = q$. Since $|\mathcal{H}_{\mathsf{left}}| = h_{\mathsf{left}} \geqslant \ell$, this gives

$$\Pr\left[\mathrm{rank}_{\mathbb{Z}_q}\left(\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}\right) < \ell\right] \leqslant \ell \cdot \Pr\left[\mathrm{Bin}\left(h_{\mathsf{left}}, 1-\alpha_q\right) \leqslant \ell - 1\right] = \delta_q.$$

Condition on an arbitrary value $\mathbf{A}$ in the support of $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ that has full row rank over $\mathbb{Z}_q$. Under this conditioning, the set $C^\star(\mathcal{H})$, the determinant $x^\star(\mathcal{H})$, and its prime divisors are fixed. Let $D = \ell!\, B^\ell$ and $r$ be the number of distinct prime divisors of $x^\star(\mathcal{H})$ that are larger than $q$. If $r \geqslant 1$, then Fact 2 gives

$$q^r \;<\; \prod_{\substack{p \mid x^\star(\mathcal{H}) \\ p > q}} p \;\leqslant\; \left|x^\star(\mathcal{H})\right| \;\leqslant\; D.$$

When $r = 0$, the same bound follows from $D \geqslant 2$. Thus for all $r \geqslant 0$, $1 + r \leqslant \left\lceil \log_q D \right\rceil$. For each prime divisor $p$, applying Lemma 8 with $I = \mathcal{H}_{\mathsf{left}}$ and $J = \mathcal{H}_{\mathsf{right}}$ gives

$$\begin{aligned}
\Pr\left[\mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{E}_{*,\mathcal{H}_{\mathsf{right}}}\right) < \ell \;\middle|\; \mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}} = \mathbf{A}\right] &\leqslant \ell \cdot \Pr\left[\mathrm{Bin}\left(h_{\mathsf{right}}, 1-\alpha_p\right) \leqslant \ell - 1\right] \\
&\leqslant \ell \cdot \Pr\left[\mathrm{Bin}\left(h_{\mathsf{left}}, 1-\alpha_q\right) \leqslant \ell - 1\right] \\
&= \delta_q,
\end{aligned}$$

where the second inequality holds because $h_{\mathsf{right}} \geqslant h_{\mathsf{left}}$ and $p > q$ implies $\alpha_p \leqslant \alpha_q$.

A union bound therefore gives

$$\Pr\left[\exists \text{ prime } p > q :\; p \mid x^\star(\mathcal{H}) \text{ and } \mathrm{rank}_{\mathbb{Z}_p}\left(\mathbf{E}_{*,\mathcal{H}_{\mathsf{right}}}\right) < \ell \;\middle|\; \mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}} = \mathbf{A}\right] \leqslant r\delta_q \leqslant \left(\left\lceil \log_q D\right\rceil - 1\right)\delta_q.$$

Averaging this bound over all full-rank submatrices $\mathbf{A}$ and accounting for $\mathbf{E}_{*,\mathcal{H}_{\mathsf{left}}}$ not having full row-rank, we have

$$\Pr\left[\mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H})\right] \leqslant \left\lceil \log_q D\right\rceil \delta_q.$$

Finally, a union bound over the $\binom{n}{h} = \binom{n}{t}$ choices of $\mathcal{H}$ proves the claim. □

The following claim bounds the probability of the small-prime bad event.

**Claim 11.2.** *We have*

$$\Pr[\mathsf{Bad}_{\mathsf{sml}}] \leqslant \sum_{\substack{p < q \\ p \text{ prime}}} \frac{p^\ell - 1}{p - 1} \Pr\left[\mathrm{Bin}\left(n, \alpha_p\right) \geqslant h\right].$$

*Proof of the claim.* Apply Lemma 9 with $s = h$ for each prime $p < q$. A union bound over these primes gives the claimed inequality. □

Combining Claims 11.1 and 11.2 with a union bound gives

$$\Pr\left[\mathsf{Bad}_{\mathsf{lrg}} \cup \mathsf{Bad}_{\mathsf{sml}}\right] \leqslant \delta_{\mathsf{lme}}.$$

The following claim constructs the determinants needed to apply Lemma 10.

**Claim 11.3.** *Suppose that neither* $\mathsf{Bad}_{\mathsf{lrg}}$ *nor* $\mathsf{Bad}_{\mathsf{sml}}$ *occurs. For every set* $\mathcal{H} \subseteq [n]$ *satisfying* $|\mathcal{H}| \geqslant h$*, there exist* $d \in \mathbb{Z}_{\geqslant 0}$ *and* $\ell$*-sized column sets* $C_0, \ldots, C_d \subseteq \mathcal{H}$ *such that, defining*

$$x_i \coloneqq \det\left(\mathbf{E}_{*,\,C_i}\right) \qquad \text{for every } i \in [0, d],$$

*each* $x_i$ *is non-zero and there exist distinct primes* $p_1, \ldots, p_d$ *and positive integers* $e_1, \ldots, e_d$ *satisfying*

$$|x_0| = \prod_{i=1}^{d} p_i^{e_i} \qquad \text{and} \qquad p_i \nmid x_i \quad \text{for every } i \in [d].$$

*When* $d = 0$*, the product is understood to equal* 1.

*Proof of the claim.* Fix a set $\mathcal{H} \subseteq [n]$ satisfying $|\mathcal{H}| \geqslant h$, and assume without loss of generality that $|\mathcal{H}| = h$. Since $\mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H})$ does not occur, $C^\star(\mathcal{H})$ is well-defined. Set

$$C_0 \coloneqq C^\star(\mathcal{H}) \qquad \text{and} \qquad x_0 \coloneqq x^\star(\mathcal{H}).$$

The determinant $x_0$ is non-zero and is not divisible by $q$. If $|x_0| = 1$, then this concludes the proof for this case.

Suppose that $|x_0| > 1$, and write

$$|x_0| = \prod_{i=1}^{d} p_i^{e_i},$$

where $p_1, \ldots, p_d$ are distinct primes and $e_1, \ldots, e_d$ are positive integers. Fix $i \in [d]$. If $p_i < q$, then the fact that $\mathsf{Bad}_{\mathsf{sml}}$ does not occur ensures that $\mathbf{E}_{*,\,\mathcal{H}}$ has rank $\ell$ over $\mathbb{Z}_{p_i}$. If $p_i > q$, then the fact that $\mathsf{Bad}_{\mathsf{lrg}}(\mathcal{H})$ does not occur ensures that $\mathbf{E}_{*,\,\mathcal{H}_{\mathsf{right}}}$ has rank $\ell$ over $\mathbb{Z}_{p_i}$. Note that $p_i \neq q$ because $q \nmid x_0$. Hence, in either possible case, $\mathbf{E}_{*,\,\mathcal{H}}$ contains an $\ell$-sized column set $C_i$ for which

$$x_i \coloneqq \det\left(\mathbf{E}_{*,\,C_i}\right) \not\equiv 0 \pmod{p_i}.$$

Thus, $p_i \nmid x_i$ and $x_i \neq 0$ for every $i \in [d]$, which proves the claim. □

Fix a matrix $\mathbf{E}$ for which neither $\mathsf{Bad}_{\mathsf{lrg}}$ nor $\mathsf{Bad}_{\mathsf{sml}}$ occurs and a set $\mathcal{H} \subseteq [n]$ satisfying $|\mathcal{H}| \geqslant h$. Choose $d$, $C_0, \ldots, C_d$, and $x_0, \ldots, x_d$ as in Claim 11.3. If $d = 0$, then $|x_0| = 1$. Taking $k = 1$, $C_1 = C_0$, and $\eta_1 = x_0$ gives the required identity and satisfies the claimed coefficient bound because $\ell!\, B^\ell \geqslant 2$.

Suppose that $d \geqslant 1$. Applying Lemma 10 gives integers $\eta_0, \ldots, \eta_d$ satisfying

$$\eta_0 x_0 + \sum_{i=1}^{d} \eta_i x_i = 1,$$

together with

$$|\eta_0| \leqslant d \cdot \max_{1 \leqslant i \leqslant d} |x_i| \qquad \text{and} \qquad |\eta_i| < |x_0| \quad \text{for every } i \in [d].$$

For every $i \in [0, d]$, the value $x_i$ is the determinant of an $\ell \times \ell$ submatrix of $\mathbf{E}$, so Fact 2 gives $|x_i| \leqslant \ell!\, B^\ell$. Moreover,

$$2^d \leqslant \prod_{i=1}^{d} p_i \leqslant |x_0| \leqslant \ell!\, B^\ell,$$

and hence $d \leqslant \lfloor \log_2 (\ell!\, B^\ell) \rfloor$. It follows that

$$|\eta_0| + \sum_{i=1}^{d} |\eta_i| \leqslant 2d\ell!\, B^\ell \leqslant 2\ell!\, B^\ell \lfloor \log_2 (\ell!\, B^\ell) \rfloor.$$

Taking $k \coloneqq d + 1$ and reindexing the constructed column sets and coefficients from 1 through $k$ gives the lemma statement. Thus, except with probability at most $\delta_{\mathsf{lme}}$, the construction applies simultaneously to every admissible $\mathcal{H}$, completing the proof. ■

### 5.1.4 Linear Mask Extractor

We now combine Theorem 7 and Lemma 11 to show that Toeplitz matrices are linear mask extractors. In particular, we set the number of rows $\ell$ to be a constant fraction $\rho$ of the number of columns $n$, giving a constant extraction rate. We also allow for the number of fixed coordinates $t$ of the source to be a constant fraction $\tau$ of the source length $n$. We show that the failure probability $\delta_{\mathsf{lme}}$ is exponentially small in $n$. We choose the bounds on $\rho$ somewhat conservatively to handle all corruption fractions $\tau < 1/2$. We discuss concrete parameters in more detail in Section 5.3.

**Theorem 12.** *Let $0 < \tau < 1/2$ and $\rho > 0$ be constants satisfying $\rho < \min\{(1/2 - \tau)^2/2, 1/700\}$. Let $\ell \coloneqq \ell(n)$ and $t \coloneqq t(n)$ be defined as $\ell(n) \coloneqq \lfloor \rho n \rfloor$ and $t(n) \coloneqq \lfloor \tau n \rfloor$. For every constant $c \in \mathbb{N}$, all sufficiently large $n$, and every $B \in \mathbb{N}$ satisfying $n \leqslant B \leqslant n^c$, let $\mathsf{MGen}$ be the $(\ell, n)$-matrix generator that samples $\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)$ (Figure 4). Then, for all $B_{\mathsf{rand}} \in \mathbb{N}$ and $B_{\mathsf{sec}} \in \mathbb{Z}_{\geqslant 0}$, $\mathsf{MGen}$ is a $\left(t, B_{\mathsf{rand}}, B_{\mathsf{sec}}, \delta_{\mathsf{sd}}, e^{-\Omega(n)}\right)$-linear mask extractor (Definition 5), where*

$$\delta_{\mathsf{sd}} \coloneqq \min\left\{1, \frac{2\ell^2 (\ell B)^{2\ell} \log(\ell B) B_{\mathsf{sec}}}{B_{\mathsf{rand}}}\right\}.$$

*Proof.* Fix a constant $c \in \mathbb{N}$, and consider parameters $n$ and $B$ satisfying the conditions in the theorem statement. Let $h \coloneqq n - t$ and $h_{\mathsf{left}} \coloneqq \lfloor h/2 \rfloor$. Since $\tau < 1/2$ and $\rho < 1/700$, we have $h \geqslant 2\ell$ for all sufficiently large $n$. This allows applying Lemma 11 with $q = 17$. For every prime $p$, let $\alpha_p = \lceil B/p \rceil / B$. The failure probability in Lemma 11 is at most $\min\{1, \delta_{\mathsf{small}} + \delta_{\mathsf{large}}\}$, where

$$\delta_{\mathsf{small}} \coloneqq \sum_{\substack{p < 17 \\ p \text{ prime}}} \frac{p^\ell - 1}{p - 1} \Pr\left[\mathrm{Bin}(n, \alpha_p) \geqslant h\right],$$

$$\delta_{\mathsf{large}} \coloneqq \binom{n}{t} \left\lceil \log_{17} (\ell!\, B^\ell) \right\rceil \ell \cdot \Pr[\mathrm{Bin}(h_{\mathsf{left}}, 1 - \alpha_{17}) \leqslant \ell - 1].$$

We simplify these two terms separately.

*Small-prime term.* For every prime $p < 17$, we have

$$\alpha_p \leqslant \frac{1}{p} + \frac{1}{B} \leqslant \frac{1}{2} + \frac{1}{n}.$$

Moreover, $h \geqslant (1 - \tau)n$. The Chernoff–Hoeffding bound (Lemma 6) therefore gives

$$\Pr\left[\mathrm{Bin}(n, \alpha_p) \geqslant h\right] \leqslant \exp\left(-2n \left(\frac{1}{2} - \tau - \frac{1}{n}\right)^2\right).$$

There are six primes below 17, and each of them satisfies

$$\frac{p^\ell - 1}{p - 1} \leqslant p^\ell \leqslant 13^\ell.$$

Thus,

$$\delta_{\text{small}} \leqslant 6 \exp\left(\ell \ln 13 - 2n\left(\frac{1}{2} - \tau - \frac{1}{n}\right)^2\right).$$

The exponent inside the exponential, divided by $n$, converges to $\rho \ln 13 - 2(1/2 - \tau)^2$. This limit is negative because $\rho < (1/2 - \tau)^2/2$ and $\ln 13 < 4$, and hence $\delta_{\text{small}} = e^{-\Omega(n)}$.

*Large-prime term.* Let

$$\beta_n \coloneqq \frac{1}{17} + \frac{1}{n}.$$

We have $\alpha_{17} \leqslant \beta_n$ and $\ell \leqslant h_{\text{left}}/2$ for all sufficiently large $n$. Expanding the binomial lower tail and using $\binom{h_{\text{left}}}{j} \leqslant \binom{h_{\text{left}}}{\ell}$ for every $0 \leqslant j \leqslant \ell - 1$, we obtain

$$\begin{aligned}\Pr[\text{Bin}(h_{\text{left}}, 1 - \alpha_{17}) \leqslant \ell - 1] &\leqslant \ell \binom{h_{\text{left}}}{\ell} \beta_n^{h_{\text{left}} - \ell} \\ &\leqslant \ell \left(\frac{e h_{\text{left}}}{\ell}\right)^{\ell} \beta_n^{h_{\text{left}} - \ell}.\end{aligned}$$

Since $\binom{n}{t} \leqslant 2^n$, substituting the above bound into $\delta_{\text{large}}$ gives

$$\delta_{\text{large}} \leqslant \ell^2 \left\lceil \log_{17}\left(\ell!\, B^{\ell}\right)\right\rceil 2^n \left(\frac{e h_{\text{left}}}{\ell}\right)^{\ell} \beta_n^{h_{\text{left}} - \ell}.$$

Using $\tau < 1/2$ and $\rho < 1/700$, a direct calculation shows that

$$2^n \left(\frac{e h_{\text{left}}}{\ell}\right)^{\ell} \beta_n^{h_{\text{left}} - \ell} = 2^{-\Omega(n)}.$$

The bound $B \leqslant n^c$ also gives

$$\ell^2 \left\lceil \log_{17}\left(\ell!\, B^{\ell}\right)\right\rceil = n^{O(1)}.$$

Consequently, $\delta_{\text{large}} = e^{-\Omega(n)}$.

Combining the two bounds gives $\delta_{\text{small}} + \delta_{\text{large}} = e^{-\Omega(n)}$. Therefore, it follows from Lemma 11 that the required Bézout identity and coefficient bound hold simultaneously for every admissible $\mathcal{H}$, except with probability $e^{-\Omega(n)}$.

*Applying the template.* By construction, MGen is $B$-bounded, and $h \geqslant 2\ell$ implies that $t \leqslant n - \ell$. The coefficient bound from Lemma 11 is

$$\Lambda = 2\ell!\, B^{\ell} \left\lfloor \log\left(\ell!\, B^{\ell}\right)\right\rfloor.$$

Applying Theorem 7 with this value of $\Lambda$ gives a masking error of at most

$$\min\left\{1, \frac{2\ell^2 (\ell - 1)!\, \ell! B^{2\ell - 1} \left\lfloor \log\left(\ell!\, B^{\ell}\right)\right\rfloor B_{\text{sec}}}{B_{\text{rand}}}\right\}.$$

Finally, using $(\ell - 1)!\, \ell! \leqslant \ell^{2\ell - 1}$ and $\log\left(\ell!\, B^{\ell}\right) \leqslant \ell \log(\ell B)$, it follows that the masking error is at most $\delta_{\text{sd}}$. This completes the proof. ∎

## 5.2 Efficiency of Extraction

We now discuss the computational efficiency of mask extraction using the Toeplitz linear mask extractor. For a source vector $\mathbf{s} \in [0, B_{\text{rand}} - 1]^n$, extraction requires computing the matrix–vector product $\mathbf{E}\mathbf{s}$ over the integers. Prior work shows that Toeplitz matrix–vector multiplication can be performed in quasi-linear time using fast polynomial and integer multiplication [FK23, SS71]. Since $\ell \leqslant n$ and the matrix entries are bounded by $B$, the resulting extraction cost is

$$\tilde{O}(n \log(1 + nBB_{\text{rand}})) \quad \text{bit operations.}$$

For the constant-rate parameters $\ell = \Theta(n)$ of Theorem 12, this is $\tilde{O}(\log(1 + nBB_{\text{rand}}))$ bit operations per extracted mask, amortized over a batch.

## 5.3 Concrete Parameters

In Theorem 12, we assume that the number of columns $n$ is sufficiently large and impose conservative bounds on the number of rows $\ell$, including $\ell < n/700$, to ensure feasibility for every fixed corruption fraction $0 < \tau < 1/2$. We now explore concrete parameter choices through a tighter, computation-aided analysis of the bound from Lemma 11. We focus on the constraints imposed by the failure probability $\delta_{\mathsf{lme}}$, which is an intrinsic property of the linear mask extractor. The hypothesis $n \leqslant B \leqslant n^c$ of Theorem 12 is only used there to simplify the bound of Lemma 11 asymptotically; the computation-aided analysis below evaluates that bound directly, for which this hypothesis is not needed, and it uses a small constant entry bound $B$ (which also keeps the extracted masks small).

Recall that the source length corresponds to the number of columns $n$ of the linear mask extractor, while the number of extracted masks corresponds to its number of rows $\ell$. For a given corruption fraction $\tau$, we explore the maximum achievable extraction rate $\rho$. Then, for a fixed extraction rate and target failure probability, we analyze the minimum possible source length at which the Toeplitz linear mask extractor is viable.

**Asymptotic rate.** We begin by analyzing how the maximum extraction rate $\rho$ depends on the corruption fraction $\tau$ *asymptotically*, for sufficiently large values of the source length $n$. Our goal is to understand the main barriers to achieving higher extraction rates. To this end, we optimize over the choice of the cutoff $q$ between the small-prime and large-prime analyses, as well as the matrix entry bound $B$. When computing the concrete parameters later, we recompute the asymptotic extraction rate more precisely for our specific choice of $q$ and $B$.

Let $\widehat{\rho}$ denote the supremum of constant extraction rates for which some constant choices of $q > 2$ and $B$ make $\delta_{\mathsf{lme}}$ in Lemma 11 tend to zero as $n$ tends to infinity. Recall that $\delta_{\mathsf{lme}}$ consists of two terms, $\delta_{\mathsf{lme}} = \delta_{\mathsf{small}} + \delta_{\mathsf{large}}$, where $\delta_{\mathsf{small}}$ and $\delta_{\mathsf{large}}$ bound the failure probabilities from the small-prime and large-prime analyses, respectively. We consider the restrictions imposed by requiring each term to tend to zero separately, denoting the corresponding suprema by $\widehat{\rho}_{\mathsf{small}}$ and $\widehat{\rho}_{\mathsf{large}}$, respectively. We find that the small-prime analysis imposes $\widehat{\rho}_{\mathsf{small}} \leqslant 1 - H_2(\tau)$, while the large-prime analysis imposes $\widehat{\rho}_{\mathsf{large}} \leqslant (1-\tau)/2$. We analyze each case in more detail below.

- *Small-prime term:* From Lemma 11, we have

$$\delta_{\mathsf{small}} = \sum_{\substack{p<q \\ p \text{ prime}}} \frac{p^\ell - 1}{p-1} \Pr\big[\mathrm{Bin}(n, \alpha_p) \geqslant n - t\big].$$

  Here, $\alpha_p = \lceil B/p \rceil / B$, which is at most $1/p + 1/B$ and is smallest when $B$ is a multiple of $p$. For simplicity, let us assume that $B$ is a multiple of $p$ for each prime $p < q$. Writing $H_2(x) = -x \log_2 x - (1-x)\log_2(1-x)$ for the binary entropy function and applying the bound $\sum_{j=0}^{t} \binom{n}{j} \leqslant 2^{nH_2(t/n)}$, we have

$$\frac{p^\ell - 1}{p-1} \Pr[\mathrm{Bin}(n, 1/p) \geqslant n - t] \;=\; \frac{p^\ell - 1}{p-1}\, p^{-n} \sum_{j=0}^{t} \binom{n}{j} (p-1)^j \;\leqslant\; p^{-n(1-H_2(\tau)-\rho)+o(n)},$$

  since $H_2(\tau) + \tau \log_2(p-1) \leqslant H_2(\tau) \log_2 p$. Thus, every term in $\delta_{\mathsf{small}}$ decays exponentially with $n$ when $\rho < 1 - H_2(\tau)$. Since $q$ is constant, this implies that $\delta_{\mathsf{small}}$ tends to zero as $n$ tends to infinity. The threshold is tight for $p = 2$, and $\alpha_2 \geqslant 1/2$ for every $B$, so $\widehat{\rho}_{\mathsf{small}} \leqslant 1 - H_2(\tau)$ in this case.

- *Large-prime term:* The two-block argument requires $\ell \leqslant (n-t)/2$, which implies that $\widehat{\rho}_{\mathsf{large}} \leqslant (1-\tau)/2$ in this case. It turns out that every rate strictly below this limit makes the large-prime error $\delta_{\mathsf{large}}$ exponentially small in $n$ too, for suitable choices of $q$ and $B$. In more detail, let $m = \lfloor (n-t)/2 \rfloor$. Bounding the sum of binomial coefficients by $2^m$ as in the proof of Theorem 12 gives

$$\delta_{\mathsf{large}} \leqslant n^{O(1)}\, 2^{n+m} \alpha_q^{m-\ell}.$$

  Choosing a sufficiently large constant prime $q$ and taking $B$ divisible by $q$ gives $\alpha_q = 1/q$. Whenever $\rho < (1-\tau)/2$, the gap $m - \ell$ grows linearly with $n$, so a sufficiently small constant $\alpha_q$ makes this bound exponentially small.

| $\tau$ | Asymptotic row factors | | | $\rho = 0.9\,\widehat{\rho}$ | | $\rho = 0.5\,\widehat{\rho}$ | |
|---|---|---|---|---|---|---|---|
| | $\widehat{\rho}_{\mathsf{small}}$ | $\widehat{\rho}_{\mathsf{large}}$ | $\widehat{\rho}$ | $n$ | $\ell$ | $n$ | $\ell$ |
| 0.1 | 0.531 | 0.310 | 0.310 | 284 | 80 | 101 | 16 |
| 0.2 | 0.278 | 0.220 | 0.220 | 461 | 92 | 221 | 25 |
| 0.3 | 0.119 | 0.152 | 0.119 | 2,988 | 320 | 612 | 37 |
| 0.4 | 0.029 | 0.099 | 0.029 | 12,166 | 319 | 2,501 | 37 |

Table 1: Concrete parameters for the Toeplitz linear mask extractor with $q = 73$ and $B = 146$ at different corruption fractions. $\widehat{\rho}_{\mathsf{small}}$ and $\widehat{\rho}_{\mathsf{large}}$ are the limiting asymptotic extraction rates imposed by the small- and large-prime analyses, respectively. $\widehat{\rho} = \min(\widehat{\rho}_{\mathsf{small}}, \widehat{\rho}_{\mathsf{large}})$ is the resulting asymptotic extraction rate for the linear mask extractor. For each extraction rate $\rho$, $n$ is the minimum source length (the number of columns) from which the failure probability $\delta_{\mathsf{lme}}$ is at most $2^{-40}$. $\ell$ is the number of extracted masks (the number of rows) at that threshold.

Combining the two restrictions gives

$$\widehat{\rho} = \min\left\{\widehat{\rho}_{\mathsf{small}}, \widehat{\rho}_{\mathsf{large}}\right\} \leqslant \min\left\{1 - H_2(\tau), \frac{1-\tau}{2}\right\},$$

for suitable choice of constants $q$ and $B$. We have $\widehat{\rho}_{\mathsf{small}} = \widehat{\rho}_{\mathsf{large}}$ at $\tau \approx 0.1334$. Below this point, the restriction imposed by the large-prime analysis dominates, and we can achieve $\rho < (1-\tau)/2$. Observe that this requirement essentially stems from the large-prime analysis relying on the honest columns being split into two submatrices, each with at least $\ell$ columns. Above this point, the restriction imposed by the small-prime analysis dominates. In fact, as discussed above, this restriction stems from requiring every admissible submatrix of honest columns to have full row rank over $\mathbb{Z}_2$. Since this holds for all corruption fractions $0.1334 < \tau < 0.5$, it forms the main barrier to improving the extraction rate for higher corruption thresholds. Indeed, the large-prime analysis alone allows for much higher rates. For example, when $\tau = 0.2$, we have $\widehat{\rho}_{\mathsf{large}} = 0.4$ while $\widehat{\rho}_{\mathsf{small}} \approx 0.278$; when $\tau = 0.4$, we have $\widehat{\rho}_{\mathsf{large}} = 0.3$ while $\widehat{\rho}_{\mathsf{small}} \approx 0.029$.

**Computing concrete parameters.** To compute the extraction rate and the minimum viable number of columns, we must first fix the prime cutoff $q$ and the matrix entry bound $B$. We do this based on the analysis above. In particular, we choose $B$ to minimize the largest probability of a residue modulo $p$ when sampling uniformly from $[0, B-1]$. Since the limiting case for the small-prime analysis is $p = 2$ and the large-prime analysis depends on $\alpha_q$, we set $B = 2q$, giving $\alpha_2 = 1/2$ and $\alpha_q = 1/q$. We then use Python scripts to compute the asymptotic extraction rate for this choice of $B$ and different values of $q$. We find that $q = 73$ is a moderate choice: for $\tau = 0.1$ and 0.2, this choice captures approximately 81% and 79% of the asymptotic rate improvements obtained by increasing $q$ from 17 to 127. Increasing $q$ further from 73 to 127 raises the rates from 0.310 to 0.329 and from 0.220 to 0.242, respectively. At this fixed cutoff and entry bound, the asymptotic extraction rates imposed by the small-prime and large-prime analyses cross at $\tau \approx 0.2553$. For higher values of $\tau$, the restriction from the small prime analysis is the bottleneck, and increasing $q$ gives no improvement in the asymptotic extraction rate.

For the remainder of this subsection, we use $\widehat{\rho}$, $\widehat{\rho}_{\mathsf{small}}$, and $\widehat{\rho}_{\mathsf{large}}$ to refer to the asymptotic extraction rates at $q = 73$ and $B = 146$. For a given $\tau$, we first compute $\widehat{\rho}$. Then, we seek the minimum number of columns above which we can achieve a constant fraction of the asymptotic extraction rate with total failure probability $\delta_{\mathsf{lme}} \leqslant 2^{-40}$. The asymptotic and finite bottlenecks can differ: at $\tau = 0.2$ , large prime analysis determines $\widehat{\rho}$, but the small-prime error dominates at $\rho = 0.9\widehat{\rho}$, where $\delta_{\mathsf{small}} \approx 2^{-40.6}$ and $\delta_{\mathsf{large}} \approx 2^{-55.5}$.

**Analysis.** Table 1 reports the parameter values at various corruption fractions. For $\tau = 0.1$ and 0.2, achieving 90% of the asymptotic extraction rate with failure probability at most $2^{-40}$ requires only 284 and 461 source coordinates. These allow extraction of 80 and 92 masks, respectively. At 50% of the asymptotic rate, the minimum viable source lengths fall to 101 and 221. Looking ahead, each source coordinate is supplied by one party in our MPC protocol, so these source lengths correspond to a few hundred parties.

For $\tau = 0.3$ and 0.4, the small-prime constraint is much more restrictive. Achieving 90% of $\widehat{\rho}$ requires 2,988 and 12,166 columns, respectively. Reducing the extraction rate to 50% of $\widehat{\rho}$ lowers these thresholds to 612 and 2,501, a

nearly 5× reduction in both cases. Thus, lowering the extraction rate can make mask extraction possible from much smaller sources and consequently support MPC with fewer parties.

**Improving the extraction rate.** As discussed earlier, the $p = 2$ case is the main bottleneck for the small-prime analysis. If we restrict the small-prime failure bound to primes $p > 2$, then $\widehat{\rho}_{\mathsf{small}}$ increases to approximately 0.251 and 0.133 when $\tau$ is 0.3 and 0.4, respectively. In this case, the $p = 3$ term imposes the main restriction on $\widehat{\rho}_{\mathsf{small}}$. These rates exceed those allowed by the large-prime analysis at $q = 73$, so increasing $q$ could give further improvements.

However, omitting $p = 2$ means that, even when the bad event captured by the modified small-prime analysis does not occur, the determinants of the $\ell \times \ell$ submatrices used in the proof of Theorem 7 may have a greatest common divisor $d > 1$. Consequently, we can no longer guarantee masking of arbitrary secrets bounded by $B_{\mathsf{sec}}$. Nevertheless, the proof of Theorem 7 still gives an integer matrix $\mathbf{Q}$ satisfying $\mathbf{MQ} = dI_\ell$, where $\mathbf{M} = \mathbf{E}_{*,\mathcal{H}}$ is the submatrix formed by the columns corresponding to the uniformly random coordinates of the source. If every coordinate of the secret is divisible by $d$, then the same smudging argument lets us mask it, as in [GJMW24]. Since $d$ must be a power of 2 and is bounded by $\ell!\, B^\ell$, we can ensure this by reformatting the secrets as in [GJMW24], making each coordinate a multiple of $2^k$, where $k = \left\lfloor \log_2(\ell!\, B^\ell) \right\rfloor$. Therefore, omitting small primes from the analysis, increasing $q$, and reformatting the secrets accordingly may help reduce the restrictions imposed by small primes on the extraction rate.

# 6 Honest-Majority MPC

In this section, we use CRT-based secret sharing and the Toeplitz linear mask extractor to construct a semi-honest MPC protocol in the honest majority setting that is secure against *adaptive* corruptions.

As discussed in Section 2, the protocol is almost identical to that in [GJMW24], except that we use the Toeplitz linear mask extractor in place of the Vandermonde matrix to extract masks. This also removes the need for the secret reformatting step from [GJMW24]: the Toeplitz linear mask extractor can mask any bounded integer secret, while the earlier construction required secrets to be multiples of $\prod_{1 \leqslant i < j \leqslant n}(j - i)$. Informally, the resulting protocol supports evaluating circuits over any integer ring $\mathbb{Z}_{p_0}$ satisfying $\log p_0 = \Omega(n \log n)$. This improvement comes from the fact that the Toeplitz linear mask extractor's masking error in Theorem 12 is at most $n^{O(n)} B_{\mathsf{sec}} / B_{\mathsf{rand}}$. Here, $B_{\mathsf{sec}}$ bounds the magnitude of the secret and $B_{\mathsf{rand}}$ bounds the source randomness used to extract the masks. To make this masking error negligible, it suffices to choose $B_{\mathsf{rand}}$ satisfying $\log B_{\mathsf{rand}} = \log B_{\mathsf{sec}} + O(n \log n)$. The Vandermonde construction instead incurred an additive term of order $n^2 \log n$. Since we now need to share random values from a smaller range, we can use smaller share moduli and hence a smaller computation modulus $p_0$ while retaining constant communication overhead.

We describe our protocol in Figures 5 and 6 and state our main theorem below. We prove the theorem in Section 6.1. Before we proceed, we review relevant notation.

**Notation.** We consider an *adaptive* semi-honest adversary that corrupts, over the course of the execution, at most $t < n/2$ parties (the formal model is recalled in Section 6.1), where we let $\mathcal{I} \subseteq [n]$ denote the set of parties that are corrupted by the end of the execution and $\mathcal{H} = [n] \setminus \mathcal{I}$ denote the honest parties. The scheme is also parameterized by pairwise coprime positive integers $p_0, p_1, \ldots, p_n$, used for CRT secret sharing. The secret space is $\mathbb{Z}_{p_0}$. For each $i \in [n]$, party $\mathsf{P}_i$ receives a share in $\mathbb{Z}_{p_i}$. Note that $p_0$ is not required to be prime. Let

$$P_{\mathsf{all}} \coloneqq \prod_{i \in [n]} p_i, \qquad P_{\mathsf{max}} \coloneqq \max_{\substack{\mathcal{I} \subseteq [n] \\ |\mathcal{I}| \leqslant t}} \prod_{i \in \mathcal{I}} p_i,$$

where $P_{\mathsf{all}}$ denotes the product of all share moduli and $P_{\mathsf{max}}$ denotes the maximum product of share moduli across all possible sets of corrupt parties. Looking ahead, the $P_{\mathsf{all}}$ affects the correctness of the protocol while $P_{\mathsf{max}}$ affects security. For a fixed set of corrupt parties $\mathcal{I}$, we also write $P_{\mathcal{I}} = \prod_{i \in \mathcal{I}} p_i$ and $P_{\mathcal{H}} = \prod_{i \in \mathcal{H}} p_i$ for the products of the corrupt and honest parties' share moduli, respectively.

**Theorem 13** (Adaptively Secure Semi-honest Honest-Majority MPC)**.** *Let $\kappa \in \mathbb{N}$ denote the statistical security parameter. Let $0 < \tau < 1/2$ and $\rho > 0$ be constants satisfying $\rho < \min\left\{(1/2 - \tau)^2/2, 1/700\right\}$. Let $n$ be the number of parties. Let $\ell \coloneqq \ell(n)$ and $t \coloneqq t(n)$ be defined as $\ell(n) \coloneqq \lfloor \rho n \rfloor$ and $t(n) \coloneqq \lfloor \tau n \rfloor$. For every constant $c \in \mathbb{N}$, all sufficiently large $n$,*

*and every $B \in \mathbb{N}$ satisfying $n \leqslant B \leqslant n^c$, let* MGen *be the $(\ell, n)$-matrix generator that samples* $\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)$ *(Figure 4).*

*Consider integers $p_0, p_1, \ldots, p_n > 0$, which are pairwise coprime, and parameters $L_t, L_{2t} \in \mathbb{N}$. Then, for any circuit $C$ over $\mathbb{Z}_{p_0}$, the protocol in Figure 5 and 6 realizes the functionality $C$ among $n$ parties, with corruption threshold $t$, with the following guarantees:*

- ***Correctness.*** *It is perfectly correct, as long as:*

$$\log P_{\mathsf{all}} - 1 \geqslant \max \begin{Bmatrix} 2\log|C| + 2\log n + 2\log B + 2\log L_t + 2\log p_0 \\ \log n + \log B + \log L_{2t} + \log p_0 \end{Bmatrix}.$$

- ***Security.*** *It is $(2^{-\kappa} + \delta_{\mathsf{lme}})$-statistically secure against adaptive semi-honest adversaries corrupting at most $t$ parties, where $\delta_{\mathsf{lme}} = e^{-\Omega(n)}$ is the failure probability of the Toeplitz linear mask extractor in Theorem 12, when:*

$$\begin{aligned} \log L_{2t} - \kappa - n - 10 &\geqslant \log\log(nB) + (2\ell+4)\log n + (2\ell+2)\log B \\ &\quad + 3\log|C| + \log p_0 + \log L_t + \log P_{\mathsf{max}}, \\ \log L_t - \kappa - n - 6 &\geqslant \log n + \log|C| + \log P_{\mathsf{max}}. \end{aligned}$$

  *In particular, it is $2^{-\kappa+1}$-statistically secure if, in addition, $n \geqslant c_0\kappa$ for a sufficiently large constant $c_0$.*

- ***Efficiency.*** *For a non-king party $\mathsf{P}_i$ (with associated modulo $p_i$), the communication and computation complexity are*

$$O\left(\log P_{\mathsf{all}} \cdot \frac{|C|}{\ell} + |C| \cdot \log p_i\right),$$

  *and*

$$O\left((\log P_{\mathsf{all}} + n\log n) \cdot \frac{|C|}{\ell} + |C| \cdot \log p_i\right).$$

  *For the king, the additional communication and computation complexity is*

$$O(\log P_{\mathsf{all}} \cdot |C|) \quad \textit{and} \quad O(\log P_{\mathsf{all}} \cdot |C| \cdot \log n).$$

  *The overall communication and computation complexity for all parties are*

$$O(\log P_{\mathsf{all}} \cdot |C|) \quad \textit{and} \quad O\left(\log P_{\mathsf{all}} \cdot |C| \cdot \log^2 n\right).$$

**Honest-majority MPC with Minimal Overhead.** We now interpret the parameters needed in this theorem for the case when the modulus bit length is $\log p_0 = \Theta(n \log n)$ and show that our protocol achieves constant computational overhead in this case, up to polylogarithmic factors in the number of parties.

Set the size of each prime $p_i$ to be $m = \lceil c \cdot \log(p_0)/(n-2t) \rceil$, where $c$ is a constant. Note that since $t = \lfloor \tau n \rfloor$ and $\tau < 1/2$, we still have $m = \Theta(\log p_0 / n)$. Therefore,

$$n(m-1) \leqslant \log P_{\mathsf{all}} = \sum_{i=1}^{n} \log p_i \leqslant nm \qquad \text{and} \qquad \log P_{\mathsf{max}} \leqslant tm.$$

Assuming $n \geqslant c_0\kappa$ for a sufficiently large constant $c_0$ (so that the extractor failure probability $\delta_{\mathsf{lme}} = e^{-\Omega(n)}$ is at most $2^{-\kappa-1}$) and $\log|\mathrm{C}| = O(\log p_0)$, and applying Theorem 13 with $\kappa+1$ in place of $\kappa$, it follows that it suffices to satisfy

$$\begin{aligned} n(m-1) &\geqslant \max \begin{Bmatrix} 2\log L_t + O(n\log n) \\ \log L_{2t} + O(n\log n) \end{Bmatrix} \\ \log L_{2t} &\geqslant \log L_t + tm + O(n\log n) \\ \log L_t &\geqslant tm + O(n\log n) \end{aligned}$$

**Pre-processing Phase**

**Public parameter:** $\mathbf{E}$ is a linear mask extractor, generated by $\mathsf{MGen}(1^n)$.

1. Each party $\mathsf{P}_i$ does the following:
   - Sample $s^{(i)} \leftarrow \{0, 1, \ldots, p_0 - 1\}$.
   - Compute sharing $\left[\!\left[S_t^{(i)}\right]\!\right] \leftarrow \mathsf{Share}_{n,t}(s^{(i)})$ and $\left[\!\left[S_{2t}^{(i)}\right]\!\right] \leftarrow \mathsf{Share}_{n,2t}(p_0 - s^{(i)})$.
   - Send $\left[\!\left[S_t^{(i)}\right]\!\right]_j$ and $\left[\!\left[S_{2t}^{(i)}\right]\!\right]_j$ to $\mathsf{P}_j$ for every $j \in [n]$ such that $j \neq i$.
   - Receive $\left[\!\left[S_t^{(j)}\right]\!\right]_i$ and $\left[\!\left[S_{2t}^{(j)}\right]\!\right]_i$ from $\mathsf{P}_j$ for every $j \in [n]$ such that $j \neq i$.
2. After receiving the shares of masks from all parties, each party $\mathsf{P}_i$ locally extracts $\ell$ masks as:

$$\begin{pmatrix} \left[\!\left[R_t^{(1)}\right]\!\right]_i \\ \vdots \\ \left[\!\left[R_t^{(\ell)}\right]\!\right]_i \end{pmatrix} = \mathbf{E} \cdot \begin{pmatrix} \left[\!\left[S_t^{(1)}\right]\!\right]_i \\ \left[\!\left[S_t^{(2)}\right]\!\right]_i \\ \vdots \\ \left[\!\left[S_t^{(n)}\right]\!\right]_i \end{pmatrix}, \text{ and } \begin{pmatrix} \left[\!\left[R_{2t}^{(1)}\right]\!\right]_i \\ \vdots \\ \left[\!\left[R_{2t}^{(\ell)}\right]\!\right]_i \end{pmatrix} = \mathbf{E} \cdot \begin{pmatrix} \left[\!\left[S_{2t}^{(1)}\right]\!\right]_i \\ \left[\!\left[S_{2t}^{(2)}\right]\!\right]_i \\ \vdots \\ \left[\!\left[S_{2t}^{(n)}\right]\!\right]_i \end{pmatrix}.$$

   Note that the above computation done by $\mathsf{P}_i$ is under modulo $p_i$.

Steps 1 and 2 are repeated $\lceil |C|/\ell \rceil + 1$ times, with fresh randomness in each repetition, to generate enough (double-sharings of) masks for the online phase, namely, $\left\{\left[\!\left[R_t^{(1)}\right]\!\right], \left[\!\left[R_{2t}^{(1)}\right]\!\right]\right\}, \left\{\left[\!\left[R_t^{(2)}\right]\!\right], \left[\!\left[R_{2t}^{(2)}\right]\!\right]\right\}, \ldots,$ $\left\{\left[\!\left[R_t^{(|C|)}\right]\!\right], \left[\!\left[R_{2t}^{(|C|)}\right]\!\right]\right\}$, and, from the last repetition, one more mask $\left\{\left[\!\left[R_t^{(0)}\right]\!\right], \left[\!\left[R_{2t}^{(0)}\right]\!\right]\right\}$ for the output gate.

Figure 5: Pre-processing phase of our semi-honest, honest-majority MPC protocol.

to ensure correctness and security since $\log B = O(\log n)$ and $\log p_0 = \Theta(n \log n)$. Consequently, choosing $L_t$ and $L_{2t}$ so that

$$\log L_t = tm + \log p_0 \qquad \text{and} \qquad \log L_{2t} = 2tm + (c-2)\log p_0.$$

ensures perfect correctness and $2^{-\kappa}$ statistical security for sufficiently large $c$ (the $n + O(1)$ and $(2\ell + 4)\log n$ terms of Theorem 13 are absorbed in the $O(n \log n)$ slack, since $\ell \leqslant n$). With these parameters, the total communication complexity is $O(|\mathsf{C}| \log p_0)$ and the total computation complexity is $\widetilde{O}(|\mathsf{C}| \cdot \log p_0)$.

**Corollary 1.** *Let $\kappa$ denote the statistical security parameter and let $n \geqslant c_0\kappa$ denote the number of parties, for a sufficiently large constant $c_0$. For all constant $0 < \tau < 1/2$, positive integers $p_0$ satisfying $\log p_0 = \Theta(n \log n)$, circuits $\mathsf{C}$ over $\mathbb{Z}_{p_0}$ satisfying $\log |\mathsf{C}| = O(\log p_0)$, and sufficiently large $n$, the protocol described in Figures 5 and 6 realizes the functionality $\mathsf{C}$ among $n$ parties with corruption threshold $t = \lfloor \tau n \rfloor$. Moreover, it has perfect correctness, $2^{-\kappa}$ statistical security against adaptive semi-honest adversaries, total communication $O(|\mathsf{C}| \log p_0)$ and total computation $O\big(|\mathsf{C}| \cdot \log p_0 \cdot \log^2 n\big)$.*

## 6.1 Proof

We now prove Theorem 13.

**Correctness.** To ensure the correctness of the protocol, the integer value of the output gate (which by itself can be an input gate, add gate, or multiplication gate), that would be reconstructed by all parties, need to not exceed $P_{\mathsf{all}}$.

**Input-dependent Phase**

- **Input Sharing.** Each party $\mathsf{P}_i$, holding input $x^{(i)} \in \mathbb{Z}_{p_0}$ compute $\left[\!\left[X_t^{(i)}\right]\!\right] \leftarrow \mathsf{Share}_{n,t}(x^{(i)})$. It sends $\left[\!\left[X_t^{(i)}\right]\!\right]_j$ to all party $\mathsf{P}_j$.
- **Circuit Emulation.** Parties emulate the circuit evaluation gate by gate as follows.
  - *Addition* $x + y \in \mathbb{Z}_{p_0}$. Let $[\![X_t]\!]$ and $[\![Y_t]\!]$ denote the $(t, n)$-sharing of the input wires $x$ and $y$. Each party $\mathsf{P}_i$ simply sets $[\![X_t]\!]_i + [\![Y_t]\!]_i$ as its $(t, n)$-sharing of the output wire.
  - *Multiplication* $x \cdot y \in \mathbb{Z}_{p_0}$.[a] Let $[\![X]\!]$ and $[\![Y]\!]$ denote the secret sharing of the input wires $x$ and $y$. Let $\{[\![R_t]\!], [\![R_{2t}]\!]\}$ be the sharing of the next pair of random masks that is not consumed yet. Then:
    - Each party $\mathsf{P}_i$ sends to the king party
    $$m_i \coloneqq \left([\![X_t]\!]_i \cdot [\![Y_t]\!]_i\right) + [\![R_{2t}]\!]_i .$$
    - After receiving $\{m_1, \ldots, m_n\}$, the king party reconstructs
    $$m \leftarrow \mathsf{Recon}_{n,2t}(\{m_i\}_{i\in[n]}).$$
    It sets $\bar{m} \coloneqq m \bmod p_0$ and sends $\bar{m}$ to all non-king parties (Note that $\bar{m}$ is of length $\log p_0$, which is large. To avoid $n \cdot \log p_0$ communication, the king party can simply send $\bar{m} \mod p_i$ to the $i$-th party).
    - Each party (king and non-king) sets its own $(t, n)$-sharing of the output wire as $\bar{m} + [\![R_t]\!]$. (Since $[\![R_{2t}]\!]$ and $[\![R_t]\!]$ are extracted from sharings of $p_0 - s^{(i)}$ and of $s^{(i)}$, respectively, we have $R_{2t} + R_t \equiv 0 \pmod{p_0}$ and hence $\bar{m} + R_t \equiv x \cdot y \pmod{p_0}$.)
- **Output Masking.** Let $[\![W_t]\!]$ denote the secret sharing of the output wire $w$. Each party $\mathsf{P}_i$ broadcast its masked output share $\tilde{w}_i = [\![W]\!]_i + \left[\!\left[R_{2t}^{(0)}\right]\!\right]_i$. After receiving $\{\tilde{w}_1, \ldots, \tilde{w}_n\}$, parties reconstruct $\tilde{w} \leftarrow \mathsf{Recon}_{n,2t}(\{\tilde{w}_i\}_{i\in[n]})$.
- **Output reconstruction.** Each party $\mathsf{P}_i$ broadcast its $t$-share of randomness $r_i = \left[\!\left[R_t^{(0)}\right]\!\right]_i$. After receiving $\{r_1, r_2, \ldots, r_n\}$, parties reconstruct $r \leftarrow \mathsf{Recon}_{n,t}(\{r_i\}_{i\in[n]})$. Each party then obtains $w = \tilde{w} + r \mod p_0$ (recall that $R_{2t}^{(0)} + R_t^{(0)} \equiv 0 \pmod{p_0}$).

[a] For multiplication with a (public) scalar, we also need to perform an "integer reduction" step. We omit the details here as it follows the exact procedure in each multiplication gate.

Figure 6: Input-dependent phase of our semi-honest, honest-majority MPC protocol.

- *Input wires.* Let $W_{\mathsf{input}}$ denote the maximum value possible in an input gate. For the correctness of the protocol, $W_{\mathsf{input}}$ cannot exceed $P_{\mathsf{all}}$. To bound $W_{\mathsf{input}}$, we have each input $x \leqslant p_0$ is shared into an integer $X = x + up_0$, where $u \in \{0, 1, \ldots, L_t - 1\}$. Hence the upper bound of $W_{\mathsf{input}}$ is:
$$W_{\mathsf{input}} \leqslant L_t p_0. \tag{3}$$
- *Linear Mask Extractor.* Let $W_{t-\mathsf{rand}}$ (or $W_{2t-\mathsf{rand}}$) denote, respectively, the maximum value of random mask $R_t$ (or $R_{2t}$) obtained in the offline phase. Each value $R_t$ is obtained as a dot product between a $B$-bounded vector and a $W_{\mathsf{input}}$-bounded vector. Hence, the maximum value of a mask $R_t$ is:
$$W_{t-\mathsf{rand}} \leqslant n \cdot B \cdot W_{\mathsf{input}} \leqslant n \cdot BL_t p_0. \tag{4}$$

Similarly, we can obtain the maximum value of a mask $R_{2t}$ to be:

$$W_{2t-\mathsf{rand}} \leqslant n \cdot BL_{2t}p_0. \tag{5}$$

- *Multiplication wires output.* Let $W_{\mathsf{mult}}$ denote the maximum value possible in after a multiplication gate. This output has form $R_t + x$, where $R_t$ is a mask extracted in the offline phase, and $x$ is a reconstructed value modulo $p_0$. The upper bound of output for multiplication gate is then:

$$W_{\mathsf{mult}} \leqslant W_{t-\mathsf{rand}} + p_0 \leqslant n \cdot BL_t p_0 + p_0. \tag{6}$$

- *Addition wires output.* Let $W_{\mathsf{add}}$ denote the maximum value of output in an addition gate. Each addition gate can contain a sum of at most $|C|$ inputs (being outputs from other gates than addition), in which each input value can be, in the worst case, as large as the output of a multiplication gate. Hence, the upper bound of output value for an addition gate is:

$$W_{\mathsf{add}} \leqslant |C| \cdot W_{\mathsf{mult}} \leqslant |C| \cdot (nBL_t + 1)p_0. \tag{7}$$

- *Multiplication wires reconstruction.* Let $W_{\mathsf{mult-recon}}$ denote the maximum value that needs to be reconstructed in the first step of the multiplication gate. Namely, this is the upper bound of the value $X \cdot Y + R_{2t}$, where $X$ and $Y$ are prior gates' outputs (worst case being addition gate), and $R_{2t}$ is a mask obtained in the offline phase. We can then bound $W_{\mathsf{mult-recon}}$ as:

$$W_{\mathsf{mult-recon}} \leqslant W_{\mathsf{add}}^2 + W_{2t-\mathsf{rand}} \leqslant |C|^2 \cdot (nBL_t + 1)^2 p_0^2 + n \cdot BL_{2t}p_0. \tag{8}$$

In conclusion, for $W_{\mathsf{input}}$, $W_{\mathsf{mult}}$, $W_{\mathsf{add}}$ and $W_{\mathsf{mult-recon}}$ all not exceeding $P_{\mathsf{all}}$, the parameters $B$, $L_t$ and $L_{2t}$ can be chosen to satisfy the following inequalities:

$$P_{\mathsf{all}} \geqslant |C| \cdot (nBL_t + 1)p_0, \tag{9}$$

and

$$P_{\mathsf{all}} \geqslant 2 \max \begin{Bmatrix} |C|^2 \cdot (nBL_t + 1)^2 p_0^2 \\ n \cdot BL_{2t}p_0 \end{Bmatrix}, \tag{10}$$

where (10) in fact implies (9). Taking log, we have:

$$\log P_{\mathsf{all}} - 1 \geqslant \max \begin{Bmatrix} 2\log|C| + 2\log n + 2\log B + 2\log L_t + 2\log p_0 \\ \log n + \log B + \log L_{2t} + \log p_0 \end{Bmatrix}. \tag{11}$$

**Efficiency.** We analyse the communication and computation cost here.

- *Matrix-vector multiplication in preprocessing.* In the preprocessing phase, party $\mathsf{P}_i$ needs to do a multiplication between the sampled linear mask extractor $\mathbf{E}$ and a vector of length $n$. The multiplication is taken in $\mathbb{Z}_{p_i}$. The computational cost to extract one batch of $\ell$ double sharings is quasi-linear, following Section 5.2.

- *Non-king parties.* There are $|C|/\ell$ random mask generating routines to do in the offline phase, each requires broadcasting shares for each other party, taking $\log P_{\mathsf{all}}$ communication. In the online phase, there are at most $|C|$ multiplication gates, each requires the party to broadcast one share of size $\log p_i$. The total communication cost is $O\left(\log P_{\mathsf{all}} \cdot \frac{|C|}{\ell} + |C| \cdot \log p_i\right)$.

  For computation in the offline phase, there are $|C|/\ell$ batches, each requires two multiplications between a Toeplitz matrix of size $\ell \times n$ and a vector of size $n$. Each of this multiplication's complexity is $O(n \log n)$. In each batch, the other cost is to generate random value, which is $O(\log P_{\mathsf{all}})$. For each gate of the online phase, the party performs a constant number of additions and multiplications modulo $p_i$ on its shares, which costs $O(\log p_i)$ (up to polylogarithmic factors); the reconstruction in multiplication gates is delegated to the king.

  Overall, the computation cost is $O\left((\log P_{\mathsf{all}} + n \log n) \cdot \frac{|C|}{\ell} + |C| \cdot \log p_i\right)$.

- *King party.* The only additional cost for a king party is the cost of reconstructing the masked value in each multiplication gate, which takes $O(\log P_{\mathsf{all}})$ communication to broadcast and $O(\log P_{\mathsf{all}} \cdot \log n)$ to compute [21]. Overall, the total additional communication cost is $O(\log P_{\mathsf{all}} \cdot |C|)$ and the total additional computation cost is $O(\log P_{\mathsf{all}} \cdot \log n \cdot |C|)$.
- *Total cost.* Summing the costs of the $n$ non-king parties and adding the additional cost of the king, and using $\ell = \lfloor \rho n \rfloor = \Theta(n)$ and $\sum_i \log p_i = \log P_{\mathsf{all}}$, the total communication is $O(\log P_{\mathsf{all}} \cdot |C|)$ and the total computation is $O((\log P_{\mathsf{all}} + n \log n) \cdot |C| + \log P_{\mathsf{all}} \cdot \log n \cdot |C|) = O(\log P_{\mathsf{all}} \cdot \log n \cdot |C|)$, since $n \leqslant \log P_{\mathsf{all}}$ (the moduli $p_1, \ldots, p_n$ are distinct integers larger than 1). Here, as above, a Toeplitz matrix–vector product is counted as $O(n \log n)$ arithmetic operations on $O(\log(nBp_i))$-bit integers (Section 5.2); carrying it out with fast integer multiplication [SS71] costs an additional factor of at most $O(\log n \cdot \log\log n)$ in the parameter regime of Corollary 1, where $\log p_i = O(\log n)$, which gives the bound $O\big(\log P_{\mathsf{all}} \cdot \log^2 n \cdot |C|\big)$ stated in Theorem 13.

**The adaptive adversary.** We prove security against an *adaptive* semi-honest adversary $\mathcal{A}$. The public parameter $\mathbf{E}$ is sampled first and given to $\mathcal{A}$. Afterwards, at any point of the execution (including after the output has been delivered), $\mathcal{A}$ may corrupt an additional party, chosen as a function of everything it has observed so far, subject to corrupting at most $t$ parties in total. Upon corrupting $\mathsf{P}_j$, the adversary learns the input $x^{(j)}$, the random tape of $\mathsf{P}_j$, and all messages that $\mathsf{P}_j$ has sent and received so far; from then on, it also observes every message that $\mathsf{P}_j$ receives. In the ideal world, the simulator $\mathcal{S}$ is given $\mathbf{E}$ and, as prescribed by the ideal experiment, the output $y = C(\mathbf{x})$ at the outset; whenever $\mathcal{A}$ corrupts $\mathsf{P}_j$, the simulator receives $x^{(j)}$ from the ideal functionality and must produce the internal state and message history of $\mathsf{P}_j$. Since statistical distance is convex, we may assume without loss of generality that $\mathcal{A}$ is deterministic. We may also assume that $\mathcal{A}$ corrupts exactly $t$ parties: an adversary that corrupts fewer parties can be modified to corrupt additional parties after the execution ends and to ignore their views, which does not change its output. We write $\mathcal{I}$ for the set of parties corrupted by the end of the experiment, $\mathcal{H} = [n] \setminus \mathcal{I}$, and $\mathcal{I}_k$ for the set of the first $k$ corrupted parties, so that $\emptyset = \mathcal{I}_0 \subsetneq \mathcal{I}_1 \subsetneq \cdots \subsetneq \mathcal{I}_t = \mathcal{I}$.

**The $J$-execution.** The simulator is built from the following family of executions, one for each candidate set of corrupted parties. For a set $J \subseteq [n]$ with $|J| \leqslant t$, inputs $x_J = (x^{(i)})_{i \in J}$ and an output value $y \in \mathbb{Z}_{p_0}$, the *$J$-execution* $\mathsf{Exec}_J(x_J, y)$, described in Figure 7, is a distribution over complete executions of the protocol (that is, over the joint views of all $n$ parties), in which the parties in $J$ use their real inputs, the parties outside $J$ use the input 0, and the output masking step is shifted so that the reconstructed output equals $y$. Two observations will be used repeatedly. First, in $\mathsf{Exec}_J(x_J, y)$ the shift $\Delta_J$ satisfies $\Delta_J \equiv 0 \pmod{p_i}$ for every $i \in J$, so the parties in $J$ send and receive exactly what they would send and receive in an honest execution, and every party outputs $\tilde{w} + r \equiv w + \Delta_J \equiv y \pmod{p_0}$, where $\tilde{w} = w + \Delta_J + R^{(0)}_{2t}$ is the reconstructed masked output and $r = R^{(0)}_t$ (recall that $R^{(0)}_{2t} + R^{(0)}_t \equiv 0 \pmod{p_0}$). Second, if $J = [n]$ and $y = C(\mathbf{x})$, then $w = C(\mathbf{x})$ and hence $\Delta_J = 0$: the execution $\mathsf{Exec}_{[n]}(\mathbf{x}, C(\mathbf{x}))$ is exactly the real execution of the protocol on input $\mathbf{x}$, which we denote by $\mathsf{Real}(\mathbf{x})$. Finally, the distribution of an execution up to (but excluding) the output masking step does not depend on $y$.

**The adaptive simulator.** The simulator, described in Figure 8, maintains a complete $J$-execution for the current set $J$ of corrupted parties and reveals the views of the corrupted parties in it. Whenever a new party is corrupted, it *resamples* the execution from the $J$-execution for the enlarged set, conditioned on everything that has already been revealed to the adversary. Since statistical security does not require the simulator to be efficient, $\mathcal{S}$ may be computationally unbounded, and in particular it can sample from this conditional distribution.

**Overview of the analysis.** The simulator in Figure 8 only ever reveals views taken from $J$-executions with output value $y = C(\mathbf{x})$, so the output of the ideal-world execution is exactly the output of the real-world one. The proof that the view of $\mathcal{A}$ is statistically close in both worlds proceeds in two steps. In the first step (Lemma 16), we fix an arbitrary set $J$ with $|J| \leqslant t$ and show that the joint view of the parties in $J$ is statistically close in $\mathsf{Real}(\mathbf{x})$ and in $\mathsf{Exec}_J(x_J, C(\mathbf{x}))$. This is a static statement, and its proof (Lemmas 14 and 15 and Claim 2) is where the parameters of the protocol enter. The one place where adaptivity already matters in this step is the public matrix $\mathbf{E}$: the adversary chooses whom to corrupt *after* seeing $\mathbf{E}$, so it does not suffice that a random matrix is good for a fixed set $J$; rather, a single sampled $\mathbf{E}$ has to be good for *all* sets $J$ simultaneously. This is precisely what Definition 5 provides (the

[21] One can compute the CRT of $n$ modulos $p_1, \ldots, p_n$ in $(\log p_1 + \log p_2 + \ldots + \log p_n) \log n$ time using a divide-and-conquer approach.

The $J$-execution $\mathsf{Exec}_J(x_J, y)$ is parameterized by a set $J \subseteq [n]$ with $|J| \leqslant t$, inputs $x_J$ for the parties in $J$, and an output value $y \in \mathbb{Z}_{p_0}$. All $n$ parties run the protocol of Figures 5 and 6 with the following modifications.

- **Pre-processing Phase.** Every party $\mathsf{P}_i$ runs the pre-processing phase honestly: it samples $s^{(i)}$, distributes $\left[\!\left[S_t^{(i)}\right]\!\right]$ and $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$, and locally extracts $\left[\!\left[R_t^{(1)}\right]\!\right]_i, \ldots, \left[\!\left[R_t^{(\ell)}\right]\!\right]_i$ and $\left[\!\left[R_{2t}^{(1)}\right]\!\right]_i, \ldots, \left[\!\left[R_{2t}^{(\ell)}\right]\!\right]_i$ as in Step 2.
- **Input Sharing.** Every party $\mathsf{P}_i$ with $i \in J$ shares its input $x^{(i)}$, while every party $\mathsf{P}_i$ with $i \notin J$ shares the input 0 instead. All sharings $\left[\!\left[X_t^{(i)}\right]\!\right]$ are computed honestly.
- **Circuit Emulation.** All parties follow the circuit emulation honestly.
- **Output Masking.** Let $w$ denote the value reconstructed from the shares $[\![W_t]\!]_1, \ldots, [\![W_t]\!]_n$ of the output wire, i.e., the output of the circuit on the modified inputs. Let $\Delta_J$ be the unique integer in $\left[0, p_0 \cdot \prod_{i \in J} p_i\right)$ such that
$$\Delta_J = y - w \mod p_0, \text{ and } \Delta_J = 0 \mod \prod_{i \in J} p_i, \tag{12}$$
which exists by the CRT. Every party $\mathsf{P}_i$ with $i \notin J$ broadcasts $[\![W_t]\!]_i + \left[\!\left[R_{2t}^{(0)}\right]\!\right]_i + \Delta_J \mod p_i$, while every party $\mathsf{P}_i$ with $i \in J$ broadcasts $[\![W_t]\!]_i + \left[\!\left[R_{2t}^{(0)}\right]\!\right]_i$ as in the protocol. All parties reconstruct the value $\tilde{w}$.
- **Output Reconstruction.** All parties follow this step honestly, obtaining in the end the value $\tilde{w} + r \bmod p_0$.

Figure 7: The $J$-execution $\mathsf{Exec}_J(x_J, y)$. For $J = [n]$ and $y = C(\mathbf{x})$, this is the real execution of the protocol.

The simulator $\mathcal{S}$ maintains the set $J$ of currently corrupted parties, their inputs $x_J$, and a complete execution $e$. It proceeds as follows.

- **Initialization.** Upon receiving the public parameter $\mathbf{E}$ and the output $y = C(\mathbf{x})$, set $J \leftarrow \emptyset$ and sample $e \leftarrow \mathsf{Exec}_\emptyset(\perp, y)$.
- **Honest rounds.** As the (simulated) protocol proceeds, whenever a party $\mathsf{P}_j$ with $j \in J$ receives a message in $e$, forward this message to $\mathcal{A}$.
- **Corruption.** When $\mathcal{A}$ corrupts a party $\mathsf{P}_j$ with $j \notin J$ at some point $\tau$ of the execution, receive $x^{(j)}$ from the ideal functionality, and set $J' \leftarrow J \cup \{j\}$. Let $\mathsf{rev} \coloneqq \mathsf{View}_J^{\leqslant \tau}(e)$ denote everything revealed to $\mathcal{A}$ so far, namely the inputs, random tapes, and messages sent and received up to point $\tau$ of all parties in $J$, as determined by $e$. Sample a fresh execution
$$e' \leftarrow \mathsf{Exec}_{J'}(x_{J'}, y) \quad \text{conditioned on} \quad \mathsf{View}_J^{\leqslant \tau}(e') = \mathsf{rev}$$
(if this event has probability zero, output $\perp$ and halt), and set $e \leftarrow e'$ and $J \leftarrow J'$. Reveal $\mathsf{View}_{\{j\}}^{\leqslant \tau}(e)$, i.e., the input, random tape, and message history of $\mathsf{P}_j$ up to point $\tau$ in $e$, to $\mathcal{A}$.
- **Output.** Output whatever $\mathcal{A}$ outputs.

Figure 8: The adaptive simulator for the MPC protocol.

existential quantifier over the source is inside the probability), and it is the reason why Theorem 12 has to control *every* $\ell \times (n - t)$ submatrix of the Toeplitz matrix rather than a random one. In the second step (Lemma 17), we show by a hybrid argument over the corruption sequence that the resampling simulator of Figure 8 turns the static guarantee of Lemma 16 into security against adaptive corruptions, at the cost of a factor that is exponential in $n$ but is absorbed by the parameters of Theorem 13.

**Step 1: Static input switching.** Fix $J \subseteq [n]$ with $|J| \leqslant t$ and inputs $\mathbf{x}$, and let $y = C(\mathbf{x})$. For the rest of this step, we

refer to $\mathsf{Real}(\mathbf{x})$ as the *real world*, to $\mathsf{Exec}_J(x_J, y)$ as the *ideal world* (or the *simulated* execution), and, with a slight abuse of terminology, to the parties in $\mathcal{I} \coloneqq J$ as the corrupted parties and to those in $\mathcal{H} \coloneqq [n] \setminus J$ as the honest parties. The adversary view in both worlds, receiving from an honest party $\mathsf{P}_i$, $i \in \mathcal{H}$, consists of:

1. The random samples shares in the offline phase: $\left[\!\left[S_t^{(i)}\right]\!\right]_j$ and $\left[\!\left[S_{2t}^{(i)}\right]\!\right]_j$, received by each corrupt party $\mathsf{P}_j$, $j \in \mathcal{I}$. These shares are also identically distributed in both worlds.

2. The input shares $\left[\!\left[X^{(i)}\right]\!\right]_j$, received by each corrupt party $\mathsf{P}_j$, $j \in \mathcal{I}$. Looking ahead, we will first show that the marginal distribution of these shares are statistically close in both worlds, by the security of CRT sharing, in Lemma 14. We then finish the proof by showing that, conditioned on any fixed set of adversary shares, the later adversary view in both worlds are also statistically close.

3. In each multiplication gate, every party will receive the reconstructed masked output $X \cdot Y + R_{2t}$, where $R_{2t}$ is one of the randomness extracted in the offline phase. Instead of analyzing each value, we consider the distribution of each batch of $\ell$ masks extracted in the offline phase. We show in Lemma 15 that the joint distribution of each batch of $\ell$ reconstructed values in multiplication gates in both worlds are statistically close, conditioned on any fixed set of corrupt parties' shares.

4. In the output masking step, every paty will receive the reconstructed masked output $W + R_{2t}^{(0)}$, where $R_{2t}^{(0)}$ is one of the randomness extracted in the offline phase. The statistical argument here is also batched into Lemma 15.

5. In the final reconstruction step, the distribution of the randomness $R_t^{(0)}$ is the same in both worlds.

We first show that the marginal distribution for each set of shares for corrupted parties is close to uniform, by proving the security of CRT-based secret sharing.

**Lemma 14** (Security of CRT sharing)**.** *Assume that $p_0, p_1, \ldots, p_n$ are coprime. For any integer $x$ and $L$, the distribution of the shared value:*

$$\left(x + p_0 \cdot \mathcal{U}_{\{0,1,\ldots,L-1\}}\right) \mod P_{\mathcal{I}} \tag{13}$$

*is $\frac{P_{\mathcal{I}}}{L}$-close to uniform, where $\mathcal{U}_{\{0,1,\ldots,L-1\}}$ is the uniform distribution over $\{0, 1, \ldots, L-1\}$.*

*Proof.* Let $L' \leqslant L$ be the largest multiple of $P_{\mathcal{I}}$ that is at most $L$. Then the following distribution is simply $L'/P_{\mathcal{I}}$ copies of the uniform distribution over $\mathbb{Z}_{P_{\mathcal{I}}}$:

$$\left\{\left(x + p_0 \cdot \mathcal{U}_{\{0,1,\ldots,L'-1\}}\right) \mod P_{\mathcal{I}}\right\} = \mathcal{U}_{\{0,1,\ldots,P_{\mathcal{I}}-1\}}. \tag{14}$$

We also have:

$$\begin{aligned}
&\mathsf{SD}\left(\left(x + p_0 \cdot \mathcal{U}_{\{0,1,\ldots,L'-1\}}\right) \mod P_{\mathcal{I}},\ \left(x + p_0 \cdot \mathcal{U}_{\{0,1,\ldots,L-1\}}\right) \mod P_{\mathcal{I}}\right) \\
&= \mathsf{SD}\left(\mathcal{U}_{\{0,1,\ldots,L'-1\}},\ \mathcal{U}_{\{0,1,\ldots,L-1\}}\right) \\
&= \frac{L - L'}{L} \leqslant \frac{P_{\mathcal{I}}}{L},
\end{aligned}$$

which concludes the proof. ∎

As we have now proven the security of marginal distribution for each set of shares for corrupted parties, from now on, we focus on the conditional distribution, regarding one particular set of shares. That is, in the rest of the proof, we assume that all shares $\left[\!\left[X^{(i)}\right]\!\right]_j$ for each $j \in \mathcal{I}$ are fixed. We note that it means we also have a security invariant, similar to [GJMW24], stated in Claim 2 and proven similarly to Claim 3 in [GJMW24], adapted to our case when the input reformatting is no longer required.

**Claim 2.** *Conditional on a fixed set of input shares for the corrupted parties, the real and the simulated executions can be coupled so that, for any wire $w$, the wire value $W$ in the simulated execution differs from the wire value in the real*

*execution by a shift $\Delta$ that is a multiple of $P_{\mathcal{I}}$, while the share $[\![W]\!]_i$ of every corrupted party $\mathsf{P}_i$, $i \in \mathcal{I}$, is identical in both executions.*

*Furthermore, $\Delta$ is always upper bounded by the following quantity:*

$$|\Delta| \leqslant 2|C| \cdot nB \cdot p_0 \cdot P_{\mathcal{I}}. \tag{15}$$

*Proof.* Considering an *input wire*, by Figure 3 the wire value has the form $s + up_0$, where $s \in \mathbb{Z}_{p_0}$ is the secret input (the real input in the real world, and 0 in the simulated one), and $u \in [0, L_t - 1]$. The shift $\Delta$ between the two wire values is determined by the following properties:

- $\Delta \bmod p_0$ is the difference between the input value $s$ in the real world and in the simulated view. Let this difference be $s_\Delta$, then $\Delta = s_\Delta \bmod p_0$.
- Conditioned on a fixed set of input shares for the corrupted parties, we then have $\Delta = 0 \bmod p_i$ for any $i \in \mathcal{I}$.

This shift $\Delta$ is then divisible by $P_{\mathcal{I}}$, and by the CRT, we can couple the offsets $u$ of the two executions so that $0 \leqslant \Delta < p_0 \cdot P_{\mathcal{I}}$ (the two offsets then differ by less than $P_{\mathcal{I}}$; the cost of this coupling is accounted for in Lemma 16).

For the output wire of a *multiplication gate*, recall that every party sets its share of the output wire to $\bar{m} + [\![R_t]\!]_i$, where $\bar{m}$ is the public value reconstructed by the king and $[\![R_t]\!]$ is a sharing extracted in the pre-processing phase from the sharings $\left[\!\left[S_t^{(i)}\right]\!\right]$. The two executions are coupled so that the public value $\bar{m}$ is the same in both (the cost of this coupling is bounded in Lemma 15). The sharings $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$ from which $[\![R_{2t}]\!]$ is extracted may then differ between the two executions and, since $\left[\!\left[S_t^{(i)}\right]\!\right]$ and $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$ share the secret $s^{(i)}$, so may the sharings $\left[\!\left[S_t^{(i)}\right]\!\right]$. However, the secrets $s^{(i)} \in \mathbb{Z}_{p_0}$ differ by less than $p_0$, and the offset of $\left[\!\left[S_t^{(i)}\right]\!\right]$ can be coupled, keeping the shares of the corrupted parties fixed, so that the integer $S_t^{(i)}$ differs by a multiple of $P_{\mathcal{I}}$ of magnitude less than $p_0 + p_0 P_{\mathcal{I}} \leqslant 2p_0 P_{\mathcal{I}}$. Hence the masks $R_t$ of the two executions, and with them the output wire $\bar{m} + R_t$, differ by $\mathbf{E}$ applied to these differences, i.e., by a multiple of $P_{\mathcal{I}}$ of magnitude at most $2nB \cdot p_0 \cdot P_{\mathcal{I}}$. In particular, the shift of the output wire of a multiplication gate does not depend on the shifts of its input wires: shifts do not accumulate across multiplication gates.

For other gates, in the worst case, the output wire of this gate is a sum of $|C|$ other wire values, each of which is an input wire or the output wire of a multiplication gate, so its shift is divisible by $P_{\mathcal{I}}$ and upper bounded by $2|C| \cdot nB \cdot p_0 \cdot P_{\mathcal{I}}$. ∎

We are now ready to come back to finish the proof, knowing that the wire values in both worlds differ by a multiple of $P_{\mathcal{I}}$. The following lemma, not only shows the small statistical distance between the masked value in multplication gate in the ideal world and in the real world, but also argues for the security of the masked output value.

**Lemma 15** (Security of multiplication gates and masked output). *Let $\mathbf{E} \leftarrow \mathsf{GenToeplitz}_{\ell,B}(1^n)$ be sampled by the Toeplitz matrix generator with the parameters of Theorem 12, and let*

$$B_{\mathsf{sec}} \coloneqq 4|C|^2 \cdot nB \cdot p_0^2 \cdot (nBL_t + 1). \tag{16}$$

*Consider any $A^{(1)}, \ldots, A^{(\ell)}$ and $B^{(1)}, \ldots, B^{(\ell)}$ such that, for each pair $(A^{(i)}, B^{(i)})$, the difference $A^{(i)} - B^{(i)}$ is a multiple of $P_{\mathcal{I}}$ with $\left|A^{(i)} - B^{(i)}\right| \leqslant B_{\mathsf{sec}} \cdot P_{\mathcal{I}}$. Let*

$$\delta_{\mathsf{sd}} = \frac{32 \cdot \log(\ell B) \cdot \ell^2 \cdot (\ell B)^{2\ell} \cdot |C|^2 \cdot nB \cdot p_0 \cdot (nBL_t + 1) \cdot P_{\max}}{L_{2t}}, \tag{17}$$

*and let $\delta_{\mathsf{lme}} = e^{-\Omega(n)}$ denote the failure probability of the Toeplitz linear mask extractor in Theorem 12; we stress that $\delta_{\mathsf{lme}}$ is a property of the matrix generator and does not decrease with $L_t$ or $L_{2t}$. Then with probability at least $1 - \delta_{\mathsf{lme}}$ over the choice of $\mathbf{E}$, we have:*

$$\mathbf{E} \odot \begin{pmatrix} S_{2t}^{(1)} \\ S_{2t}^{(2)} \\ \vdots \\ S_{2t}^{(n)} \end{pmatrix} + \begin{pmatrix} A^{(1)} \\ A^{(2)} \\ \vdots \\ A^{(\ell)} \end{pmatrix} \approx_{\delta_{\mathsf{sd}}} \mathbf{E} \odot \begin{pmatrix} S_{2t}^{(1)} \\ S_{2t}^{(2)} \\ \vdots \\ S_{2t}^{(n)} \end{pmatrix} + \begin{pmatrix} B^{(1)} \\ B^{(2)} \\ \vdots \\ B^{(\ell)} \end{pmatrix}, \tag{18}$$

*with $S_{2t}^{(i)}$ being fixed for $i \in \mathcal{I}$, and each $S_{2t}^{(i)}$, $i \in \mathcal{H}$, being uniformly distributed over the integers in $[1, p_0 L_{2t}]$ that fall in a fixed residue class modulo $P_{\mathcal{I}}$, i.e., over*

$$s_{2t}^{(i)},\ s_{2t}^{(i)} + P_{\mathcal{I}},\ s_{2t}^{(i)} + 2 \cdot P_{\mathcal{I}},\ \ldots,\ s_{2t}^{(i)} + \left\lfloor \frac{p_0 L_{2t} - s_{2t}^{(i)}}{P_{\mathcal{I}}} \right\rfloor \cdot P_{\mathcal{I}} \tag{19}$$

*for some $s_{2t}^{(i)} \in [1, P_{\mathcal{I}}]$. This is exactly the distribution of $S_{2t}^{(i)} = (p_0 - s^{(i)}) + p_0 v^{(i)}$ for $s^{(i)} \leftarrow \mathbb{Z}_{p_0}$ and $v^{(i)} \leftarrow [0, L_{2t} - 1]$, conditioned on the shares of the corrupted parties, since $(s^{(i)}, v^{(i)}) \mapsto p_0(1 + v^{(i)}) - s^{(i)}$ is a bijection onto $[1, p_0 L_{2t}]$. Moreover, if every $A^{(i)} - B^{(i)}$ is a multiple of $p_0 \cdot P_{\mathcal{I}}$, the same bound holds when the secrets $s^{(i)}$ are fixed and only the offsets $v^{(i)}$, $i \in \mathcal{H}$, are random.*

*Proof.* Let $\mathbf{a} = \begin{pmatrix} A^{(1)} & A^{(2)} & \ldots & A^{(\ell)} \end{pmatrix}^\top$, $\mathbf{b} = \begin{pmatrix} B^{(1)} & B^{(2)} & \ldots & B^{(\ell)} \end{pmatrix}^\top$, and let $\mathbf{s} = \begin{pmatrix} S_{2t}^{(1)} & S_{2t}^{(2)} & \ldots & S_{2t}^{(n)} \end{pmatrix}^\top$. We rewrite the statistical distance to calculate into:

$$\begin{aligned}
&\mathrm{SD}(\mathbf{E} \cdot \mathbf{s} + \mathbf{a},\ \mathbf{E} \cdot \mathbf{s} + \mathbf{b}) && \triangleright \text{ Statistical distance to calculate} \\
&= \mathrm{SD}(\mathbf{E} \cdot \mathbf{s} + (\mathbf{a} - \mathbf{b}),\ \mathbf{E} \cdot \mathbf{s}) && \triangleright \text{ Translation invariance} \\
&= \mathrm{SD}(\mathbf{E} \cdot \mathbf{s} + \mathbf{u} \cdot P_{\mathcal{I}}, \mathbf{E} \cdot \mathbf{s}) && \triangleright \text{ Let } \mathbf{a} - \mathbf{b} = \mathbf{u} \cdot P_{\mathcal{I}} \\
&= \mathrm{SD}\big(\mathbf{E}_{*,\mathcal{I}} \cdot \mathbf{s}_{\mathcal{I}} + \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{s}_{\mathcal{H}} + \mathbf{u} \cdot P_{\mathcal{I}}, \mathbf{E}_{*,\mathcal{I}} \cdot \mathbf{s}_{\mathcal{I}} + \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{s}_{\mathcal{H}}\big) && \triangleright \text{ Note } \mathbf{s}_{\mathcal{I}} \text{ is fixed} \\
&= \mathrm{SD}\big(\mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{s}_{\mathcal{H}} + \mathbf{u} \cdot P_{\mathcal{I}}, \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{s}_{\mathcal{H}}\big) && \triangleright \text{ Translation invariance} \\
&= \mathrm{SD}\big(\mathbf{E}_{*,\mathcal{H}} \cdot (\mathbf{s'}_{\mathcal{H}} + \mathbf{v} \cdot P_{\mathcal{I}}) + \mathbf{u} \cdot P_{\mathcal{I}}, \mathbf{E}_{*,\mathcal{H}} \cdot (\mathbf{s'}_{\mathcal{H}} + \mathbf{v} \cdot P_{\mathcal{I}})\big) && \triangleright \ \mathbf{s}' \text{ fixed, } \mathbf{v} \text{ uniform; see below} \\
&= \mathrm{SD}\big(\mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} \cdot P_{\mathcal{I}} + \mathbf{u} \cdot P_{\mathcal{I}}, \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} \cdot P_{\mathcal{I}}\big) && \triangleright \text{ Translation invariance} \\
&= \mathrm{SD}\big((\mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} + \mathbf{u}) \cdot P_{\mathcal{I}}, \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} \cdot P_{\mathcal{I}}\big) && \\
&= \mathrm{SD}\big(\mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} + \mathbf{u}, \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v}\big)\,. && \triangleright \text{ Scale invariance}
\end{aligned}$$

Here, in the sixth line, we write $\mathbf{s}_{\mathcal{H}} = \mathbf{s'}_{\mathcal{H}} + \mathbf{v} \cdot P_{\mathcal{I}}$, where $\mathbf{s}'$ is fixed by the conditioning on the shares of the corrupted parties and each coordinate of $\mathbf{v}$ is uniform over a range of consecutive integers of length about $p_0 L_{2t} / P_{\mathcal{I}}$.

We now apply Theorem 12 to the last expression, in which the source is $\mathbf{v}$ and the shift to be masked is $\mathbf{u} = (\mathbf{a} - \mathbf{b})/P_{\mathcal{I}}$; note that the factor $P_{\mathcal{I}}$ has been divided out by the scale invariance above, so that $\|\mathbf{u}\|_\infty \leqslant B_{\mathsf{sec}}$ by assumption. Each coordinate of $\mathbf{v}$ is uniform over a range of consecutive integers of length at least $\lfloor p_0 L_{2t} / P_{\mathcal{I}} \rfloor$, and these lengths differ by at most one. Truncating all ranges to the common length $B_{\mathsf{rand}} \coloneqq \lfloor p_0 L_{2t} / P_{\mathcal{I}} \rfloor - 1 \geqslant p_0 L_{2t} / (2P_{\mathcal{I}})$ changes each of the two distributions by at most $nP_{\mathcal{I}} / (p_0 L_{2t})$ in statistical distance, so, by Theorem 12 applied with these $B_{\mathsf{rand}}$ and $B_{\mathsf{sec}}$, with probability at least $1 - \delta_{\mathsf{lme}}$ over the choice of $\mathbf{E}$ we have

$$\begin{aligned}
&\mathrm{SD}\big(\mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v} + \mathbf{u}, \mathbf{E}_{*,\mathcal{H}} \cdot \mathbf{v}\big) \\
&\leqslant \frac{2 \cdot \log(\ell B) \cdot \ell^2 \cdot (\ell B)^{2\ell} \cdot B_{\mathsf{sec}}}{B_{\mathsf{rand}}} + \frac{2nP_{\mathcal{I}}}{p_0 L_{2t}} \\
&\leqslant \frac{16 \log(\ell B)\, \ell^2 (\ell B)^{2\ell} |C|^2\, nB\, p_0 (nBL_t + 1) P_{\mathcal{I}}}{L_{2t}} + \frac{2nP_{\mathcal{I}}}{p_0 L_{2t}} \leqslant \delta_{\mathsf{sd}},
\end{aligned}$$

where the last inequality uses $P_{\mathcal{I}} \leqslant P_{\mathsf{max}}$. If every $A^{(i)} - B^{(i)}$ is a multiple of $p_0 P_{\mathcal{I}}$ and the secrets $s^{(i)}$ are fixed, the same computation applies with $p_0 P_{\mathcal{I}}$ in place of $P_{\mathcal{I}}$ in the scale invariance step: the source $\mathbf{v}$ then has $B_{\mathsf{rand}} \geqslant L_{2t} / (2P_{\mathcal{I}})$ and the shift is bounded by $B_{\mathsf{sec}} / p_0$, which gives the same bound. Finally, by Definition 5 (see also Theorem 7), the failure probability $\delta_{\mathsf{lme}}$ is incurred only once: except with probability $\delta_{\mathsf{lme}}$ over the choice of $\mathbf{E}$, the guarantee holds simultaneously for every batch and, crucially, for every admissible choice of the corrupted set. ∎

We can now conclude the first step.

**Lemma 16** (Static input switching)**.** *Let* Good *denote the event, over the choice of* $\mathbf{E}$*, that* $\mathbf{E}$ *satisfies the structural property of Theorem 7 with the parameters used in the proof of Theorem 12; then* $\Pr[\neg\mathrm{Good}] \leqslant \delta_{\mathsf{lme}}$ *and, conditioned*

*on* Good*, the guarantee of Lemma 15 holds for every batch and every corrupted set simultaneously. For every* $J \subseteq [n]$ *with* $|J| \leqslant t$ *and every input* $\mathbf{x}$*, conditioned on* Good*,*

$$\mathsf{SD}\big(\mathsf{View}_J(\mathsf{Real}(\mathbf{x})),\ \mathsf{View}_J(\mathsf{Exec}_J(x_J, C(\mathbf{x})))\big) \leqslant \varepsilon_{\mathrm{stat}} \coloneqq (|C|+1)\cdot\left(\delta_{\mathrm{sd}} + \frac{4n\cdot P_{\max}}{L_t}\right),$$

*where* $\mathsf{View}_J(\cdot)$ *denotes the joint view (inputs, random tapes, and all messages sent and received) of the parties in* $J$*.*

*Proof.* We exhibit a coupling of $\mathsf{Real}(\mathbf{x})$ and $\mathsf{Exec}_J(x_J, y)$ under which the views of the parties in $J$ coincide except with probability $\varepsilon_{\mathrm{stat}}$; this implies the claimed bound on the statistical distance. The view of the parties in $J$ consists of their inputs and random tapes, of the shares they receive from the parties outside $J$ (their shares of the input sharings $\left[\!\left[X_t^{(i)}\right]\!\right]$ for $i \notin J$ and of the sharings $\left[\!\left[S_t^{(i)}\right]\!\right]$, $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$ of every batch of the pre-processing phase), and of the public values reconstructed in the multiplication gates, in the output masking step, and in the output reconstruction step; all other messages are determined by these. Let $N = \lceil |C|/\ell \rceil + 1 \leqslant |C| + 1$ denote the number of batches of masks (Figure 5), the last of which is used only for the output gate, and for $k \in [N]$ let $\mathbf{m}^{(k)}$ denote the vector of values reconstructed with the masks of batch $k$ (the values $m$ of the multiplication gates and, for $k = N$, the masked output $\tilde{w}$).

*Shares.* The inputs and random tapes of the parties in $J$, as well as all pre-processing sharings, are generated in the same way in both worlds, and we couple them to be equal. By Lemma 14, the shares $\left[\!\left[X_t^{(i)}\right]\!\right]_j$, $j \in J$, of the input of a party $\mathsf{P}_i$ with $i \notin J$ are $(P_{\mathcal{I}}/L_t)$-close to uniform in both worlds, hence $(2P_{\mathcal{I}}/L_t)$-close to each other, and we couple them to be equal except with probability at most $2nP_{\max}/L_t$ in total. From now on, we condition on all these shares, which fixes the residues modulo $P_{\mathcal{I}}$ of all honest sharings, identically in both worlds. Every message sent by a party in $J$, and every share held by a party in $J$, is determined by these shares and by the public reconstructed values, so it remains to couple the sequence $(\mathbf{m}^{(1)}, \ldots, \mathbf{m}^{(N)}, r)$, where $r$ is the value reconstructed in the output reconstruction step.

*Input wires.* Conditioned on its shares for the parties in $J$, the offset $u$ of the input sharing of a party $\mathsf{P}_i$ with $i \notin J$ is uniform over the integers in $[0, L_t - 1]$ in a fixed residue class modulo $P_{\mathcal{I}}$ (a different class in the two worlds). As in the proof of Claim 2, we couple the two offsets so that the input wire values differ by the shift $\Delta \in [0, p_0 P_{\mathcal{I}})$ determined by the CRT. The two offsets then differ by less than $P_{\mathcal{I}}$, and the two progressions of admissible offsets differ in at most one element, so this coupling fails with probability at most $P_{\mathcal{I}}/L_t$ per input, i.e., at most $nP_{\max}/L_t$ in total.

*Batches.* We couple the batches one at a time, maintaining the following invariant after processing batch $k$: the values $\mathbf{m}^{(1)}, \ldots, \mathbf{m}^{(k)}$ coincide in both executions; every wire value computed so far differs between the two executions by a multiple of $P_{\mathcal{I}}$ of magnitude at most $2|C|nBp_0P_{\mathcal{I}}$ (Equation (15)); and the honest randomness of the batches $k+1, \ldots, N$ is still distributed as in Lemma 15 in both executions, independently of everything coupled so far (this holds because the sharings of different batches and the input sharings are sampled independently). The invariant holds for $k = 0$ by the two previous paragraphs and Claim 2.

Consider batch $k$, and let $\mathbf{a}$ and $\mathbf{b}$ denote the unmasked vectors in the real and in the ideal execution, respectively: for a multiplication gate, the corresponding coordinates are the products $X \cdot Y$ of the two input wire values, and for the output gate (in batch $N$) they are $w$ and $w' + \Delta_J$, where $w'$ is the value of the output wire in the ideal execution. By the invariant, each coordinate of $\mathbf{a} - \mathbf{b}$ is a multiple of $P_{\mathcal{I}}$; for a multiplication gate, writing $X' = X - \Delta_X$ and $Y' = Y - \Delta_Y$ for the ideal wire values, we have $XY - X'Y' = \Delta_X Y + X'\Delta_Y$, so, using the bound $W_{\mathrm{add}} \leqslant |C|(nBL_t + 1)p_0$ on the magnitude of a wire value from the correctness analysis, $|XY - X'Y'| \leqslant 2 \cdot 2|C|nBp_0P_{\mathcal{I}} \cdot W_{\mathrm{add}} \leqslant B_{\mathrm{sec}} \cdot P_{\mathcal{I}}$ with $B_{\mathrm{sec}}$ as in Equation (16). For the output gate, $w - (w' + \Delta_J)$ is a multiple of $P_{\mathcal{I}}$ of magnitude at most $(2|C|nBp_0 + p_0)P_{\mathcal{I}} \leqslant B_{\mathrm{sec}}P_{\mathcal{I}}$; moreover, since $w \equiv C(\mathbf{x}) \equiv y \pmod{p_0}$ by correctness and $w' + \Delta_J \equiv y \pmod{p_0}$ by the choice of $\Delta_J$, it is even a multiple of $p_0P_{\mathcal{I}}$. The masks $\mathbf{R}_{2t}^{(k)}$ are extracted from the sharings $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$ of batch $k$, whose honest values are, by the invariant, distributed as in Lemma 15 in both executions, and $\mathbf{m}^{(k)} = \mathbf{a} + \mathbf{R}_{2t}^{(k)}$ in the real execution while $\mathbf{m}^{(k)} = \mathbf{b} + \mathbf{R}_{2t}^{(k)}$ in the ideal one. Conditioned on Good, Lemma 15 gives $\mathsf{SD}\left(\mathbf{a} + \mathbf{R}_{2t}^{(k)}, \mathbf{b} + \mathbf{R}_{2t}^{(k)}\right) \leqslant \delta_{\mathrm{sd}}$, so there is a coupling of the honest sharings $\left[\!\left[S_{2t}^{(i)}\right]\!\right]$ of batch $k$ in the two executions (keeping the shares of the parties in $J$ fixed)

under which $\mathbf{m}^{(k)}$ coincides except with probability $\delta_{\mathsf{sd}}$. For $k < N$, the secrets $s^{(i)}$ of the honest sharings of batch $k$ may differ between the two coupled executions (by less than $p_0$); since $\left[\!\left[ S_t^{(i)} \right]\!\right]$ shares the same secret $s^{(i)}$, we couple its offset as in the proof of Claim 2, so that the integer $S_t^{(i)}$ differs by a multiple of $P_{\mathcal{I}}$ of magnitude less than $2p_0P_{\mathcal{I}}$ while the shares of the parties in $J$ remain fixed; as for the input wires, this fails with probability at most $P_{\mathcal{I}}/L_t$ per honest sharing, i.e., at most $nP_{\mathsf{max}}/L_t$ for the batch. Consequently, the masks $\mathbf{R}_t^{(k)}$ differ by multiples of $P_{\mathcal{I}}$ of magnitude at most $2nBp_0P_{\mathcal{I}}$, and so do the output wires $\bar{m} + [\![R_t]\!]$ of the multiplication gates of batch $k$ (the public values $\bar{m}$ coincide); the output wires of subsequent addition gates differ by sums of at most $|C|$ such shifts, which maintains the invariant. For $k = N$, the shift of the output gate is a multiple of $p_0P_{\mathcal{I}}$, so we use the last part of Lemma 15: the coupling only changes the offsets $v^{(i)}$ and leaves the secrets $s^{(i)}$, hence the sharings $\left[\!\left[ S_t^{(i)} \right]\!\right]$ of batch $N$, the mask $R_t^{(0)}$, and the value $r = R_t^{(0)}$ reconstructed in the output reconstruction step, unchanged.

Summing up, the views coincide except with probability at most $2nP_{\mathsf{max}}/L_t + nP_{\mathsf{max}}/L_t + N \cdot (\delta_{\mathsf{sd}} + nP_{\mathsf{max}}/L_t) \leqslant \varepsilon_{\mathsf{stat}}$. ■

**Step 2: From static to adaptive security.** We now return to the adaptive adversary $\mathcal{A}$ and the simulator $\mathcal{S}$ of Figure 8. Recall that $\mathcal{A}$ is deterministic and corrupts exactly $t$ parties. We describe what $\mathcal{A}$ observes as a sequence of $t$ *blocks*. For $k \in [t]$, let $j_k$ denote the $k$-th corrupted party, let $\tau_k$ denote the point of the execution at which it is corrupted, and recall that $\mathcal{I}_k = \{j_1, \ldots, j_k\}$; we let $\tau_{t+1}$ denote the end of the experiment. The $k$-th block $b_k$ consists of the view of $\mathsf{P}_{j_k}$ up to $\tau_k$ together with the messages received by the parties in $\mathcal{I}_k$ between $\tau_k$ and $\tau_{k+1}$. Since $\mathcal{A}$ is deterministic, $j_k$ and $\tau_k$ are functions of $(b_1, \ldots, b_{k-1})$, and everything $\mathcal{A}$ has observed by the end of the $k$-th block is $(b_1, \ldots, b_k) = \mathsf{View}_{\mathcal{I}_k}^{\leqslant \tau_{k+1}}(e)$, where $e$ is the underlying execution. In the real world, $(b_1, \ldots, b_t)$ is thus a function of a single execution $e \leftarrow \mathsf{Real}(\mathbf{x})$, whereas in the ideal world the $k$-th block is sampled from $\mathsf{Exec}_{\mathcal{I}_k}(x_{\mathcal{I}_k}, y)$ conditioned on $(b_1, \ldots, b_{k-1})$. The output of $\mathcal{A}$ is a function of $(b_1, \ldots, b_t)$, so it suffices to bound the statistical distance between the distributions of this tuple in the two worlds.

**Lemma 17** (From static to adaptive security)**.** *For every deterministic adaptive semi-honest adversary $\mathcal{A}$ corrupting at most $t$ parties and every input $\mathbf{x}$, the statistical distance between the output of $\mathcal{A}$ in the real execution of the protocol and its output in the ideal execution with the simulator of Figure 8 is at most*

$$\delta_{\mathsf{lme}} + 2\sum_{k=1}^{t} \binom{n}{k} \cdot \varepsilon_{\mathsf{stat}} \leqslant \delta_{\mathsf{lme}} + 2^{n+1} \cdot \varepsilon_{\mathsf{stat}}.$$

*Proof.* The event Good of Lemma 16 depends only on $\mathbf{E}$, which is sampled identically in both worlds before any corruption takes place, so it suffices to bound the distance conditioned on Good and to add $\Pr[\neg\mathsf{Good}] \leqslant \delta_{\mathsf{lme}}$. For the rest of the proof, all distributions are implicitly conditioned on Good, and we write Real for $\mathsf{Real}(\mathbf{x})$ and $\mathsf{Exec}_J$ for $\mathsf{Exec}_J(x_J, C(\mathbf{x}))$.

For $m \in \{0, 1, \ldots, t\}$, consider the hybrid experiment $\mathsf{Hyb}_m$ in which the first $t - m$ blocks are generated as in the real world, i.e., $(b_1, \ldots, b_{t-m}) = \mathsf{View}_{\mathcal{I}_{t-m}}^{\leqslant \tau_{t-m+1}}(e)$ for a single execution $e \leftarrow \mathsf{Real}$, and each subsequent block $b_k$, $k > t - m$, is generated as in the ideal world, i.e., by sampling $e_k \leftarrow \mathsf{Exec}_{\mathcal{I}_k}$ conditioned on $\mathsf{View}_{\mathcal{I}_{k-1}}^{\leqslant \tau_k}(e_k) = (b_1, \ldots, b_{k-1})$ and letting $b_k$ be the $k$-th block of $e_k$. Then $\mathsf{Hyb}_0$ is the real world and $\mathsf{Hyb}_t$ is the ideal world, since generating the blocks of a single execution $e \leftarrow \mathsf{Real}$ one at a time is the same as sampling each block from the conditional distribution of Real given the previous blocks.

Fix $m \in [t]$ and let $k \coloneqq t - m + 1$ be the index of the block in which $\mathsf{Hyb}_{m-1}$ and $\mathsf{Hyb}_m$ differ. In both hybrids, the prefix $\mathbf{p} \coloneqq (b_1, \ldots, b_{k-1})$ is distributed as in Real, and the blocks $b_{k+1}, \ldots, b_t$ are generated from $(\mathbf{p}, b_k)$ by the same (ideal-world) procedure. By the data processing inequality for statistical distance, it therefore suffices to bound the distance between the distributions of $(\mathbf{p}, b_k)$ in the two hybrids, which are

$$\Pr_{\mathsf{Real}}[\mathbf{p}, b_k] \quad \text{and} \quad \Pr_{\mathsf{Real}}[\mathbf{p}] \cdot \Pr_{\mathsf{Exec}_{J(\mathbf{p})}}[b_k \mid \mathbf{p}],$$

respectively, where $J(\mathbf{p}) \coloneqq \mathcal{I}_k$ is the set of the first $k$ corrupted parties, which is a function of $\mathbf{p}$. (If $\Pr_{\mathsf{Exec}_{J(\mathbf{p})}}[\mathbf{p}] = 0$, the simulator outputs $\perp$; in this case we let $\Pr_{\mathsf{Exec}_{J(\mathbf{p})}}[\,\cdot \mid \mathbf{p}]$ denote the distribution that puts all its mass on $\perp$.) The

key difficulty is that the adversary may choose $J(\mathbf{p})$ depending on what it has seen, so we cannot directly apply Lemma 16 to a single set $J$. Instead, we partition the prefixes according to $J(\mathbf{p})$: for a set $J$ of size $k$, let $S_J$ be the set of prefixes $\mathbf{p}$ with $J(\mathbf{p}) = J$. The sets $S_J$ are disjoint, and for every $\mathbf{p} \in S_J$ and every $b_k$, the triangle inequality gives

$$\left| \Pr_{\mathsf{Real}}[\mathbf{p}, b_k] - \Pr_{\mathsf{Real}}[\mathbf{p}] \cdot \Pr_{\mathsf{Exec}_J}[b_k \mid \mathbf{p}] \right| \leqslant \left| \Pr_{\mathsf{Real}}[\mathbf{p}, b_k] - \Pr_{\mathsf{Exec}_J}[\mathbf{p}, b_k] \right| + \Pr_{\mathsf{Exec}_J}[b_k \mid \mathbf{p}] \cdot \left| \Pr_{\mathsf{Exec}_J}[\mathbf{p}] - \Pr_{\mathsf{Real}}[\mathbf{p}] \right|.$$

Summing over $b_k$ and over $\mathbf{p} \in S_J$, and then over $J$, we obtain

$$\mathsf{SD}\big(\mathsf{Hyb}_{m-1}, \mathsf{Hyb}_m\big) \leqslant \sum_{J:|J|=k} \left( \frac{1}{2} \sum_{\mathbf{p} \in S_J,\, b_k} \left| \Pr_{\mathsf{Real}}[\mathbf{p}, b_k] - \Pr_{\mathsf{Exec}_J}[\mathbf{p}, b_k] \right| + \frac{1}{2} \sum_{\mathbf{p} \in S_J} \left| \Pr_{\mathsf{Real}}[\mathbf{p}] - \Pr_{\mathsf{Exec}_J}[\mathbf{p}] \right| \right). \tag{20}$$

Now fix $J$ with $|J| = k$, and consider the following function $g_J$ of an execution $e$: run $\mathcal{A}$ on $e$, revealing the views of the corrupted parties as requested; as soon as $\mathcal{A}$ asks to corrupt a party outside $J$ before its $k$-th corruption, output $\perp$; otherwise, output $(\mathbf{p}, b_k)$. Since the choices of $\mathcal{A}$ are determined by the views of the parties it has corrupted so far, $g_J$ is a function of $\mathsf{View}_J(e)$ alone, and $g_J(e) = (\mathbf{p}, b_k)$ exactly when $\mathbf{p} \in S_J$. Hence the first sum over $S_J$ in Equation (20) is bounded by $\mathsf{SD}\big(g_J(\mathsf{Real}), g_J(\mathsf{Exec}_J)\big)$, and so is the second one, since $\mathbf{p}$ is a function of $(\mathbf{p}, b_k)$. By the data processing inequality and Lemma 16, both are therefore at most $\mathsf{SD}\big(\mathsf{View}_J(\mathsf{Real}), \mathsf{View}_J(\mathsf{Exec}_J)\big) \leqslant \varepsilon_{\mathsf{stat}}$. Summing over the at most $\binom{n}{k}$ sets $J$ of size $k$ yields $\mathsf{SD}\big(\mathsf{Hyb}_{m-1}, \mathsf{Hyb}_m\big) \leqslant 2\binom{n}{k}\varepsilon_{\mathsf{stat}}$, and summing over $m \in [t]$ concludes the proof. ∎

**Concluding the proof of Theorem 13.** By Lemma 17, the protocol is $\big(\delta_{\mathsf{lme}} + 2^{n+1} \cdot \varepsilon_{\mathsf{stat}}\big)$-statistically secure against adaptive semi-honest adversaries corrupting at most $t$ parties, where $\delta_{\mathsf{lme}} = e^{-\Omega(n)}$ is the failure probability of Theorem 12 and $\varepsilon_{\mathsf{stat}} = (|C| + 1)(\delta_{\mathsf{sd}} + 4nP_{\max}/L_t)$ with $\delta_{\mathsf{sd}}$ as in Equation (17). Using $|C| + 1 \leqslant 2|C|$, $\ell \leqslant n$, and $nBL_t + 1 \leqslant 2nBL_t$, the first condition on $L_{2t}$ in Theorem 13 ensures that $2^{n+1}(|C| + 1) \cdot \delta_{\mathsf{sd}} \leqslant 2^{-\kappa-2}$, and the second condition on $L_t$ ensures that $2^{n+1}(|C|+1) \cdot 4nP_{\max}/L_t \leqslant 2^{-\kappa-2}$. Hence the total error is at most $2^{-\kappa-1} + \delta_{\mathsf{lme}} \leqslant 2^{-\kappa} + \delta_{\mathsf{lme}}$, which completes the proof.

We remark that the factor $2^{n+1}$ stems from the union bound over the corrupted sets in the proof of Lemma 17 and only affects the statistical distance terms, which decay with $L_t$ and $L_{2t}$; it costs $n + 1$ additional bits in $\log L_t$ and $\log L_{2t}$, which is absorbed in the $O(n \log n)$ slack of the parameter analysis following Theorem 13. In contrast, the failure probability $\delta_{\mathsf{lme}}$ of the mask extractor is incurred only once, because Definition 5 guarantees that a single sampled matrix is good for all corrupted sets simultaneously; this is where the adaptive guarantee of Theorem 12 is used. Note that $\delta_{\mathsf{lme}}$ cannot be made small by increasing $L_t$ or $L_{2t}$: it is $e^{-\Omega(n)}$ by Theorem 12, and it is below $2^{-\kappa}$ as soon as $n \geqslant c_0\kappa$ for a sufficiently large constant $c_0$, which is the regime of Corollary 1.

**AI Disclosure.** ChatGPT-6 Astra was used to write the Python scripts for computing the concrete parameters reported in Table 1. AI tools were also used to polish the text throughout the paper.

**Acknowledgments.** The authors would like to thank Yuval Ishai for helpful discussions and for bringing [CDdH07] to our attention. Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing. Aditya Hegde was supported in part by NSF CAREER Award 1942789 and a Johns Hopkins University Catalyst Award. Phuoc Van Long Pham is supported in part by NSF CAREER Award 2442384. Mingyuan Wang is supported in part by a research grant from Open Philanthropy.

# References

[AB83] Charles Asmuth and John Bloom. A modular approach to key safeguarding. *IEEE Trans. Inf. Theory*, 29(2):208–210, 1983. 6, 16

[AIK11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In Rafail Ostrovsky, editor, *52nd Annual Symposium on Foundations of Computer Science*, pages 120–129, Palm Springs, CA, USA, October 22–25, 2011. IEEE Computer Society Press. doi:10.1109/FOCS.2011.40. 6

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501, Cambridge, UK, April 15–19, 2012. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-29011-4_29. 17

[BBMHS22] Carsten Baum, Lennart Braun, Alexander Munch-Hansen, and Peter Scholl. Moz$\mathbb{Z}_{2^k}$arella: Efficient vector-OLE and zero-knowledge proofs over $\mathbb{Z}_{2^k}$. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 329–358, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-15985-5_12. 3

[Ber04] Daniel J. Bernstein. Scaled remainder trees. Technical note, 2004. URL: https://cr.yp.to/arith/scaledmod-20040820.pdf. 16

[BGJK21] Gabrielle Beck, Aarushi Goel, Abhishek Jain, and Gabriel Kaptchuk. Order-C secure multiparty computation for highly repetitive circuits. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 663–693, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-77886-6_23. 3, 4, 5

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 1–10, Chicago, IL, USA, May 2–4, 1988. ACM Press. doi:10.1145/62212.62213. 3

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 11–19, Chicago, IL, USA, May 2–4, 1988. ACM Press. doi:10.1145/62212.62214. 3

[CDdH07] Ronald Cramer, Ivan Damgård, and Robbert de Haan. Atomic secure multi-party multiplication with low communication. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 329–346, Barcelona, Spain, May 20–24, 2007. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-540-72540-4_19. 3, 48

[CDE+18] Ronald Cramer, Ivan Damgård, Daniel Escudero, Peter Scholl, and Chaoping Xing. SPD $\mathbb{Z}_{2^k}$: Efficient MPC mod $2^k$ for dishonest majority. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 769–798, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Cham, Switzerland. doi:10.1007/978-3-319-96881-0_26. 3

[CKL21] Jung Hee Cheon, Dongwoo Kim, and Keewoo Lee. MHz2k: MPC from HE over $\mathbb{Z}_{2^k}$ with new packing, simpler reshare, and better ZKP. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 426–456, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-84245-1_15. 3

[CLMZ23] Alessandro Chiesa, Ryan Lehmkuhl, Pratyush Mishra, and Yinuo Zhang. Eos: Efficient private delegation of zkSNARK provers. In Joseph A. Calandrino and Carmela Troncoso, editors, *USENIX Security 2023: 32nd USENIX Security Symposium*, pages 6453–6469, Anaheim, CA, USA, August 9–11, 2023. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/chiesa. 4

[CRX21] Ronald Cramer, Matthieu Rambaud, and Chaoping Xing. Asymptotically-good arithmetic secret sharing over $\mathbb{Z}/p^\ell\mathbb{Z}$ with strong multiplication and its applications to efficient MPC. In Tal Malkin and

Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 656–686, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-84252-9_22. 3, 5

[DIK10] Ivan Damgård, Yuval Ishai, and Mikkel Krøigaard. Perfectly secure multiparty computation and the computational overhead of cryptography. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 445–465, French Riviera, May 30 – June 3, 2010. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-13190-5_23. 3, 4, 5

[DLN19] Ivan Damgård, Kasper Green Larsen, and Jesper Buus Nielsen. Communication lower bounds for statistically secure MPC, with or without preprocessing. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 61–84, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland. doi:10.1007/978-3-030-26951-7_3. 4

[DN07] Ivan Damgård and Jesper Buus Nielsen. Scalable and unconditionally secure multiparty computation. In Alfred Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 572–590, Santa Barbara, CA, USA, August 19–23, 2007. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-540-74143-5_32. 7

[EXY22] Daniel Escudero, Chaoping Xing, and Chen Yuan. More efficient dishonest majority secure computation over $\mathbb{Z}_{2^k}$ via Galois rings. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 383–412, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland. doi:10.1007/978-3-031-15802-5_14. 3

[FK23] Dankrad Feist and Dmitry Khovratovich. Fast amortized KZG proofs. Cryptology ePrint Archive, Report 2023/033, 2023. URL: https://eprint.iacr.org/2023/033. 11, 33

[FY92] Matthew K. Franklin and Moti Yung. Communication complexity of secure computation (extended abstract). In *24th Annual ACM Symposium on Theory of Computing*, pages 699–710, Victoria, BC, Canada, May 4–6, 1992. ACM Press. doi:10.1145/129712.129780. 3, 4, 5

[GGJ+23] Sanjam Garg, Aarushi Goel, Abhishek Jain, Guru-Vamsi Policharla, and Sruthi Sekar. zkSaaS: Zero-knowledge SNARKs as a service. In Joseph A. Calandrino and Carmela Troncoso, editors, *USENIX Security 2023: 32nd USENIX Security Symposium*, pages 4427–4444, Anaheim, CA, USA, August 9–11, 2023. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity23/presentation/garg. 4

[GGJ+25] Sanjam Garg, Aarushi Goel, Abhishek Jain, Bhaskar Roberts, and Sruthi Sekar. Malicious security in collaborative zk-SNARKs: More than meets the eye. In *Advances in Cryptology – CRYPTO 2025, Part VII*, Lecture Notes in Computer Science, pages 396–428, Santa Barbara, CA, USA, August 2025. Springer, Cham, Switzerland. doi:10.1007/978-3-032-01907-3_13. 4

[GJM+23] Sanjam Garg, Abhishek Jain, Pratyay Mukherjee, Rohit Sinha, Mingyuan Wang, and Yinuo Zhang. Cryptography with weights: MPC, encryption and signatures. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 295–327, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland. doi:10.1007/978-3-031-38557-5_10. 6, 16

[GJMW24] Sanjam Garg, Abhishek Jain, Pratyay Mukherjee, and Mingyuan Wang. Scalable multiparty computation from non-linear secret sharing. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VIII*, volume 14927 of *Lecture Notes in Computer Science*, pages 384–417, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-68397-8_12. 3, 4, 5, 6, 8, 9, 16, 19, 36, 43

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, NY, USA, May 25–27, 1987. ACM Press. doi:10.1145/28395.28420. 3

[Gol04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004. doi:10.1017/CBO9780511721656. 15

[GRS99] Oded Goldreich, Dana Ron, and Madhu Sudan. Chinese remaindering with errors. In *31st Annual ACM Symposium on Theory of Computing*, pages 225–234, Atlanta, GA, USA, May 1–4, 1999. ACM Press. doi:10.1145/301250.301309. 6, 16

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th Annual ACM Symposium on Theory of Computing*, pages 21–30, San Diego, CA, USA, June 11–13, 2007. ACM Press. doi:10.1145/1250790.1250794. 3

[LED+24] Yun Li, Daniel Escudero, Yufei Duan, Zhicong Huang, Cheng Hong, Chao Zhang, and Yifan Song. Sublinear distributed product checks on replicated secret-shared data over $Z_{2^k}$ without ring extensions. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024: 31st Conference on Computer and Communications Security*, pages 825–839, Salt Lake City, UT, USA, October 14–18, 2024. ACM Press. doi:10.1145/3658644.3690260. 3

[Mig83] Maurice Mignotte. How to share a secret? In Thomas Beth, editor, *Advances in Cryptology – EUROCRYPT'82*, volume 149 of *Lecture Notes in Computer Science*, pages 371–375, Burg Feuerstein, Germany, March 29 – April 2, 1983. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-39466-4_27. 6, 16

[OB22] Alex Ozdemir and Dan Boneh. Experimenting with collaborative zk-SNARKs: Zero-knowledge proofs for distributed secrets. In Kevin R. B. Butler and Kurt Thomas, editors, *USENIX Security 2022: 31st USENIX Security Symposium*, pages 4291–4308, Boston, MA, USA, August 10–12, 2022. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/ozdemir. 4

[SS71] Arnold Schönhage and Volker Strassen. Schnelle multiplikation grosser zahlen. *Computing*, 7(3):281–292, 1971. 33, 41

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, October 27–29, 1986. IEEE Computer Society Press. doi:10.1109/SFCS.1986.25. 3, 6