Practical Bootstrapping in Quasilinear Time[⋆]

Jacob Alperin-Sheriff and Chris Peikert

School of Computer Science, Georgia Institute of Technology

Abstract. Gentry's "bootstrapping" technique (STOC 2009) constructs a fully homomorphic encryption (FHE) scheme from a "somewhat homomorphic" one that is powerful enough to evaluate its own decryption function. To date, it remains the only known way of obtaining unbounded FHE. Unfortunately, bootstrapping is computationally very expensive, despite the great deal of effort that has been spent on improving its efficiency. The current state of the art, due to Gentry, Halevi, and Smart (PKC 2012), is able to bootstrap "packed" ciphertexts (which encrypt up to a linear number of bits) in time only *quasilinear* $\tilde{O}(\lambda) = \lambda \cdot \log^{O(1)} \lambda$ in the security parameter. While this performance is *asymptotically* optimal up to logarithmic factors, the practical import is less clear: the procedure composes multiple layers of expensive and complex operations, to the point where it appears very difficult to implement, and its concrete runtime appears worse than those of prior methods (all of which have quadratic or larger asymptotic runtimes).

In this work we give *simple*, *practical*, and entirely *algebraic* algorithms for bootstrapping in quasilinear time, for both "packed" and "non-packed" ciphertexts. Our methods are easy to implement (especially in the non-packed case), and we believe that they will be substantially more efficient in practice than all prior realizations of bootstrapping. One of our main techniques is a substantial enhancement of the "ring-switching" procedure of Gentry et al. (SCN 2012), which we extend to support switching between two rings where neither is a subring of the other. Using this procedure, we give a natural method for homomorphically evaluating a broad class of structured linear transformations, including one that lets us evaluate the decryption function efficiently.

1 Introduction

Bootstrapping, a central technique from the breakthrough work of Gentry [8, 7] on fully homomorphic encryption (FHE), converts a sufficiently powerful "somewhat homomorphic" encryption (SHE) scheme into a fully homomorphic one.

[⋆] This material is based upon work supported by the National Science Foundation under CAREER Award CCF-1054495, by the Alfred P. Sloan Foundation, and by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under Contract No. FA8750-11-C-0098. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the National Science Foundation, the Sloan Foundation, DARPA or the U.S. Government.

(An SHE scheme can support a bounded number of homomorphic operations on freshly generated ciphertexts, whereas an FHE scheme has no such bound.) In short, bootstrapping works by *homomorphically* evaluating the SHE scheme's decryption function on a ciphertext that cannot support any further homomorphic operations. This has the effect of "refreshing" the ciphertext, i.e., it produces a new one that encrypts the same message and can handle more homomorphic operations. Bootstrapping remains the only known way to achieve *unbounded* FHE, i.e., a scheme that can homomorphically evaluate any efficient function using keys and ciphertexts of a fixed size.[1]

In order to be "bootstrappable," an SHE scheme must be powerful enough to homomorphically evaluate its own decryption function, using whatever homomorphic operations it supports. For security reasons, the key and ciphertext sizes of all known SHE schemes grow with the *depth* and, to a lesser extent, the *size* of the functions that they can homomorphically evaluate. For instance, under plausible hardness conjectures, the key and ciphertext sizes of the most efficient SHE scheme to date [3] grow quasilinearly in both the supported multiplicative depth d and the security parameter λ, i.e., as $\tilde{O}(d \cdot \lambda)$. Clearly, the runtime of bootstrapping must also grow with the sizes of the keys, ciphertexts, and decryption function. This runtime is perhaps the most important measure of efficiency for FHE, because bootstrapping is currently the biggest bottleneck by far in instantiations, both in theory and in practice.

The past few years have seen an intensive study of different forms of decryption procedures for SHE schemes, and their associated bootstrapping operations [8, 7, 18, 10, 4, 9, 3, 13]. The first few bootstrapping methods had moderate polynomial runtimes in the security parameter λ, e.g., $\tilde{O}(\lambda^4)$. Brakerski, Gentry, and Vaikuntanathan [3] gave a major efficiency improvement, reducing the runtime to $\tilde{O}(\lambda^2)$. They also gave an amortized method that bootstraps $\tilde{\Omega}(\lambda)$ ciphertexts at once in $\tilde{O}(\lambda^2)$ time, i.e., quasilinear runtime per ciphertext. However, these results apply only to "non-packed" ciphertexts, i.e., ones that encrypt essentially just one bit each, which combined with the somewhat large runtimes makes these methods too inefficient to be used very much in practice. Most recently, Gentry, Halevi, and Smart [12] achieved bootstrapping for "packed" ciphertexts (i.e., ones that encrypt up to $\tilde{\Omega}(\lambda)$ bits each) in *quasilinear* $\tilde{O}(\lambda)$ runtime, which is asymptotically optimal in space and time, up to polylogarithmic factors. For this they relied on a general "compiler" from another work of theirs [13], which achieved SHE/FHE for sufficiently wide circuits with polylogarithmic multiplicative "overhead," i.e., cost relative to evaluating the circuit "in the clear."

Bootstrapping and FHE in quasi-optimal time and space is a very attractive and powerful theoretical result. However, the authors of [13, 12] caution that their constructions may have limited potential for use in practice, for two main

[1] This stands in contrast with *leveled* FHE schemes, which can homomorphically evaluate a function of any *a priori* bounded depth, but using keys and ciphertexts whose sizes depend on the bound. Leveled FHE can be constructed without resorting to bootstrapping [3].

reasons: first, the runtimes, while asymptotically quasilinear, include very large polylogarithmic factors. For realistic values of the security parameter, these polylogarithmic terms exceed the rather small (but asymptotically worse) quasilinear overhead obtained in [3]. The second reason is that their bootstrapping operation is algorithmically very complex and difficult to implement (see the next paragraphs for details). Indeed, while there are now a few working implementations of bootstrapping (e.g., [10, 6]) that follow the templates from [8, 7, 18, 3], we are not aware of any attempt to implement any method having subquadratic runtime.

Is quasilinear efficient? The complexity and large practical overhead of the constructions in [13, 12] arise from two kinds of operations. First, the main technique from [13] is a way of homomorphically evaluating any sufficiently shallow and wide arithmetic circuit on a "packed" ciphertext that encrypts a high-dimensional vector of plaintexts in multiple "slots." It works by first using ring automorphisms and key-switching operations [4, 3] to obtain a small, fixed set of "primitive" homomorphic permutations on the slots. It then composes those permutations (along with other homomorphic operations) in a log-depth permutation network, to obtain any permutation. Finally, it homomorphically evaluates the desired circuit by combining appropriate permutations with relatively simple homomorphic slot-selection and ring operations.

In the context of bootstrapping, one of the key observations from [12] is that a main step of the decryption procedure can be evaluated using the above technique. Specifically, they need an operation that moves the coefficients of an encrypted plaintext polynomial, reduced modulo a cyclotomic polynomial $\Phi_m(X)$, into the slots of a packed ciphertext (and back again). Once the coefficients are in the slots, they can be rounded in a batched (SIMD) fashion, and then mapped back to coefficients of the plaintext. The operations that move the coefficients into slots and vice-versa can be expressed as $O(\log \lambda)$-depth arithmetic circuits of size $O(\lambda \log \lambda)$, roughly akin to the classic FFT butterfly network. Hence they can be evaluated homomorphically with polylogarithmic overhead, using [13]. However, as the authors of [12] point out, the decryption circuit is quite large and complex – especially the part that moves the slots back to the coefficients, because it involves reduction modulo $\Phi_m(X)$ for an m having several prime divisors. This modular reduction is the most expensive part of the decryption circuit, and avoiding it is one of the main open problems given in [12]. However, even a very efficient decryption circuit would still incur the large polylogarithmic overhead factors from the techniques of [13].

1.1 Our Contributions

We give a new bootstrapping algorithm that runs in *quasilinear* $\tilde{O}(\lambda)$ time per ciphertext with *small* polylogarithmic factors, and is algorithmically much simpler than previous methods. It is easy to implement, and we believe that it will be substantially more efficient in practice than all prior methods. We provide a unified bootstrapping procedure that works for both "non-packed" ciphertexts (which encrypt integers modulo some p, e.g., bits) and "packed" ciphertexts (which encrypt elements of a high-dimensional ring), and also interpolates between the two cases to handle an intermediate concept we call "semi-packed" ciphertexts.

Our procedure for non-packed ciphertexts is especially simple and efficient. In particular, it can work very naturally using only cyclotomic rings having power-of-two index, i.e., rings of the form $\mathbb{Z}[X]/(1 + X^{2^k})$, which admit very fast implementations. This improves upon the method of [3], which achieves quasilinear *amortized* runtime when bootstrapping $\tilde{\Omega}(\lambda)$ non-packed ciphertexts at once. Also, while that method can also use power-of-two cyclotomics, it can only do so by emulating $\mathbb{Z}_2$ (bit) arithmetic within $\mathbb{Z}_p$ for some moderately large prime p, which translates additions in $\mathbb{Z}_2$ into much more costly multiplications in $\mathbb{Z}_p$. By contrast, our method works "natively" with any plaintext modulus.

For packed ciphertexts, our procedure draws upon high-level ideas from [13, 12], but our approach is conceptually and technically very different. Most importantly, it completely avoids the two main inefficiencies from those works: first, unlike [13], we do not use permutation networks or any explicit permutations of the plaintext slots, nor do we rely on a general-purpose compiler for homomorphically evaluating arithmetic circuits. Instead, we give direct, practically efficient procedures for homomorphically mapping the coefficients of an encrypted plaintext element into slots and vice-versa. In particular, our procedure does not incur the large cost or algorithmic complexity of homomorphically reducing modulo $\Phi_m(X)$, which was the main bottleneck in the decryption circuit of [12].

At a higher level, our bootstrapping method has two other attractive and novel features: first, it is entirely "algebraic," by which we mean that the full procedure (including generation of all auxiliary data it uses) can be described as a short sequence of elementary operations from the "native instruction set" of the SHE scheme. By contrast, all previous methods at some point invoke rather generic arithmetic circuits, e.g., for modular addition of values represented as bit strings, or reduction modulo a cyclotomic polynomial $\Phi_m(X)$. Of course, arithmetic circuits can be evaluated using the SHE scheme's native operations, but we believe that the distinction between "algebraic" and "non-algebraic" is an important qualitative one, and it certainly affects the simplicity and concrete efficiency of the bootstrapping procedure.

The second nice feature of our method is that it completely decouples the algebraic structure of the SHE plaintext ring from that which is needed by the bootstrapping procedure. In previous methods that use amortization (or "batching") for efficiency (e.g., [17, 3, 12]), the ring and plaintext modulus of the SHE scheme must be chosen so as to provide many plaintext slots. However, this structure may not always be a natural match for the SHE application's efficiency or functionality requirements. For example, the lattice-based pseudorandom function of [1] works very well with a ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ where both q and n are powers of two, but for such parameters R_q has only *one* slot. Our method can bootstrap even for this kind of plaintext ring (and many others), while still using batching to achieve quasilinear runtime.

1.2 Techniques

At the heart of our bootstrapping procedure are two novel homomorphic operations for SHE schemes over cyclotomic rings: for non-packed (or semi-packed) ciphertexts, we give an operation that *isolates the message-carrying coefficient(s)* of a high-dimensional ring element; and for (semi-)packed ciphertexts, we give an operation that *maps coefficients to slots* and vice-versa.

Isolating coefficients. Our first homomorphic operation is most easily explained in the context of non-packed ciphertexts, which encrypt single elements of the quotient ring $\mathbb{Z}_p$ for some small modulus p, using ciphertexts over some cyclotomic quotient ring $R_q = R/qR$ of moderately large degree $d = \deg(R/\mathbb{Z}) = \tilde{O}(\lambda)$. We first observe that a ciphertext to be bootstrapped can be reinterpreted as an encryption of an R_q-element, one of whose $\mathbb{Z}_q$-coefficients (with respect to an appropriate basis of the ring) "noisily" encodes the message, and whose other coefficients are just meaningless noise terms. We give an simple and efficient homomorphic operation that preserves the meaningful coefficient, and maps all the others to zero. Having isolated the message-encoding coefficient, we can then homomorphically apply an efficient integer "rounding" function (see [12] and the full version) to recover the message from its noisy encoding, which completes the bootstrapping procedure. (Note that it is necessary to remove the meaningless noise coefficients first, otherwise they would interfere with the correct operation of the rounding function.)

Our coefficient-isolating procedure works essentially by applying the *trace function* $\mathrm{Tr}_{R/\mathbb{Z}} \colon R \to \mathbb{Z}$ to the plaintext. The trace is the "canonical" $\mathbb{Z}$-linear function from R to $\mathbb{Z}$, and it turns out that for the appropriate choice of $\mathbb{Z}$-basis of R used in decryption, the trace simply outputs (up to some scaling factor) the message-carrying coefficient we wish to isolate. One simple and very efficient way of applying the trace homomorphically is to use the "ring-switching" technique of [11], but unfortunately, this requires the ring-LWE problem [15] to be hard over the target ring $\mathbb{Z}$, which is clearly not the case. Another way follows from the fact that $\mathrm{Tr}_{R/\mathbb{Z}}$ equals the sum of all d automorphisms of R; therefore, it can be computed by homomorphically applying each automorphism and summing the results. Unfortunately, this method takes at least *quadratic* $\Omega(\lambda^2)$ time, because applying each automorphism homomorphically takes $\Omega(\lambda)$ time, and there are $d = \Omega(\lambda)$ automorphisms.

So, instead of inefficiently computing the trace by summing all the automorphisms at once, we consider a *tower* of cyclotomic rings $\mathbb{Z} = R^{(0)} \subseteq R^{(1)} \subseteq \cdots \subseteq R^{(r)} = R$, usually written as $R^{(r)}/\cdots/R^{(1)}/R^{(0)}$. Then $\mathrm{Tr}_{R/\mathbb{Z}}$ is the composition of the individual trace functions $\mathrm{Tr}_{R^{(i)}/R^{(i-1)}} \colon R^{(i)} \to R^{(i-1)}$, and these traces are equal to the sums of all automorphisms of $R^{(i)}$ that fix $R^{(i-1)}$ pointwise, of which there are exactly $d_i = \deg(R^{(i)}/R^{(i-1)}) = \deg(R^{(i)}/\mathbb{Z})/\deg(R^{(i-1)}/\mathbb{Z})$. We can therefore compute each $\mathrm{Tr}_{R^{(i)}/R^{(i-1)}}$ in time linear in λ and in d_i; moreover, the number of trace functions to apply is at most logarithmic in $d = \deg(R/\mathbb{Z}) = \tilde{O}(\lambda)$, because each one reduces the degree by a factor of at least two. Therefore, by ensuring that the degrees of $R^{(r)}, R^{(r-1)}, \ldots, R^{(0)}$

decrease gradually enough, we can homomorphically apply the full $\mathrm{Tr}_{R/\mathbb{Z}}$ in quasilinear time. For example, a particularly convenient choice is to let $R^{(i)}$ be the 2^{i+1}st cyclotomic ring $\mathbb{Z}[X]/(1+X^{2^i})$ of degree 2^i, so that every $d_i = 2$, and there are exactly $\log_2(d) = O(\log \lambda)$ trace functions to apply.

More generally, when bootstrapping a *semi-packed* ciphertext we start with a plaintext value in R_q that noisily encodes a message in S_p, for some subring $S \subseteq R$. (The case $S = \mathbb{Z}$ corresponds to a non-packed ciphertext.) We show that applying the trace function $\mathrm{Tr}_{R/S}$ to the R_q-plaintext yields a new plaintext *in* S_q that noisily encodes the message, thus isolating the meaningful part of the noisy encoding and vanishing the rest. We then homomorphically apply a rounding function to recover the S_p message from its noisy S_q encoding, which uses the technique described next.

Mapping coefficients to slots. Our second technique, and main technical innovation, is in bootstrapping (semi-)packed ciphertexts. We enhance the recent "ring-switching" procedure of [11], and use it to efficiently move "noisy" plaintext coefficients (with respect to an appropriate decryption basis) into slots for batch-rounding, and finally move the rounded slot values back to coefficients. We note that all previous methods for loading plaintext data into slots used the *same* ring for the source and destination, and so required the plaintext to come from a ring designed to have many slots. In this work, we use ring-switching to go from the SHE plaintext ring to a *different* ring having many slots, which is used only temporarily for batch-rounding. This is what allows the SHE plaintext ring to be decoupled from the rings used in bootstrapping, as mentioned above.

To summarize our technique, we first recall the ring-switching procedure of [11]. It was originally devised to provide moderate efficiency gains for SHE/FHE schemes, by allowing them to switch ciphertexts from high-degree cyclotomic rings to *subrings* of smaller degree (once enough homomorphic operations have been performed to make this secure). We generalize the procedure, showing how to switch between two rings where neither ring need be a subring of the other. The procedure has a very simple implementation, and as long as the two rings have a large *common subring*, it is also very efficient (e.g., quasilinear in the dimension). Moreover, it supports, as a side effect, the homomorphic evaluation of any function that is *linear over the common subring*. However, the larger the common subring is, the more restrictive this condition on the function becomes.

We show how our enhanced ring-switching can move the plaintext coefficients into the slots of the target ring (and back), which can be seen as just evaluating a certain $\mathbb{Z}$-linear function. Here we are faced with the main technical challenge: for efficiency, the common subring of the source and destination rings must be large, but then the supported class of linear functions is very restrictive, and certainly does not include the $\mathbb{Z}$-linear one we want to evaluate. We solve this problem by switching through a short sequence of "hybrid" rings, where adjacent rings have a large common subring, but the initial and final rings have only the integers $\mathbb{Z}$ in common. Moreover, we show that for an appropriately chosen sequence of hybrid rings, the $\mathbb{Z}$-linear function we want to evaluate *is* realizable by a sequence of allowed linear functions between adjacent hybrid rings. Very critically, this decomposition requires the SHE scheme to use a *highly structured* basis of the ring for decryption. The usual representation of a cyclotomic ring as $\mathbb{Z}[X]/\Phi_m(X)$ typically does not correspond to such a basis, so we instead rely on the *tensorial decomposition* of the ring and its corresponding bases, as recently explored in [16]. At heart, this is what allows us to avoid the expensive homomorphic reduction modulo $\Phi_m(X)$, which is one of the main bottlenecks in previous work [12].[2]

Stepping back a bit, the technique of switching through hybrid rings and bases is reminiscent of standard "sparse decompositions" for linear transformations like the FFT, in that both decompose a complicated high-dimensional transform into a short sequence of simpler, structured transforms. (Here, the simple transforms are computed merely as a side-effect of passing through the hybrid rings.) Because of these similarities, we believe that the enhanced ring-switching procedure will be applicable in other domain-specific applications of homomorphic encryption, e.g., signal-processing transforms or statistical analysis.

Organization. Due to space restrictions, this version of the paper omits much of the algebraic background, several proofs, and some lower-level descriptions of our procedures; see the full version for complete details. Section 2.1 recalls some of the algebraic background required for our constructions, and Section 2.2 recalls a standard ring-based SHE scheme and some of its natural homomorphic operations. Section 3 defines the general bootstrapping procedure. Sections 4 and 5 respectively fill in further details of the two novel homomorphic operations used in the bootstrapping procedure.

The full version also documents a folklore transformation between two essentially equivalent ways of encoding messages in SHE schemes (namely, the "least/most significant bit" encodings), describes an integer rounding procedure that simplifies the one given in [12], and gives some concrete choices of rings that our method can use in practice.

Acknowledgments. We thank Oded Regev for helpful discussions during the early stages of this research, and the anonymous CRYPTO'13 reviewers for their thoughtful comments.

2 Preliminaries

For a positive integer k, we let $[k] = \{0, \ldots, k-1\}$. For an integer modulus q, we let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ denote the quotient ring of integers modulo q. For integers q, q', we define the integer "rounding" function $\lfloor \cdot \rceil_{q'} : \mathbb{Z}_q \to \mathbb{Z}_{q'}$ as $\lfloor x \rceil_{q'} = \lfloor (q'/q) \cdot x \rceil \bmod q'$.

[2] The use of more structured representations of cyclotomic rings in [16] was initially motivated by the desire for simpler and more efficient algorithms for cryptographic operations. Interestingly, these representations yield moderate efficiency improvements for computations "in the clear," but dramatic benefits for their homomorphic counterparts!

2.1 Algebraic Background

Throughout this work, by "ring" we mean a commutative ring with identity. For two rings $R \subseteq R'$, an R-basis of R' is a set $B \subset R'$ such that every $r \in R'$ can be written uniquely as an R-linear combination of elements of B. For two rings R, S with a common subring E, an E-linear function $L \colon R \to S$ is one for which $L(r + r') = L(r) + L(r')$ for all $r, r' \in R$, and $L(e \cdot r) = e \cdot L(r)$ for all $e \in E, r \in R$. It is immediate that such a function is defined uniquely by its values on any E-basis of R.

Cyclotomic Rings. For a positive integer m called the *index*, let $\mathcal{O}_m = \mathbb{Z}[\zeta_m]$ denote the mth *cyclotomic ring*, where ζ_m is an abstract element of order m over $\mathbb{Q}$. (In particular, we do not view ζ_m as any particular complex root of unity.) The minimal polynomial of ζ_m over $\mathbb{Q}$ is the mth *cyclotomic polynomial* $\Phi_m(X) = \prod_{i \in \mathbb{Z}_m^*} (X - \omega_m^i) \in \mathbb{Z}[X]$, where $\omega_m = \exp(2\pi\sqrt{-1}/m) \in \mathbb{C}$ is the principal mth complex root of unity, and the roots $\omega_m^i \in \mathbb{C}$ range over all the *primitive* complex mth roots of unity. Therefore, $\mathcal{O}_m$ is a ring extension of degree $n = \varphi(m)$ over $\mathbb{Z}$. (In particular, $\mathcal{O}_1 = \mathcal{O}_2 = \mathbb{Z}$.) Clearly, $\mathcal{O}_m$ is isomorphic to the polynomial ring $\mathbb{Z}[X]/\Phi_m(X)$ by identifying ζ_m with X, and has the "power basis" $\{1, \zeta_m, \ldots, \zeta_m^{n-1}\}$ as a $\mathbb{Z}$-basis. However, for non-prime-power m the power basis can be somewhat cumbersome and inefficient to work with. In Section 2.1 we consider other, more structured bases that are essential to our techniques.

If $m|m'$, we can view the mth cyclotomic ring $\mathcal{O}_m$ as a subring of $\mathcal{O}_{m'} = \mathbb{Z}[\zeta_{m'}]$, via the ring embedding (i.e., injective ring homomorphism) that maps ζ_m to $\zeta_{m'}^{m'/m}$. The ring extension $\mathcal{O}_{m'}/\mathcal{O}_m$ has degree $d = \varphi(m')/\varphi(m)$, and also d automorphisms τ_i (i.e., automorphisms of $\mathcal{O}_{m'}$ that fix $\mathcal{O}_m$ pointwise), which are defined by $\tau_i(\zeta_{m'}) = \zeta_{m'}^i$ for each $i \in \mathbb{Z}_{m'}^*$ such that $i = 1 \pmod{m}$. The *trace* function $\mathrm{Tr} = \mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m} : \mathcal{O}_{m'} \to \mathcal{O}_m$ can be defined as the sum of these automorphisms:

$$\mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m}(a) = \sum_i \tau_i(a) \in \mathcal{O}_m.$$

Notice that Tr is $\mathcal{O}_m$-linear by definition. If $\mathcal{O}_{m''}/\mathcal{O}_{m'}/\mathcal{O}_m$ is a tower of ring extensions, then the trace satisfies the composition property $\mathrm{Tr}_{\mathcal{O}_{m''}/\mathcal{O}_m} = \mathrm{Tr}_{\mathcal{O}_{m'}/\mathcal{O}_m} \circ \mathrm{Tr}_{\mathcal{O}_{m''}/\mathcal{O}_{m'}}$.

An important element in the mth cyclotomic ring is

$$g := \prod_{\text{odd prime } p|m} (1 - \zeta_p) \in \mathcal{O}_m. \tag{1}$$

Also define $\hat{m} = m/2$ if m is even, otherwise $\hat{m} = m$, for any cyclotomic index m. It is known that $g|\hat{m}$ (see, e.g., [16, Section 2.5.4]). The following lemma shows how the elements g in different cyclotomic rings, and the ideals they generate, are related by the trace function. (See the full version for a proof.)

Lemma 2.1. *Let $m|m'$ be positive integers and let $g \in R = \mathcal{O}_m, g' \in R' = \mathcal{O}_{m'}$ and $\hat{m}, \hat{m}'$ be as defined above. Then $\mathrm{Tr}_{R'/R}(g'R') = (\hat{m}'/\hat{m}) \cdot gR$, and in particular, $\mathrm{Tr}_{R'/R}(g') = (\hat{m}'/\hat{m}) \cdot g$.*

Later on we use the *scaled* trace function $(\hat{m}/\hat{m}')\operatorname{Tr}_{R'/R}$, which by the above lemma maps the ideal $g'R$ to gR, and g' to g.

Tensorial Decomposition of Cyclotomics. An important fact from algebraic number theory, used centrally in this work (and in [16]), is the *tensorial decomposition* of cyclotomic rings (and their bases) in terms of subrings. Let $\mathcal{O}_{m_1}, \mathcal{O}_{m_2}$ be cyclotomic rings. Then their largest common subring is $\mathcal{O}_{m_1} \cap \mathcal{O}_{m_2} = \mathcal{O}_g$ where $g = \gcd(m_1, m_2)$, and their smallest common extension ring, called the *compositum*, is $\mathcal{O}_{m_1} + \mathcal{O}_{m_2} = \mathcal{O}_l$ where $l = \operatorname{lcm}(m_1, m_2)$. When considered as extensions of $\mathcal{O}_g$, the ring $\mathcal{O}_l$ is isomorphic to the *ring tensor product* of $\mathcal{O}_{m_1}$ and $\mathcal{O}_{m_2}$, written as (sometimes suppressing $\mathcal{O}_g$ when it is clear from context)

$$\mathcal{O}_l/\mathcal{O}_g \cong (\mathcal{O}_{m_1}/\mathcal{O}_g) \otimes (\mathcal{O}_{m_2}/\mathcal{O}_g).$$

On the right, the ring tensor product is defined as the set of all $\mathcal{O}_g$-linear combinations of *pure tensors* $a_1 \otimes a_2$, with ring operations defined by $\mathcal{O}_g$-bilinearity and the mixed-product property $(a_1 \otimes a_2) \cdot (b_1 \otimes b_2) = (a_1 b_1) \otimes (a_2 b_2)$. The isomorphism with $\mathcal{O}_l/\mathcal{O}_g$ then simply identifies $a_1 \otimes a_2$ with $a_1 \cdot a_2 \in \mathcal{O}_l$. Note that any $a_1 \in \mathcal{O}_{m_1}$ corresponds to the pure tensor $a_1 \otimes 1$, and similarly for any $a_2 \in \mathcal{O}_{m_2}$.

The following simple lemma will be central to our techniques.

Lemma 2.2. *Let $m_1, m_2 > 0$ be integers and $g = \gcd(m_1, m_2)$, $l = \operatorname{lcm}(m_1, m_2)$. Then for any $\mathcal{O}_g$-linear function $\bar{L}\colon \mathcal{O}_{m_1} \to \mathcal{O}_{m_2}$, there is an (efficiently computable) $\mathcal{O}_{m_2}$-linear function $L\colon \mathcal{O}_l \to \mathcal{O}_{m_2}$ that coincides with $\bar{L}$ on the subring $\mathcal{O}_{m_1} \subseteq \mathcal{O}_l$.*

Proof. Write $\mathcal{O}_l \cong \mathcal{O}_{m_1} \otimes \mathcal{O}_{m_2}$, where the common base ring $\mathcal{O}_g$ is implicit. Let $L\colon (\mathcal{O}_{m_1} \otimes \mathcal{O}_{m_2}) \to \mathcal{O}_{m_2}$ be the $\mathcal{O}_g$-linear function uniquely defined by $L(a_1 \otimes a_2) = \bar{L}(a_1) \cdot a_2 \in \mathcal{O}_{m_2}$ for all pure tensors $a_1 \otimes a_2$. Then because $(a_1 \otimes a_2) \cdot b_2 = a_1 \otimes (a_2 b_2)$ for any $b_2 \in \mathcal{O}_{m_2}$ by the mixed-product property, L is also $\mathcal{O}_{m_2}$-linear. Finally, for any $a_1 \in \mathcal{O}_{m_1}$ we have $L(a_1 \otimes 1) = \bar{L}(a_1)$ by construction.

Ideal Factorization and Plaintext Slots. In the full version we recall the unique factorization of prime integers into prime ideals in cyclotomic rings, and, following [17], how the Chinese remainder theorem can yield several plaintext "slots" that embed $\mathbb{Z}_q$ as a subring, even for composite q.

In brief, for any prime integer p and cyclotomic ring R, the ideal pR factors as $pR = \prod_i \mathfrak{p}_i^e$ for some distinct prime ideals $\mathfrak{p}_i$ and some $e \geq 1$. Moreover, for any power $q = p^r$ where $r \geq 1$, the quotient ring $R/\mathfrak{p}_i^{re}$ embeds $\mathbb{Z}_q$ as a subring. By the Chinese Remainder Theorem (CRT), the natural ring homomorphism from R_q to the product ring $\bigoplus_i (R/\mathfrak{p}_i^{re})$ is an isomorphism. When the natural plaintext space of a cryptosystem is R_q, we refer to the quotient rings $R/\mathfrak{p}_i^{re}$ as the plaintext "$\mathbb{Z}_q$-slots" (or just "slots"), and use them to store vectors of $\mathbb{Z}_q$-elements via the CRT isomorphism. With this encoding, ring operations in R_q induce "batch" (or "SIMD") component-wise operations on the corresponding vectors of $\mathbb{Z}_q$ elements. We note that the CRT isomorphism is easy to compute in both directions. In particular, to map from a vector of $\mathbb{Z}_q$-elements to R_q just requires knowing a fixed "mod-q CRT set" $C = \{c_i\} \subset R$ for which $c_i = 1 \pmod{\mathfrak{p}_i^{re}}$ and $c_i = 0 \pmod{\mathfrak{p}_j^{re}}$ for all $j \neq i$. Such a set can be precomputed using, e.g., a generalization of the extended Euclidean algorithm.

Product Bases. Our bootstrapping technique relies crucially on certain highly structured bases and CRT sets, which we call "product bases (sets)," that arise from towers of cyclotomic rings. Let $\mathcal{O}_{m''}/\mathcal{O}_{m'}/\mathcal{O}_m$ be such a tower, let $B'' = \{b''_{j''}\} \subset \mathcal{O}_{m''}$ be any $\mathcal{O}_{m'}$-basis of $\mathcal{O}_{m''}$, and let $B' = \{b'_{j'}\} \subset \mathcal{O}_{m'}$ be any $\mathcal{O}_m$-basis of $\mathcal{O}_{m'}$. Then it follows immediately that the product set $B'' \cdot B' := \{b''_{j''} \cdot b'_{j'}\} \subset \mathcal{O}_{m''}$ is an $\mathcal{O}_m$-basis of $\mathcal{O}_{m''}$.[3] Of course, for a tower of several cyclotomic extensions and relative bases, we can obtain product bases that factor with a corresponding degree of granularity.

In the full version we show that the "powerful" and "decoding" bases of cyclotomic rings R, as defined in [16], admit "finest-possible" product structures, corresponding to any desired tower $R/\cdots/\mathbb{Z}$ of cyclotomic rings. (Other commonly used bases of $\mathcal{O}_m$, such as the power $\mathbb{Z}$-basis, do not admit such factorizations unless m is a prime power.) Similarly, we show how to construct CRT sets that have finest-possible factorizations.

2.2 Ring-Based Homomorphic Cryptosystem

Here we recall a somewhat-homomorphic encryption scheme whose security is based on the ring-LWE problem [15] in arbitrary cyclotomic rings. For our purposes we focus mainly on its decryption function, though below we also recall its support for "ring switching" [11]. For further details on its security guarantees, homomorphic properties, and efficient implementation, see [15, 5, 3, 14, 11, 16].

Let $R = \mathcal{O}_m \subseteq R' = \mathcal{O}_{m'}$ be respectively the mth and m'th cyclotomic rings, where $m|m'$. The plaintext ring is the quotient ring R_p for some integer p; ciphertexts are made up of elements of R'_q for some integer q, which for simplicity we assume is divisible by p; and the secret key is some $s \in R'$. The case $m = 1$ corresponds to "non-packed" ciphertexts, which encrypt elements of $\mathbb{Z}_p$ (e.g., single bits), whereas $m = m'$ corresponds to "packed" ciphertexts, and $1 < m < m'$ corresponds to what we call "semi-packed" ciphertexts. Note that without loss of generality we can treat any ciphertext as packed, since R'_p embeds R_p. But the smaller m is, the simpler and more practically efficient our bootstrapping procedure can be. Since our focus is on refreshing ciphertexts that have large noise rate, we can think of m' as being somewhat small (e.g., in the several hundreds) via ring-switching [11], and q also as being somewhat small (e.g., in the several thousands) via modulus-switching. Our main focus in this work is on a plaintext modulus p that is a power of two, though for generality we present all our techniques in terms of arbitrary p.

[3] Formally, this basis is a *Kronecker* product of the bases B'' and B', which is typically written using the $\otimes$ operator. We instead use $\cdot$ to avoid confusion with pure tensors in a ring tensor product, which the elements of $B'' \cdot B'$ may not necessarily be.

A ciphertext encrypting a message $\mu \in R_p$ under secret key $s' \in R'$ is some pair $c' = (c'_0, c'_1) \in R'_q \times R'_q$ satisfying the relation

$$c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \pmod{qR'} \tag{2}$$

for some error (or "noise") term $e' \in R'$ such that $e' \cdot g' \in g'R'$ is sufficiently "short," where $g' \in R'$ is as defined in Equation (1).[4] Informally, the "noise rate" of the ciphertext is the ratio of the "size" of e' (or more precisely, the magnitude of its coefficients in a suitable basis) to q/p.

We note that Equation (2) corresponds to what is sometimes called the "most significant bit" (msb) message encoding, whereas somewhat-homomorphic schemes are often defined using "least significant bit" (lsb) encoding, in which p and q are coprime and $c'_0 + c'_1 s' = e' \pmod{qR'}$ for some error term $e' \in \mu + pR'$. For our purposes the msb encoding is more natural, and in any case the two encodings are essentially equivalent; see the full version for details.

Decryption. At a high level, the decryption algorithm works in two steps: the "linear" step simply computes $v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \in R'_q$, and the "non-linear" step outputs $\lfloor v' \rceil_p \in R_p$ using a certain "ring rounding function" $\lfloor \cdot \rceil_p \colon R'_q \to R_p$. As long as the error term e' is within the tolerance of the rounding function, the output will be $\mu \in R_p$. This is all entirely analogous to decryption in LWE-based systems, but here the rounding is n-dimensional, rather than just from $\mathbb{Z}_q$ to $\mathbb{Z}_p$.

Concretely, the ring rounding function $\lfloor \cdot \rceil_p \colon R'_q \to R_p$ is defined in terms of the integer rounding function $\lfloor \cdot \rceil_p \colon \mathbb{Z}_q \to \mathbb{Z}_p$ and a certain "decryption" $\mathbb{Z}$-basis $B' = \{b_j\}$ of R', as follows.[5] Represent the input $v' \in R'_q$ in the decryption basis as $v' = \sum_j v'_j \cdot b'_j$ for some coefficients $v'_j \in \mathbb{Z}_q$, then independently round the coefficients, yielding an element $\sum \lfloor v'_j \rceil_p \cdot b'_j \in R'_p$ that corresponds to the message $\mu \in R_p$ (under the standard embedding of R_p into R'_p).

Changing the Plaintext Modulus. We use two operations on ciphertexts that alter the plaintext modulus p and encrypted message $\mu \in R_p$. The first operation changes p to any multiple $p' = dp$, and produces an encryption of

[4] Quantitatively, "short" is defined with respect to the *canonical embedding* of R', whose precise definition is not needed in this work. The above system is equivalent to the one from [16] in which the message, error term, and ciphertext components are all taken over the "dual" fractional ideal $(R')^\vee = (g'/\hat{m}')R'$ in the m'th cyclotomic number field, and the error term has an essentially spherical distribution over $(R')^\vee$. In that system, decryption is best accomplished using a certain $\mathbb{Z}$-basis of $(R')^\vee$, called the *decoding basis*, which optimally decodes spherical errors. The above formulation is more convenient for our purposes, and simply corresponds with multiplying everything in the system of [16] by an $\hat{m}'/g'$ factor. This makes $e' \cdot g' \in g'R' = \hat{m}'(R')^\vee)$ short and essentially spherical in our formulation. See [15, 16] for further details.

[5] In our formulation, the basis B' is $(\hat{m}'/g')$ times the decoding basis of $(R')^\vee$. See Section 2.1 and Footnote 4.

some $\mu' \in R'_{p'}$ such that $\mu' = \mu \pmod{pR'}$. To do this, it simply "lifts" the input ciphertext $c' = (c'_0, c'_1) \in (R'_q)^2$ to an arbitrary $c'' = (c''_0, c''_1) \in (R'_{q'})^2$ such that $c''_j = c'_j \pmod{qR'}$, where $q' = dq$. The second operation applies to an encryption of a message $\mu \in R_p$ that is known to be divisible by some divisor d of p, and produces an encryption of $\mu/d \in R_{p/d}$. The operation actually leaves the ciphertext c' unchanged; it just declares the associated plaintext modulus to be p/d (which affects how decryption is performed).

Ring Switching. We rely heavily on the cryptosystem's support for switching ciphertexts to a cyclotomic subring S' of R', which as a side-effect homomorphically evaluates any desired S'-linear function on the plaintext. Notice that the linear function L is applied to the plaintext as embedded in R'_p; this obviously applies the induced function on the true plaintext space R_p.

Proposition 2.3 ([11], full version). *Let $S' \subseteq R'$ be cyclotomic rings. Then the above-described cryptosystem supports the following homomorphic operation: given any S'-linear function $L\colon R'_p \to S'_p$ and a ciphertext over R'_q encrypting (with sufficiently small error term) a message $\mu \in R'_p$, the output is a ciphertext over S'_q encrypting $L(\mu) \in S'_p$.*

The security of the procedure described in Proposition 2.3 is based on the hardness of the ring-LWE problem in S', so the dimension of S' must be sufficiently large. The procedure itself is quite simple and efficient: it first switches to a secret key that lies in the subring S', then it multiplies the resulting ciphertext by an appropriate fixed element of R' (which is determined solely by the function L). Finally, it applies to the ciphertext the trace function $\mathrm{Tr}_{R'/S'}\colon R' \to S'$. All of these operations are quasi-linear time in the dimension of $R'/\mathbb{Z}$, and very efficient in practice. In particular, the trace is a trivial linear-time operation when elements are represented in any of the bases we use. The ring-switching procedure increases the effective error rate of the ciphertext by a factor of about the square root of the dimension of R', which is comparable to that of a single homomorphic multiplication. See [11] for further details.

3 Overview of Bootstrapping Procedure

Here we give a high-level description of our bootstrapping procedure. We present a unified procedure for non-packed, packed, and semi-packed ciphertexts, but note that for non-packed ciphertexts, Steps 3a and 3c (and possibly 1c) are null operations, while for packed ciphertexts, Steps 1b, 1c, and 2 are null operations.

Recalling the cryptosystem from Section 2.2, the plaintext ring is R_p and the ciphertext ring is R'_q, where $R = \mathcal{O}_m \subseteq R' = \mathcal{O}_{m'}$ are cyclotomic rings (so $m|m'$), and q is a power of p. The procedure also uses a larger cyclotomic ring $R'' = \mathcal{O}_{m''} \supseteq R'$ (so $m'|m''$) to work with ciphertexts that encrypt elements of the original ciphertext ring R'_q. We can choose m'' however we like, subject to the constraints below.

To obtain quasilinear runtimes and exponential security under standard hardness assumptions, our procedure imposes some mild conditions on the indices m, m', and m'':

- The dimension $\varphi(m'')$ of R'' must be quasilinear, so we can represent elements of R'' efficiently.
- For Steps 2 and 3, all the prime divisors of m and m' must be small (i.e., polylogarithmic).
- For Step 3, m and m''/m must be coprime, which implies that m and m'/m must be coprime also. Note that the former condition is always satisfied for non-packed ciphertexts (where $m = 1$). For packed ciphertexts (where $m = m'$), the latter condition is always satisfied, which makes it easy to choose a valid m''. For semi-packed ciphertexts (where $1 < m < m'$), we can always satisfy the latter condition either by increasing m (at a small expense in practical efficiency in Step 3), or by effectively decreasing m slightly (at a possible improvement in practical efficiency); see the full version for details.

The input to the procedure is a ciphertext $c' = (c'_0, c'_1) \in (R'_q)^2$ that encrypts some plaintext $\mu \in R_p$ under a secret key $s' \in R'$, i.e., it satisfies the relation

$$v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \pmod{qR'}$$

for some small enough error term $e' \in R'$. The procedure computes a new encryption of $\lfloor v \rceil_p = \mu$ (under some secret key, not necessarily s') that has substantially smaller noise rate than the input ciphertext. It proceeds as follows (explanatory remarks appear in italics):

1. Convert c' to a "noiseless" ciphertext c'' over a large ring R''_Q that encrypts a plaintext $(g'/g)u' \in R'_{q'}$, where $g' \in R', g \in R$ are as defined in Equation (1), $q' = (\hat{m}'/\hat{m})q$, and $u' = v' \pmod{qR'}$. This proceeds in the following substeps (see Section 3.1 for further details).
   Note that $g'/g \in R'$ by definition, and that $(g'/g) | (\hat{m}'/\hat{m})$.
   (a) Reinterpret c' as a noiseless encryption of $v' = \frac{q}{p} \cdot \mu + e' \in R'_q$ as a *plaintext*, noting that both the plaintext and ciphertext rings are now taken to be R'_q.
   This is purely a conceptual change in perspective, and does not involve any computation.
   (b) Using the procedure described in Section 2.2, change the plaintext (and ciphertext) modulus to $q' = (\hat{m}'/\hat{m})q$, yielding a noiseless encryption of some $u' \in R'_{q'}$ such that $u' = v' \pmod{qR'}$.
   Note that this step is a null operation if the original ciphertext was packed, i.e., if $m = m'$.
   We need to increase the plaintext modulus because homomorphically computing $\mathrm{Tr}_{R'/R}$ in Step 2 below introduces an $\hat{m}'/\hat{m}$ factor into the plaintext, which we will undo by scaling the plaintext modulus back down to q.
   (c) Multiply the ciphertext from the previous step by $(g'/g) \in R'$, yielding a noiseless encryption of plaintext $(g'/g)u' \in R'_{q'}$.

The factor $(g'/g) \in R'$ is needed when we homomorphically compute $\mathrm{Tr}_{R'/R}$ in Step 2 below. Note that $g'/g = 1$ if and only if every odd prime divisor of m' also divides m, e.g., if $m = m'$.

(d) Convert to a noiseless ciphertext c'' that still encrypts $(g'/g)u' \in R'_{q'}$, but using a large enough ciphertext ring R''_Q for some $R'' = \mathcal{O}_{m''} \supseteq R'$ and modulus $Q \gg q'$.

A larger ciphertext ring R''_Q is needed for security in the upcoming homomorphic operations, to compensate for the low noise rates that will need to be used. These operations will expand the initial noise rate by a quasipolynomial $\lambda^{O(\log \lambda)}$ factor in total, so the dimension of R'' and the bit length of Q can be $\tilde{O}(\lambda)$ and $\tilde{O}(1)$, respectively.

The remaining steps are described here only in terms of their effect on the *plaintext* value and ring. Using ring- and modulus-switching, the ciphertext ring R'' and modulus Q may be made smaller as is convenient, subject to the security and functionality requirements. (Also, the ciphertext ring implicitly changes during Steps 3a and 3c.)

2. Homomorphically apply the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ to the encryption of $(g'/g)u' \in R'_{q'}$, to obtain an encryption of plaintext

$$u = \frac{\hat{m}}{\hat{m}'} \cdot \mathrm{Tr}_{R'/R}\Big(\frac{g'}{g} \cdot u'\Big) = \frac{q}{p} \cdot \mu + e \in R_q$$

for some suitably small error term $e \in R$. See Section 4 further details.

This step changes the plaintext ring from $R'_{q'}$ to R_q, and homomorphically isolates the noisy R_q-encoding of μ. It is a null operation if the original ciphertext was packed, i.e., if $m = m'$.

3. Homomorphically apply the ring rounding function $\lfloor\cdot\rceil_p \colon R_q \to R_p$, yielding an output ciphertext that encrypts $\lfloor u\rceil_p = \mu \in R_p$. This proceeds in three sub-steps, all of which are applied homomorphically (see Section 5 for details):

(a) Map the coefficients u_j of $u \in R_q$ (with respect to the decryption basis B of R) to the $\mathbb{Z}_q$-slots of a ring S_q, where S is a suitably chosen cyclotomic. *This step changes the plaintext ring from R_q to S_q. It is a null operation if the original ciphertext was non-packed (i.e., if $m = 1$), because we can let $S = R = \mathbb{Z}$.*

(b) Batch-apply the integer rounding function $\lfloor\cdot\rceil \colon \mathbb{Z}_q \to \mathbb{Z}_p$ to the $\mathbb{Z}_q$-slots of S_q, yielding a ciphertext that encrypts the values $\mu_j = \lfloor u_j\rceil_p \in \mathbb{Z}_p$ in its $\mathbb{Z}_p$-slots.

This step changes the plaintext ring from S_q to S_p. It constitutes the only non-linear operation on the plaintext, with multiplicative depth $\lceil \lg p\rceil \cdot (\log_p(q) - 1) \approx \log(q)$, and as such is the most expensive in terms of runtime, noise expansion, etc.

(c) Reverse the map from the step 3a, sending the values μ_j from the $\mathbb{Z}_p$-slots of S_p to coefficients with respect to the decryption basis B of R_p, yielding an encryption of $\mu = \sum_j \mu_j b_j \in R_p$.

This step changes the plaintext ring from S_p to R_p. Just like step 3a, it is a null operation for non-packed ciphertexts.

In the full version we describe a few minor variants and practical optimizations of our basic procedure.

3.1 Obtaining a Noiseless Ciphertext

Step 1 of our bootstrapping procedure is given as input a ciphertext $c' = (c'_0, c'_1)$ over R'_q that encrypts (typically with a high noise rate) a message $\mu \in R_p$ under key $s' \in R$, i.e., $v' = c'_0 + c'_1 \cdot s' = \frac{q}{p} \cdot \mu + e' \in R'_q$ for some error term e'. We first change our perspective and view c' as a "noiseless" encryption (still under s') of the plaintext value $v' \in R'_q$, taking both the plaintext and ciphertext rings to be R'_q. This view is indeed formally correct, because

$$c'_0 + c'_1 \cdot s' = \frac{q}{q} \cdot v' + 0 \pmod{qR'}.$$

Next, in preparation for the upcoming homomorphic operations we increase the plaintext (and ciphertext) modulus to q', and multiply the resulting ciphertext by g'/g. These operations clearly preserve noiselessness. Finally, we convert the ciphertext ring to R''_Q for a sufficiently large cyclotomic $R'' \supseteq R'$ and modulus $Q \gg q$ that is divisible by q. This is done simply by embedding R' into R'' and introducing extra precision, i.e., scaling the ciphertext up by a Q/q factor. It is easy to verify that these operations also preserve noiselessness.

4 Homomorphic Trace

Here we show how to perform Step 2 of our bootstrapping procedure, which homomorphically evaluates the scaled trace function $(\hat{m}/\hat{m}') \operatorname{Tr}_{R'/R}$ on an encryption of $(g'/g)u' \in R'_{q'}$, where recall that: $g' \in R', g \in R$ are as defined in Equation (1), and (g'/g) divides $(\hat{m}'/\hat{m})$; the plaintext modulus is $q' = (\hat{m}'/\hat{m})q$; and

$$u' = v' = \frac{q}{p} \cdot \mu + e' \pmod{qR'},$$

where $e' \cdot g' \in g'R'$ is sufficiently short. Our goal is to show that:

1. the scaled trace of the plaintext $(g'/g)u'$ is some $u = \frac{q}{p} \cdot \mu + e \in R_q$, where $e \cdot g \in gR$ is short, and
2. we can efficiently homomorphically apply the scaled trace on a ciphertext c'' over some larger ring $R'' = \mathcal{O}_{m''} \supseteq R'$.

4.1 Trace of the Plaintext

We first show the effect of the scaled trace on the plaintext $(g'/g)u' \in R'_{q'}$. By the above description of $u' \in R'_{q'}$ and the fact that $(g'/g)q$ divides $q' = (\hat{m}'/\hat{m})q$, we have

$$(g'/g)u' = (g'/g)v' = (g'/g)\left(\frac{q}{p} \cdot \mu + e'\right) \pmod{(g'/g)qR'}.$$

Therefore, letting $\mathrm{Tr} = \mathrm{Tr}_{R'/R}$, by R-linearity of the trace and Lemma 2.1, we have

$$\begin{aligned}\mathrm{Tr}((g'/g)u') &= \mathrm{Tr}(g'/g) \cdot \frac{q}{p} \cdot \mu + \mathrm{Tr}(e' \cdot g')/g \\ &= \frac{\hat{m}'}{\hat{m}}\left(\frac{q}{p} \cdot \mu + e\right) \pmod{q'R},\end{aligned}$$

where $e = (\hat{m}/\hat{m}')\,\mathrm{Tr}(e' \cdot g')/g \in R$. Therefore, after scaling down the plaintext modulus q' by an $\hat{m}'/\hat{m}$ factor (see Section 2.2), the plaintext is $\frac{q}{p} \cdot \mu + e \in R_q$.

Moreover, $e \cdot g = (\hat{m}/\hat{m}')\,\mathrm{Tr}(e' \cdot g') \in gR$ is short because $e' \cdot g' \in g'R'$ is short; see, e.g., [11, Corollary 2.2]. In fact, by basic properties of the decoding/decryption basis (as defined in [16]) under the trace, the coefficient vector of e with respect to the decryption basis of R is merely a subvector of the coefficient vector of e' with respect to the decryption basis of R'. Therefore, e is within the error tolerance of the rounding function on R_q, assuming e' is within the error tolerance of the rounding function on R'_q.

4.2 Applying the Trace

Now we show how to efficiently homomorphically apply the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ to an encryption of any plaintext in $R'_{q'}$ that is divisible by (g'/g). Note that this condition ensures that the output of the trace is a multiple of $\hat{m}/\hat{m}'$ in $R_{q'}$ (see Lemma 2.1), making the scaling a well-defined operation that results in an element of R_q.

First recall that $\mathrm{Tr}_{R'/R}$ is the sum of all $\varphi(m')/\varphi(m)$ automorphisms of R'/R, i.e., automorphisms of R' that fix R pointwise. So as mentioned in the introduction, one way of homomorphically computing the scaled trace is to homomorphically apply the proper automorphisms, sum the results, and scale down the plaintext and its modulus. While this "sum-automorphisms" procedure yields the correct result, computing the trace in this way does not run in quasilinear time, unless the number $\varphi(m')/\varphi(m)$ of automorphisms is only polylogarithmic.

Instead, we consider a sufficiently fine-grained tower of cyclotomic rings

$$R^{(r)}/\cdots/R^{(1)}/R^{(0)},$$

where $R' = R^{(r)}$, $R = R^{(0)}$, and each $R^{(i)} = \mathcal{O}_{m_i}$, where m_i is divisible by m_{i-1} for $i > 0$. E.g., for the finest granularity we would choose the tower so that every m_i/m_{i-1} is prime. Notice that the scaled trace function $(\hat{m}/\hat{m}')\,\mathrm{Tr}_{R'/R}$ is the composition of the scaled trace functions $(\hat{m}_{i-1}/\hat{m}_i)\,\mathrm{Tr}_{R^{(i)}/R^{(i-1)}}$, and that g'/g is the product of all $g^{(i)}/g^{(i-1)}$ for $i = 1, \ldots, r$, where $g^{(i)} \in R^{(i)}$ is as defined in Equation (1). So, another way of homomorphically applying the full scaled trace is to apply the corresponding scaled trace in sequence for each level of the tower, "climbing down" from $R' = R^{(r)}$ to $R = R^{(0)}$. In particular, if we use the above sum-automorphisms procedure with a tower of finest granularity, then there are at most $\log_2(m'/m) = O(\log \lambda)$ levels, and since we have assumed that every prime divisor of m'/m is bounded by polylogarithmic in the security parameter λ, the full procedure will run in quasilinear $\tilde{O}(\lambda)$ time.

In the full version we give all the details of the sum-automorphisms procedure sketched above, as well as an alternative procedure using ring-switching that is preferable in certain cases.

5 Homomorphic Ring Rounding

In this section we describe how to efficiently homomorphically evaluate the "ring rounding function" $\lfloor\cdot\rceil_p \colon R_q \to R_p$, where $R = \mathcal{O}_m$ is the mth cyclotomic ring. Conceptually, we follow the high-level strategy from [12], but instantiate it with very different technical components. Recall from Section 2.2 that the rounding function expresses its input u in the "decryption" $\mathbb{Z}$-basis $B = \{b_j\}$ of R, as $u = \sum_j u_j \cdot b_j$ for $u_j \in \mathbb{Z}_q$, and outputs $\lfloor u \rceil_p := \sum_j \lfloor u_j \rceil_p \cdot b_j \in R_p$. Unlike with integer rounding from $\mathbb{Z}_q$ to $\mathbb{Z}_p$, it is not clear whether this rounding function has a low-depth arithmetic formula using just the ring operations of R. One difficulty is that there are an *exponentially* large number of values in R_q that map to a given value in R_p, which might be seen as evidence that a corresponding arithmetic formula must have large depth. Fortunately, we show how to circumvent this issue by using an additional homomorphic operation, namely, an enhancement of ring-switching. In short, we reduce the homomorphic evaluation of the ring rounding function (from R_q to R_p) very simply and efficiently to that of several parallel (batched) evaluations of the integer rounding function (from $\mathbb{Z}_q$ to $\mathbb{Z}_p$).

Suppose we choose some cyclotomic ring $S = \mathcal{O}_\ell$ having a mod-q CRT set $C = \{c_j\} \subset S$ of cardinality exactly $|B|$. That is, we have a ring embedding from the product ring $\mathbb{Z}_q^{|B|}$ into S_q, given by $\mathbf{u} \mapsto \sum_j u_j \cdot c_j$. Note that the choice of the ring S is at our convenience, and need not have any relationship to the plaintext ring R_q. We express the rounding function $R_q \to R_p$ as a sequence of three steps:

1. Map $u = \sum_j u_j \cdot b_j \in R_q$ to $\sum_j u_j \cdot c_j \in S_q$, i.e., send the $\mathbb{Z}_q$-coefficients of u (with respect to the decryption basis B) to the $\mathbb{Z}_q$-slots of S_q.
2. Batch-apply the integer rounding function from $\mathbb{Z}_q$ to $\mathbb{Z}_p$ to the slot values u_j, to get $\sum_j \lfloor u_j \rceil_p \cdot c_j \in S_2$.
3. Invert the map from the first step to obtain $\lfloor u \rceil_p = \sum_j \lfloor u_j \rceil_2 \cdot b_j \in R_2$.

Using batch/SIMD operations [17], the second step is easily achieved using the fact that S_q embeds the product of several copies of the ring $\mathbb{Z}_q$, via the CRT elements c_j. That is, we can simultaneously round all the coefficients u_j to $\mathbb{Z}_p$, using just one evaluation of an arithmetic procedure over S corresponding to one for the integer rounding function from $\mathbb{Z}_q$ to $\mathbb{Z}_p$.

We now describe one way of expressing the first and third steps above, in terms of operations that can be evaluated homomorphically. The first simple observation is that the function mapping $u = \sum_j u_j \cdot b_j$ to $\sum_j u_j \cdot c_j$ is induced by a $\mathbb{Z}$-linear function $\bar{L} \colon R \to S$. Specifically, $\bar{L}$ simply maps each $\mathbb{Z}$-basis element b_j to c_j. Now suppose that we choose S so that its largest common subring with R is $\mathbb{Z}$, i.e., the indices m, ℓ are coprime. Then letting $T = R + S = \mathcal{O}_{m\ell} \cong R \otimes S$ be the compositum ring, Lemma 2.2 yields an S-linear function $L \colon T \to S$ that coincides with $\bar{L}$ on $R \subseteq T$, and in particular on u. The ring-switching procedure from Proposition 2.3 can homomorphically evaluate any S-linear function from T to S, and in particular, the function L. Therefore, by simply embedding R into T, we can homomorphically evaluate $\bar{L}(x) = L(x)$ by applying the ring-switching procedure with L.

Unfortunately, there is a major problem with the efficiency of the above approach: the *dimension* (over $\mathbb{Z}$) of the compositum ring T is the product of those of R and S, which are each at least linear in the security parameter. Therefore, representing and operating on arbitrary elements in T requires at least quadratic time.

Efficiently Mapping from $\boldsymbol{B}$ to $\boldsymbol{C}$. In hindsight, the quadratic runtime of the above approach should not be a surprise, because we treated $\bar{L} \colon R \to S$ as an arbitrary $\mathbb{Z}$-linear transformation, and B, C as arbitrary sets. To do better, $\bar{L}$, B, and C must have some structure we can exploit. Fortunately, they can—*if* we choose them carefully. We now describe a way of expressing the first and third steps above in terms of simple operations that can be evaluated homomorphically in quasilinear time.

The main idea is as follows: instead of mapping directly from R to S, we will express $\bar{L}$ as a *sequence* of linear transformations $\bar{L}_1, \ldots, \bar{L}_r$ through several "hybrid" cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$. For sets B and C with an appropriate product structure, these transformations will respectively map $A_0 = B \subset H^{(0)}$ to some structured subset $A_1 \subset H^{(1)}$, then A_1 to some structured subset $A_2 \subset H^{(2)}$, and so on, finally mapping A_{r-1} to $A_r = C \subset H^{(r)}$. In contrast to the inefficient method described above, the hybrid rings will be chosen so that each compositum $T^{(i)} = H^{(i-1)} + H^{(i)}$ of adjacent rings has dimension just *slightly larger* (by only a polylogarithmic factor) than that of R. This is achieved by choosing the indices of $H^{(i-1)}, H^{(i)}$ to have large greatest common divisor, and hence small least common multiple. For example, the indices can share almost all the same prime divisors (with multiplicity), and have just one different prime divisor each. Of course, other tradeoffs between the number of hybrid rings and the dimensions of the compositums are also possible.

The flip side of this approach is that using ring-switching, we can homomorphically evaluate only $E^{(i)}$*-linear* functions $\bar{L}_i \colon H^{(i-1)} \to H^{(i)}$, where $E^{(i)} = H^{(i-1)} \cap H^{(i)}$ is the largest common subring of adjacent hybrid rings. Since each $E^{(i)}$ is large by design (to keep the compositum $T^{(i)}$ small), this requirement is quite strict, yet we still need to construct linear functions $\bar{L}_i$ that sequentially map $B = A_0$ to $C = A_r$. To achieve this, we construct all the sets A_i to have appropriate product structure. Specifically, we ensure that for each $i = 1, \ldots, r$, we have factorizations

$$A_{i-1} = A_{i-1}^{\text{out}} \cdot Z_i, \quad A_i = A_i^{\text{in}} \cdot Z_i \tag{3}$$

for some set $Z_i \subset E^{(i)}$, where both A_{i-1}^{out} and A_i^{in} are linearly independent over $E^{(i)}$. (Note that for $1 \leq i < r$, each A_i needs to factor in two ways over two subrings $E^{(i-1)}$ and $E^{(i)}$, which is why we need two sets A_i^{in} and A_i^{out}.) Then, we simply define $\bar{L}_i$ to be an arbitrary $E^{(i)}$-linear function that bijectively maps A_{i-1}^{out} to A_i^{in}. (Note that A_{i-1}^{out} and A_i^{in} have the same cardinality, because A_{i-1} and A_i do.) It immediately follows that $\bar{L}_i$ bijectively maps A_{i-1} to A_i, because

$$\bar{L}_i(A_{i-1}) = \bar{L}_i(A_{i-1}^{\text{out}} \cdot Z_i) = \bar{L}_i(A_{i-1}^{\text{out}}) \cdot Z_i = A_i^{\text{in}} \cdot Z_i$$

by $E^{(i)}$-linearity and the fact that $Z_i \subset E^{(i)}$.

Summarizing the above discusion, we have the following theorem.

Theorem 5.1. *Suppose there are cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$ and sets $A_i \subset H^{(i)}$ such that for all $i = 1, \ldots, r$, we have $A_{i-1} = A_{i-1}^{out} \cdot Z_i$ and $A_i = A_i^{in} \cdot Z_i$ for some sets $Z_i \subset E^{(i)} = H^{(i-1)} \cap H^{(i)}$ and A_{i-1}^{out}, A_i^{in} that are each $E^{(i)}$-linearly independent and of equal cardinality. Then there is a sequence of $E^{(i)}$-linear maps $\bar{L}_i \colon H^{(i-1)} \to H^{(i)}$, for $i = 1, \ldots, r$, whose composition $\bar{L}_r \circ \cdots \circ \bar{L}_1$ bijectively maps A_0 to A_r.*

So far we have described how our desired map between *plaintext* rings R and S can be expressed as a sequence of linear maps through hybrid rings. In the context of bootstrapping, for security these plaintext rings typically need to be embedded in some larger ciphertext rings, because the dimensions of R, S are not large enough to securely support the very small noise used in bootstrapping. For example, following Step 2 of our bootstrapping procedure (Section 3), we have a ciphertext over the ring R'' where $R'' = \mathcal{O}_{m''} \supseteq R$ for some m'' of our choice that is divisible by m. We need to choose the sequence of hybrid *ciphertext* rings so that they admit suitable linear functions that induce the desired ones on the corresponding *plaintext* rings. Achieving this is easy; see the full version for details.

Construction. In the full version we construct hybrid cyclotomic rings $R = H^{(0)}, H^{(1)}, \ldots, H^{(r)} = S$ and sets $A_i \subset H^{(i)}$ (where $A_0 = B$ and $A_r = C$) to satisfy the following two properties for each $i = 1, \ldots, r$:

1. Each compositum $T^{(i)} = H^{(i-1)} + H^{(i)}$ is not too large, i.e., its dimension is quasilinear.
2. The sets A_{i-1}, A_i factor as described in Equation (3).

The main ideas are as follows: view R as the top level of a fine-grained cyclotomic tower, and choose a target ring S as the top level of a fine-grained tower that has sufficiently many $\mathbb{Z}_q$-slots at each level. Consider finest-possible factorizations of the decryption basis B of R, and of a mod-q CRT set C of S. Then to define the hybrid rings and sets A_{i-1}, A_i, for each successive hybrid ring we "tear down" a level from the top of the R-tower and the corresponding component of B, and "build up" another level from the bottom of the S-tower and the corresponding component of the CRT set C.

References

[1] Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 719–737. Springer, Heidelberg (2012)

[2] Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapSVP. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 868–886. Springer, Heidelberg (2012)

[3] Brakerski, Z., Gentry, C., Vaikuntanathan, V. (Leveled) fully homomorphic encryption without bootstrapping. In: ICTS, pp. 309–325 (2012)

[4] Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: FOCS, pp. 97–106 (2011)

[5] Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 505–524. Springer, Heidelberg (2011)

[6] Cheon, J.H., Coron, J.-S., Kim, J., Lee, M.S., Lepoint, T., Tibouchi, M., Yun, A.: Batch fully homomorphic encryption over the integers. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 315–335. Springer, Heidelberg (2013)

[7] Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009), `http://crypto.stanford.edu/craig`

[8] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC, pp. 169–178 (2009)

[9] Gentry, C., Halevi, S.: Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In: FOCS, pp. 107–109 (2011)

[10] Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011)

[11] Gentry, C., Halevi, S., Peikert, C., Smart, N.P.: Ring switching in BGV-style homomorphic encryption. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 19–37. Springer, Heidelberg (2012), `http://eprint.iacr.org/2012/240`

[12] Gentry, C., Halevi, S., Smart, N.P.: Better bootstrapping in fully homomorphic encryption. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 1–16. Springer, Heidelberg (2012)

[13] Gentry, C., Halevi, S., Smart, N.P.: Fully homomorphic encryption with polylog overhead. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 465–482. Springer, Heidelberg (2012)

[14] Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: Safavi-Naini, R. (ed.) CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (2012)

[15] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. J. ACM (2013); To appear Preliminary version In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010)

[16] Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 35–54. Springer, Heidelberg (2013)

[17] Smart, N.P., Vercauteren, F.: Fully homomorphic SIMD operations. Cryptology ePrint Archive, Report 2011/133 (2011), `http://eprint.iacr.org/`

[18] van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 24–43. Springer, Heidelberg (2010)