Packing Messages and Optimizing Bootstrapping in GSW-FHE

Ryo Hiromasa[1], Masayuki Abe[2], and Tatsuaki Okamoto[2]

[1] Kyoto University
hiromasa@ai.soc.i.kyoto-u.ac.jp
[2] NTT Secure Platform Laboratories
{abe.masayuki, okamoto.tatsuaki}@lab.ntt.co.jp

Abstract. We construct the first fully homomorphic encryption (FHE) scheme that encrypts *matrices* and supports homomorphic *matrix* addition and multiplication. This is a natural extension of packed FHE and thus supports more complicated homomorphic operations. We optimize the bootstrapping procedure of Alperin-Sheriff and Peikert (CRYPTO 2014) by applying our scheme. Our optimization decreases the lattice approximation factor from $\tilde{O}(n^3)$ to $\tilde{O}(n^{2.5})$. By taking a lattice dimension as a larger polynomial in a security parameter, we can also obtain the same approximation factor as the best known one of standard lattice-based public-key encryption *without* successive dimension-modulus reduction, which was essential for achieving the best factor in prior works on bootstrapping of standard lattice-based FHE.

1 Introduction

Fully homomorphic encryption (FHE) allows us to evaluate any function over encrypted data by only using public information. This can be used, for example, to outsource computations to remote servers without compromising privacy. Since the breakthrough work by Gentry [12, 13], many different varieties of FHE have been proposed [5–8, 11, 17, 18]. To date, the fastest (and simplest) FHE based on the *standard* lattice assumption is the one by Gentry, Sahai, and Waters [17]. (hereafter, referred to as GSW-FHE). However, it is required to take heavy cost for evaluating a large number of ciphertexts. The way to deal with this issue is to *pack* multiple messages into one ciphertext.

Packing messages allows us to apply *single-instruction-multiple data* (SIMD) homomorphic operations to all encrypted messages. In the case where a remote server stores encrypted data and we want to retrieve certain data from this server, we first apply the equality function to every encrypted data. If the stored data have been packed into one ciphertext, we can do that by only one homomorphic evaluation of the equality function. Smart and Vercautren [25], for the first time, showed that applying the Chinese reminder theorem (CRT) to number fields partitions the message space of the Gentry's FHE [12, 13] scheme into a vector of *plaintext slots.* On the standard lattice-based FHE schemes, Brakerski, Gentry, and Halevi [4] used the method of [22], which described a way to construct packed

Regev's encryption [23], to pack messages in the FHE variants [5–7] of [23]. In this paper, we construct a matrix variant of [17] (whose security is also based on the standard lattice assumption) to implement SIMD homomorphic operations, and describe how to bring out the potential of our scheme: specifically optimizing *bootstrapping*.

The bootstrapping technique [12, 13] is currently the only way to go from limited amount of homomorphism to unlimited amount of homomorphism. The limited nature is caused by noise terms included in ciphertexts of all known FHE, which are needed to ensure security. Since homomorphic operations increases the noise level and the noise prevents us from correctly decrypting ciphertexts if the level increases too high, it is required to consider methods that reduce the noise. The bootstrapping technique is the one of such a methods, and achieved by homomorphically evaluating the decryption circuit of FHE.

There have recently been the significant progresses [1, 9] in improving the bootstrapping procedure on standard lattice-based FHE. Their progresses stem from the observation that noise terms in ciphertexts of GSW-FHE grow *asymmetrically*: for a parameter n (the dimension in the underlying lattice assumption), the noise of multiplication between two ciphertexts with noise size e_1 and e_2 grows to $e_1 + \mathrm{poly}(n) \cdot e_2$. For example, if we want to multiply ℓ ciphertexts with the same noise size in *sequence*, the noise in the result increases by a factor of $\ell \cdot \mathrm{poly}(n)$, which is in contrast to the noise blowup factor for all known FHE, $\mathrm{poly}(n)^{\log \ell}$. To suppress the growth in noise from the bootstrapping procedure, the two recent developments [1, 9] tried to *sequentialize* the decryption circuit.

Brakerski and Vaikuntanathan [9] transformed the decryption circuit of [17] to a branching program by using the Barrington's theorem [2], and homomorphically evaluated the program. Since the Barrington's theorem can convert the decryption circuit to a polynomial length branching program, evaluating the program increases the noise by a factor of $\mathrm{poly}(n)$. This procedure, however, has a significant drawback: the Barrington's theorem generates a branching program of *large* polynomial length. The scheme [9] also used a kind of *dimension leveraging* technique and successive dimension-modulus reduction to obtain the best approximation factor that is the same as standard lattice-based (plain) PKE.

Unlike most previous works, Alperin-Sheriff and Peikert [1] viewed the decryption as an arithmetic circuit. The decryption of all known standard lattice-based FHE consists of the inner product and rounding: for a ciphertext vector $\boldsymbol{c}$ and secret key vector $\boldsymbol{s}$, the decryption algorithm computes $\lfloor \langle \boldsymbol{c}, \boldsymbol{s} \rangle \rceil_2 \in \{0, 1\}$ (where $\lfloor \cdot \rceil_2$ is the rounding function introduced later). The authors observed that the inner product in the decryption can be expressed as a subset sum of the secret key elements. The subset sum can be computed only in the additive group, and the additive group is isomorphic to a group of cyclic permutations. The authors rewrote the inner product to the sequence of compositions of the cyclic permutations. Since this does not use the Barrington's theorem, the bootstrapping procedure of [1] can refresh ciphertexts faster and keep the noise growth in a *smaller* polynomial than that of [9], but the underlying security assumption was slightly stronger than that of [9] [3]. In addition, the procedure of [1] was not fully sequentialized, that is, there is a little room for sequentializing the decryption: the rounding.

1.1 Our Results

In this paper, we construct the first FHE scheme that encrypts matrices and supports homomorphic matrix operations. This is a natural extension of packed FHE and supports more complicated homomorphic operations. Using this scheme, we fully sequentialize and thus optimize the bootstrapping procedure of [1]. The result of the optimization is described in the following:

Theorem 1. *Our optimized bootstrapping scheme can be secure assuming the hardness of approximating the standard lattice problem to within the factor* $\tilde{O}(n^{1.5}\lambda)$ *on any* n *dimensional lattices.*

For 2^λ hardness, we need to take $n = \Omega(\lambda)$. Asymptotically minimal selection of $n = \tilde{O}(\lambda)$ leads to the approximation factor $\tilde{O}(n^{2.5})$ for the underlying worst-case lattice assumption, which is smaller than $\tilde{O}(n^3)$, the factor of [1]. Using a kind of dimension leveraging technique: selecting a larger dimension $n = \lambda^{1/\epsilon}$ for $\epsilon \in (0,1)$, we can also obtain the best known approximation factor, $\tilde{O}(n^{1.5+\epsilon})$, *without* successive dimension-modulus reduction, which was essential for achieving the best factor in the prior works on bootstrapping of standard lattice-based FHE.

1.2 Our Techniques

Matrix GSW-FHE. The starting point of our scheme is the GSW-FHE scheme. In that scheme, a ciphertext of a plaintext $m \in \{0,1\}$ is a matrix $\boldsymbol{C} \in \mathbb{Z}_q^{(n+1)\times N}$ such that $\boldsymbol{sC} = m \cdot \boldsymbol{sG} + \boldsymbol{e}$ for a secret key vector $\boldsymbol{s} \in \mathbb{Z}_q^{n+1}$, small noise vector $\boldsymbol{e} \in \mathbb{Z}^N$, and fixed matrix $\boldsymbol{G} \in \mathbb{Z}_q^{(n+1)\times N}$. A simple extension of the plaintext space from bits to binary vectors cannot yield plaintext-slot-wise addition and multiplication. Instead, we use matrices to store binary vectors in their diagonal entries. Actually, our construction even supports homomorphic matrix addition and multiplication that are richer than homomorphic plaintext-slot-wise operations.

Let $\boldsymbol{S} \in \mathbb{Z}_q^{r\times(n+r)}$ be a secret key matrix, $\boldsymbol{B} \in \mathbb{Z}_q^{n\times m}$ be a Learning with Errors (LWE) matrix such that $\boldsymbol{SB} \approx \boldsymbol{0}$, and $\boldsymbol{G} \in \mathbb{Z}^{(n+r)\times N}$ be a fixed matrix. To encrypt a square integer matrix $\boldsymbol{M} \in \{0,1\}^{r\times r}$, the ciphertext $\boldsymbol{C} \in \mathbb{Z}^{(n+r)\times N}$ must be of the form $\boldsymbol{BR} + \boldsymbol{XG}$ for a matrix $\boldsymbol{X} \in \mathbb{Z}^{(n+r)\times(n+r)}$ such that $\boldsymbol{SX} = \boldsymbol{MS}$, and small random matrix $\boldsymbol{R} \in \mathbb{Z}^{m\times N}$. The ciphertext $\boldsymbol{C}$ satisfies $\boldsymbol{SC} = \boldsymbol{E} + \boldsymbol{MSG}$ for a small noise matrix $\boldsymbol{E} \in \mathbb{Z}^{r\times N}$. Homomorphic matrix

[3] By using successive dimension-modulus reduction, [1] can also obtain the same approximation factor as that of [9].

addition is just matrix addition. For example, given two ciphertexts $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$, it holds that

$$\boldsymbol{S}(\boldsymbol{C}_1 + \boldsymbol{C}_2) = (\boldsymbol{E}_1 + \boldsymbol{E}_2) + (\boldsymbol{M}_1 + \boldsymbol{M}_2)\boldsymbol{S}\boldsymbol{G}.$$

Homomorphic matrix multiplication corresponds to a simple preimage sampling and matrix multiplication. For a matrix $\boldsymbol{C} \in \mathbb{Z}_q^{(n+r)\times N}$, let $\boldsymbol{G}^{-1}(\boldsymbol{C})$ be the function that outputs a matrix $\boldsymbol{X}' \in \mathbb{Z}_q^{N\times N}$ such that $\boldsymbol{G}\boldsymbol{X}' \equiv \boldsymbol{C} \pmod q$. If we let $\boldsymbol{X}'_2 \xleftarrow{R} \boldsymbol{G}^{-1}(\boldsymbol{C}_2)$, then it holds that

$$\begin{aligned} \boldsymbol{S}\boldsymbol{C}_1\boldsymbol{X}'_2 &= (\boldsymbol{E}_1 + \boldsymbol{M}_1\boldsymbol{S}\boldsymbol{G})\boldsymbol{X}'_2 \\ &= \boldsymbol{E}_1\boldsymbol{X}'_2 + \boldsymbol{M}_1\boldsymbol{E}_2 + \boldsymbol{M}_1\boldsymbol{M}_2\boldsymbol{S}\boldsymbol{G}. \end{aligned}$$

Now, the problem is how to construct a matrix $\boldsymbol{X}$ such that $\boldsymbol{S}\boldsymbol{X} = \boldsymbol{M}\boldsymbol{S}$. By construction, $\boldsymbol{S}$ includes an identity matrix: $\boldsymbol{S} = [\boldsymbol{I} \parallel \boldsymbol{S}']$ for a matrix $\boldsymbol{S}' \in \mathbb{Z}_q^{r\times n}$. The idea is to make $\boldsymbol{X}$ have $\boldsymbol{M}\boldsymbol{S}$ in its top rows and 0 below. This $\boldsymbol{X}$ clearly satisfies the condition, but cannot publicly be computed without knowing the secret key. We translate the resulting symmetric scheme to the asymmetric one by using the method similar to [3, 24]. In particular, let $\boldsymbol{M}_{(i,j)} \in \{0,1\}^{r\times r}$ $(i, j = 1, \ldots, r)$ be the matrix with 1 in the (i, j)-th entry and 0 in the others. We first publish symmetric encryptions of $\boldsymbol{M}_{(i,j)}$ for all $i, j \in [r]$. A ciphertext for a plaintext matrix $\boldsymbol{M}$ is publicly computed by summing up all encryptions of $\boldsymbol{M}_{(i,j)}$ such that the (i, j)-th entry of $\boldsymbol{M}$ is equal to 1, and using $\boldsymbol{B}$ to randomize the sum. Since the public key includes the ciphertexts that encrypt partial information of the secret key, security of our scheme cannot directly be proven from the LWE assumption. The way to deal with this problem is to introduce a circular security assumption.

Optimizing Bootstrapping of [1]. For a dimension d and modulus q, let $\boldsymbol{c} \in \{0,1\}^d$ be the $\ell - 1$-th column of a binary GSW-FHE ciphertext under the secret key $\boldsymbol{s} \in \mathbb{Z}_q^d$. Since the decryption algorithm of GSW-FHE computes $\lfloor\langle \boldsymbol{c}, \boldsymbol{s}\rangle\rceil_2$ ($\lfloor\cdot\rceil_2$ is the rounding function that outputs 1 if the input is close to $q/4$ and 0 otherwise), and $\langle \boldsymbol{c}, \boldsymbol{s}\rangle = \sum_{i=1}^{d} c_i s_i = \sum_{i\in[d]:c_i=1} s_i$, the decryption can be viewed as a subset sum of $\{s_i\}_{i\in[d]}$. To bootstrap ciphertexts, we only have to be able to compute additions in $\mathbb{Z}_q$ homomorphically. The additive group $\mathbb{Z}_q^+$ is isomorphic to a group of cyclic permutations, where $x \in \mathbb{Z}_q^+$ corresponds to a cyclic permutation that can be represented by an indicator vector with 1 in the x-th position. The permutation matrix can be obtained from the cyclic rotation of the indicator vector. The addition in $\mathbb{Z}_q^+$ leads to the composition of the permutations (i.e., the multiplication of the corresponding permutation matrices), and the rounding function $\lfloor\cdot\rceil_2 : \mathbb{Z}_q \to \{0,1\}$ can be computed by summing the entries of the indicator vector corresponding to those values in $\mathbb{Z}_q$.

The bootstrapping procedure of [1] consists of two parts that compute an inner product and a rounding operation. The rounding checks equalities and computes summation. The matrix GSW-FHE scheme allows us to rewrite the bootstrapping procedure except for the summation as a *sequence* of homomorphic matrix multiplications, while the procedure of [1] computes only the inner product part as a sequence. Intuitively, our optimization use the matrix GSW-FHE scheme to *sequentialize* the bootstrapping procedure of [1]. The asymmetric noise growth property is more effective in estimating how much noise the procedure yields.

The inner product can be computed by compositions of cyclic permutations. The bootstrapping procedure of [1] represents elements in $\mathbb{Z}_q$ as cyclic permutations, and evaluates their compositions by the naive matrix multiplication algorithm on the ciphertexts that encrypt every elements in the permutation matrices. Instead of that, our bootstrapping procedure uses homomorphic matrix multiplication to directly evaluate the compositions. The rounding part tests for every value close to $q/4$ whether the output of the inner product part encrypts the permutation corresponding to the value, and sums their results (that are 0 or 1). Our procedure also use homomorphic matrix multiplication to realize the equality test. The result of the inner product is represented as an indicator vector, and encrypted component-wise in a SIMD encryption. The inner product equals to x if and only if its indicator vector has 1 in the x-th position. The homomorphic equality test between the inner product and x is computed by homomorphically permuting x-th slot to the first slot in the SIMD ciphertext. The result of the test is encrypted in the first slot. From the above, the bootstrapping procedure except for the summation can be represented as a sequence of $\tilde{O}(\lambda)$ homomorphic multiplications for a security parameter λ.

1.3 Related Work

Multilinear maps [10,14,15] are extensions of bilinear maps, and built from variants of FHE. The new multilinear maps construction of Gentry, Gorbunov, and Halevi [15] also starts from GSW-FHE. Recall that in GSW-FHE, a ciphertext of $m \in \{0,1\}$ is a matrix $\boldsymbol{C} \in \mathbb{Z}_q^{(n+1)\times N}$ such that $\boldsymbol{sC} = m \cdot \boldsymbol{sG} + \boldsymbol{e}$ for a secret key vector $\boldsymbol{s} \in \mathbb{Z}_q^{(n+1)}$ and small noise vector $\boldsymbol{e} \in \mathbb{Z}^N$. That is, valid ciphertexts of GSW-FHE have the secret key as the *approximate eigenvector* and the message as the eigenvalue. The multilinear maps construction of [15] replaced the approximate eigenvector with the *approximate eigenspace* by increasing the dimension. In the construction, an encoding of $\boldsymbol{M} \in \mathbb{Z}^{r\times r}$ is a matrix $\boldsymbol{C} \in \mathbb{Z}_q^{N\times N}$ such that $\boldsymbol{SC} = \boldsymbol{E} + \boldsymbol{MS}$ for a random matrix $\boldsymbol{S} \in \mathbb{Z}_q^{r\times N}$ and small noise matrix $\boldsymbol{E} \in \mathbb{Z}^{r\times N}$. The approximate eigenspace is the matrix $\boldsymbol{S}$. To obtain the encoding $\boldsymbol{C}$, the construction samples a preimage of $\boldsymbol{MS} + \boldsymbol{E}$ for the function $f_{\boldsymbol{S}}(\boldsymbol{x}) = \boldsymbol{Sx} \bmod q$. In our scheme, a ciphertext $\boldsymbol{C} \in \mathbb{Z}_q^{N\times N}$ of $\boldsymbol{M} \in \mathbb{Z}^{r\times r}$ is a preimage of

$$\boldsymbol{BR} + \begin{pmatrix} \boldsymbol{MS} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{G}$$

for the function $f_{\boldsymbol{G}}$. Since the ciphertext $\boldsymbol{C}$ satisfies $(\boldsymbol{SG})\boldsymbol{C} = \boldsymbol{M}(\boldsymbol{SG}) + \boldsymbol{E}$ for a small noise matrix $\boldsymbol{E} \in \mathbb{Z}^{r\times N}$, the matrix $\boldsymbol{SG}$ can be seen as the approximate eigenspace.

1.4 Organization

In Section 2, we describe some preliminaries on the LWE assumption and subgaussian random variables. In Section 3, we present how to construct a matrix variant of [17]. In Section 4, we show that our scheme improves the bootstrapping procedure of [1].

Acknowledgement. We thank anonymous PKC reviewers for their helpful comments.

2 Preliminaries

We denote the set of natural numbers by $\mathbb{N}$, the set of integers by $\mathbb{Z}$, the set of rational numbers by $\mathbb{Q}$, and the set of real numbers by $\mathbb{R}$. Let $\mathbb{G}$ be some group and $\mathcal{P}$ be some probability distribution, then we use $a \overset{U}{\leftarrow} \mathbb{G}$ to denote that a is chosen from $\mathbb{G}$ uniformly at random, and use $b \overset{R}{\leftarrow} \mathcal{P}$ to denote that b is chosen along $\mathcal{P}$. We take all logarithms to base 2, unless otherwise noted.

We assume that vectors are in column form and are written by using bold lower-case letters, e.g., $\boldsymbol{x}$, and the i-th element of a vector is denoted by x_i. We denote the ℓ_∞ norm (the maximum norm) of the vector $\boldsymbol{x}$ by $\|\boldsymbol{x}\|_\infty$, and the ℓ_2 norm (the Euclidean norm) of $\boldsymbol{x}$ by $\|\boldsymbol{x}\|_2$. The inner product between two vectors is denoted by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$. Matrices are written by using bold capital letters, e.g., $\boldsymbol{X}$, and the i-th column vector of a matrix is denoted by $\boldsymbol{x}_i$. For a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, we define the ℓ_∞ and ℓ_2 norms of $\boldsymbol{X}$ as $\|\boldsymbol{X}\|_\infty := \max_{i \in [n]}\{\|\boldsymbol{x}_i\|_\infty\}$ and $\|\boldsymbol{X}\|_2 := \max_{i \in [n]}\{\|\boldsymbol{x}_i\|_2\}$, respectively. For a matrix $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, the notation $\boldsymbol{X}^T \in \mathbb{R}^{n \times m}$ denotes the transpose of $\boldsymbol{X}$. For matrices $\boldsymbol{A} \in \mathbb{R}^{m \times n_1}$ and $\boldsymbol{B} \in \mathbb{R}^{m \times n_2}$, $[\boldsymbol{A} \parallel \boldsymbol{B}] \in \mathbb{R}^{m \times (n_1 + n_2)}$ denotes the concatenation of $\boldsymbol{A}$ with $\boldsymbol{B}$. When we refer to the $n \times n$ identity matrix, we denote it by $\boldsymbol{I}_n$.

2.1 Learning with Errors

The *learning with errors (LWE) assumption* was first introduced by Regev [23].

Definition 1 (DLWE). *For a security parameter λ, let $n := n(\lambda)$ be an integer dimension, let $q := q(\lambda) \geq 2$ be an integer modulus, and let $\chi := \chi(\lambda)$ be an error distribution over $\mathbb{Z}$. $\mathsf{DLWE}_{n,q,\chi}$ is the problem to distinguish the following two distributions: In the first distribution, a tuple $(\boldsymbol{a}_i, b_i)$ is sampled from uniform over $\mathbb{Z}_q^n \times \mathbb{Z}_q$; In the second distribution, $\boldsymbol{s} \overset{U}{\leftarrow} \mathbb{Z}_q^n$ and then a tuple $(\boldsymbol{a}_i, b_i)$ is sampled by sampling $\boldsymbol{a}_i \overset{U}{\leftarrow} \mathbb{Z}_q^n$, $e_i \overset{R}{\leftarrow} \chi$, and setting $b_i := \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i \bmod q$. The $\mathsf{DLWE}_{n,q,\chi}$ assumption is that $\mathsf{DLWE}_{n,q,\chi}$ is infeasible.*

Recall that GapSVP_γ is the promise problem to distinguish between the case in which the lattice has a vector shorter than $r \in \mathbb{Q}$, and the case in which all the lattice vectors are greater that $\gamma \cdot r$. SIVP_γ is the problem to find the set of short linearly independent vectors in a lattice. $\mathsf{DLWE}_{n,q,\chi}$ has reductions to the standard lattice assumptions as follows. These reductions take χ to be a discrete Gaussian distribution $D_{\mathbb{Z},\alpha q}$ (that is centered around 0 and has parameter αq for some $\alpha < 1$), which is statistically indistinguishable from a B-bounded distribution (i.e., $\mathbb{E}[X] = 0$ and $|X| \leq B$) for an appropriate B.

Corollary 1 ([19–21, 23]). *Let $q := q(n) \in \mathbb{N}$ be a power of primes $q := p^r$ or a product of distinct prime numbers $q := \prod_i q_i$ ($q_i := \mathrm{poly}(n)$ for all i), and let $\alpha \geq \sqrt{n}/q$. If there exists an efficient algorithm that solves (average-case) $\mathsf{DLWE}_{n,q,D_{\mathbb{Z},\alpha q}}$,*

- *there exists an efficient quantum algorithm that can solve $\mathsf{GapSVP}_{\tilde{O}(n/\alpha)}$ and $\mathsf{SIVP}_{\tilde{O}(n/\alpha)}$ in the worst-case for any n-dimensional lattices.*
- *if in addition we have $q \geq \tilde{O}(2^{n/2})$, there exists an efficient classical algorithm that can solve $\mathsf{GapSVP}_{\tilde{O}(n/\alpha)}$ in the worst-case for any n-dimensional lattices.*

2.2 Subgaussian

A real random variable X is subgaussian with parameter s if for all $t \in \mathbb{R}$, its (scaled) moment generating function holds $\mathbb{E}[\exp(2\pi t X)] \leq \exp(\pi s^2 t^2)$. Any B-bounded (centered) random variable X is subgaussian with parameter $B \cdot \sqrt{2\pi}$. Subgaussian random variables have the following two properties that can be easily obtained from the definition of subgaussian random variables:

- Homogeneity: If the subgaussian random variable X has parameter s, then cX is subgaussian with parameter cs.
- Pythagorean additivity: For two subgaussian random variables X_1 and X_2 (that is independent from X_1) with parameter s_1 and s_2, respectively, $X_1 + X_2$ is subgaussian with parameter $\sqrt{s_1^2 + s_2^2}$.

The above can be extended to vectors. A real random vector $\boldsymbol{x}$ is subgaussian with parameter s if for all real unit vectors $\boldsymbol{u}$, their marginal $\langle \boldsymbol{u}, \boldsymbol{x} \rangle$ is subgaussian with parameter s. It is clear from the definition that the concatenation of subgaussian variables or vectors, each of which has a parameter s and is independent of the prior one, is also subgaussian with parameter s. The homogeneity and Pythagorean additivity also hold from linearity of vectors. It is known that the euclidean norm of the subgaussian random vector has the following upper bound.

Lemma 1 ([26]). *Let $\boldsymbol{x} \in \mathbb{R}^n$ be a random vector that has independent subgaussian coordinates with parameter s. Then there exists a universal constant C such that $\Pr[\|\boldsymbol{x}\|_2 > C \cdot s\sqrt{n}] \leq 2^{-\Omega(n)}$.*

To suppress the growth in noise, Gentry et al. [17] made use of a procedure that decomposes a vector in binary representation. Alperin-Sheriff and Peikert [1] observed that instead of the decomposition procedure, using the following algorithm $\boldsymbol{G}^{-1}$ that samples a subgaussian random vector allows us to re-randomize errors in ciphertexts and tightly analyze the noise growth in [17]. Lemma 2 can be extended to matrices in the obvious way. Let $\boldsymbol{g}^T := (1, 2, 2^2, \ldots, 2^{\lceil \log q \rceil - 1})$ and $\boldsymbol{G} := \boldsymbol{g}^T \otimes \boldsymbol{I}_n$.

Lemma 2 ([1], which is adapted from [20]). *There is a randomized, efficiently computable function* $\boldsymbol{G}^{-1} : \mathbb{Z}_q^n \to \mathbb{Z}^{n \cdot \lceil \log q \rceil}$ *such that for any* $\boldsymbol{a} \in \mathbb{Z}_q^n$, $\boldsymbol{x} \overset{R}{\leftarrow} \boldsymbol{G}^{-1}(\boldsymbol{a})$ *is subgaussian with parameter* $O(1)$ *and* $\boldsymbol{a} = [\boldsymbol{G}\boldsymbol{x}]_q$

2.3 Homomorphic Encryption, Circular Security, and Bootstrapping

Here we describe the syntax of homomorphic encryption scheme to introduce a definition of circular security and the Gentry's bootstrapping theorem. Let $\mathcal{M}$ and $\mathcal{C}$ be the message and ciphertext space. A homomorphic encryption scheme consists of four algorithms, $\{\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}\}$.

- $\mathsf{KeyGen}(1^\lambda)$: output a public encryption key pk, a secret decryption key sk, and a public evaluation key evk.
- $\mathsf{Enc}_{\mathsf{pk}}(m)$: using a public key pk, encrypt a plaintext $m \in \mathcal{M}$ into a ciphertext $c \in \mathcal{C}$.
- $\mathsf{Dec}_{\mathsf{sk}}(c)$: using a secret key sk, recover the message encrypted in the ciphertext c.
- $\mathsf{Eval}_{\mathsf{evk}}(f, c_1, \ldots, c_\ell)$: using the evaluation key evk, output a ciphertext $c_f \in \mathcal{C}$ that is obtained by applying the function $f : \mathcal{M}^\ell \to \mathcal{M}$ to $c_1, \ldots, c_\ell$.

To prove the security of our construction, we introduce a special kind of circular security for a homomorphic encryption scheme.

Definition 2 (Circular security). *Let* $\mathcal{K}$ *be the key space defined by a security parameter* λ*. Let* f *be a function from* $\mathcal{K}$ *to* $\mathcal{C}$*. A homomorphic encryption scheme* $\mathsf{HE} = \{\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}\}$ *is circular secure with respect to* f *if for all probabilistic polynomial-time adversary* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the following game is negligible in* λ*:*

1. *A challenger computes* $(\mathsf{pk}, \mathsf{sk}, \mathsf{evk}) \overset{R}{\leftarrow} \mathsf{KeyGen}(1^\lambda)$*, and chooses a bit* $b \overset{U}{\leftarrow} \{0,1\}$.
2. *Let* $f_+ : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ *be a function that computes* $f_+(x,y) := x + y \in \mathcal{M}$*. The challenger computes a challenge ciphertext* c^* *as follows and sends it to* $\mathcal{A}$*.*
$$c^* := \begin{cases} \mathsf{Eval}_{\mathsf{evk}}(f_+, \mathsf{Enc}_{\mathsf{pk}}(0), f(\mathsf{sk})) & \text{if } b = 0, \\ \mathsf{Enc}_{\mathsf{pk}}(0) \in \mathcal{C} & \text{otherwise.} \end{cases}$$
3. $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$.

The advantage of $\mathcal{A}$ *is* $\Pr[b = b'] - 1/2$.

In LWE-based FHE schemes, $\mathsf{Eval}_{\mathsf{evk}}(f_+, \mathsf{Enc}_{\mathsf{pk}}(0), f(\mathsf{sk}))$ can be seen as a kind of ciphertexts that encrypt $f(\mathsf{sk})$. This is why we call the above security notion circular security.

Gentry's bootstrapping theorem states the way to go from limited homomorphism to unlimited homomorphism. This relates to augmented decryption circuits.

Definition 3 (Augmented decryption circuit). *Let* $(\mathsf{sk}, \mathsf{pk}, \mathsf{evk})$ *be a tuple of keys generated appropriately, and* $\mathcal{C}$ *be the set of decryptable ciphertexts. Then the set of augmented decryption functions* $\{f_{c_1,c_2}\}_{c_1,c_2\in\mathcal{C}}$ *is defined by*

$$f_{c_1,c_2}(x) = \overline{\mathsf{Dec}_x(c_1) \wedge \mathsf{Dec}_x(c_1)}.$$

That is, the function uses its input as the secret key, decrypts c_1 *and* c_2*, and returns the NAND of the results.*

Theorem 2 (Bootstrapping theorem [12, 13]). *A scheme that can evaluate the family of the augmented decryption circuits can be transformed into a "leveled" FHE scheme (in which* KeyGen *takes as additional input* 1^L *and we can only evaluate depth* L *circuits) with the same decryption circuit, ciphertext space, and public key.*

In addition, if the above scheme is weak circular secure (remains secure against an adversary that can obtain encryptions of the bits of the secret key), it can be "pure" FHE scheme (in which the number of homomorphic evaluations is unlimited).

3 Matrix GSW-FHE

We translate [17] to be able to encrypt a *matrix* and homomorphically compute *matrix* addition and multiplication. This is a natural extension of packed FHE schemes. In Section 3.1, we present our matrix FHE scheme. In Section 3.2, we discuss the relationship between our scheme and packed FHE schemes.

3.1 Construction

Let λ be the security parameter. Our scheme is parameterized by an integer lattice dimension n, an integer modulus q, and a distribution χ over $\mathbb{Z}$ that is assumed to be subgaussian , all of which depends on λ. We let $\ell := \lceil \log q \rceil$, $m := O((n+r)\log q)$, and $N := (n+r)\cdot \ell$. Let r be the number of bits to be encrypted, which defines the message space $\{0,1\}^{r\times r}$. The ciphertext space is $\mathbb{Z}_q^{(n+r)\times N}$. Our scheme uses the rounding function $\lfloor \cdot \rceil_2$ that for any $x \in \mathbb{Z}_q$, $\lfloor x \rceil_2$ outputs 1 if x is close to $q/4$, and 0 otherwise. Recall that $\boldsymbol{g}^T = (1, 2, \ldots, 2^{\ell-1})$ and $\boldsymbol{G} = \boldsymbol{g}^T \otimes \boldsymbol{I}_{n+r}$.

- $\mathsf{KeyGen}(1^\lambda, r)$: Set the parameters n, q, m, ℓ, N, and χ as described above. Sample a uniformly random matrix $\boldsymbol{A} \overset{U}{\longleftarrow} \mathbb{Z}_q^{n\times m}$, secret key matrix $\boldsymbol{S}' \overset{R}{\longleftarrow} \chi^{r\times n}$, and noise matrix $\boldsymbol{E} \overset{R}{\longleftarrow} \chi^{r\times m}$. Let $\boldsymbol{S} := [\boldsymbol{I}_r \parallel -\boldsymbol{S}'] \in \mathbb{Z}_q^{r\times(n+r)}$. We denote by $\boldsymbol{s}_i^T$ the i-th row of $\boldsymbol{S}$. Set
$$\boldsymbol{B} := \left(\frac{\boldsymbol{S}'\boldsymbol{A} + \boldsymbol{E}}{\boldsymbol{A}}\right) \in \mathbb{Z}_q^{(n+r)\times m}.$$

Let $\boldsymbol{M}_{(i,j)} \in \{0,1\}^{r\times r}$ $(i,j = 1,\dots,r)$ be the matrix with 1 in the (i,j)-th position and 0 in the others. For all $i,j = 1,\dots,r$, first sample $\boldsymbol{R}_{(i,j)} \xleftarrow{U} \{0,1\}^{m\times N}$, and set

$$\boldsymbol{P}_{(i,j)} := \boldsymbol{B}\boldsymbol{R}_{(i,j)} + \begin{pmatrix} \boldsymbol{M}_{(i,j)}\boldsymbol{S} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{G} \in \mathbb{Z}_q^{(n+r)\times N}.$$

Output $\mathsf{pk} := (\{\boldsymbol{P}_{(i,j)}\}_{i,j\in[r]}, \boldsymbol{B})$ and $\mathsf{sk} := \boldsymbol{S}$.

- $\mathsf{SecEnc}_{\mathsf{sk}}(\boldsymbol{M} \in \{0,1\}^{r\times r})$: Sample a random matrices $\boldsymbol{A}' \xleftarrow{U} \mathbb{Z}_q^{n\times N}$ and $\boldsymbol{E} \xleftarrow{R} \chi^{r\times N}$, parse $\boldsymbol{S} = [\boldsymbol{I}_r \parallel -\boldsymbol{S}']$, and output the ciphertext

$$\boldsymbol{C} := \left[\begin{pmatrix} \boldsymbol{S}'\boldsymbol{A}' + \boldsymbol{E} \\ \boldsymbol{A}' \end{pmatrix} + \begin{pmatrix} \boldsymbol{M}\boldsymbol{S} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{G} \right]_q \in \mathbb{Z}_q^{(n+r)\times N}.$$

- $\mathsf{PubEnc}_{\mathsf{pk}}(\boldsymbol{M} \in \{0,1\}^{r\times r})$: Sample a random matrix $\boldsymbol{R} \xleftarrow{U} \{0,1\}^{m\times N}$, and output the ciphertext

$$\boldsymbol{C} := \boldsymbol{B}\boldsymbol{R} + \sum_{i,j\in[r]:\boldsymbol{M}[i,j]=1} \boldsymbol{P}_{(i,j)} \in \mathbb{Z}_q^{(n+r)\times N},$$

where $\boldsymbol{M}[i,j]$ is the (i,j)-th element of $\boldsymbol{M}$.
- $\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{C})$: Output the matrix $\boldsymbol{M} = (\lfloor \langle \boldsymbol{s}_i, \boldsymbol{c}_{j\ell-1} \rangle \rceil_2)_{i,j\in[r]} \in \{0,1\}^{r\times r}$.
- $\boldsymbol{C}_1 \oplus \boldsymbol{C}_2$: Output $\boldsymbol{C}_{add} := \boldsymbol{C}_1 + \boldsymbol{C}_2 \in \mathbb{Z}_q^{(n+r)\times N}$ as the result of homomorphic addition between the input ciphertexts.
- $\boldsymbol{C}_1 \odot \boldsymbol{C}_2$: Output $\boldsymbol{C}_{mult} := \boldsymbol{C}_1 \boldsymbol{G}^{-1}(\boldsymbol{C}_2) \in \mathbb{Z}_q^{(n+r)\times N}$ as the result of homomorphic multiplication between the input ciphertexts.

Definition 4. *We say that a ciphertext $\boldsymbol{C}$ encrypts a plaintext matrix $\boldsymbol{M}$ with noise matrix $\boldsymbol{E}$ if $\boldsymbol{C}$ is an encryption of $\boldsymbol{M}$ and $\boldsymbol{E} = \boldsymbol{S}\boldsymbol{C} - \boldsymbol{M}\boldsymbol{S}\boldsymbol{G} \pmod q$.*

The following lemma states the correctness of our asymmetric encryption. Similar to this, the correctness of our symmetric encryption can be proven immediately.

Lemma 3. *If a ciphertext $\boldsymbol{C}$ encrypts a plaintext matrix $\boldsymbol{M} \in \{0,1\}^{r\times r}$ with noise matrix $\boldsymbol{E}$ such that $\|\boldsymbol{E}\|_\infty < q/8$, then $\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{C}) = \boldsymbol{M}$.*

Proof. We have

$$\begin{aligned} \boldsymbol{S}\boldsymbol{C} &= \boldsymbol{S}\left(\boldsymbol{B}\boldsymbol{R} + \textstyle\sum_{i,j\in[r]:\boldsymbol{M}[i,j]=1} \boldsymbol{B}\boldsymbol{R}_{(i,j)} + \begin{pmatrix} \boldsymbol{M}\boldsymbol{S} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{G} \right) \\ &= \boldsymbol{E}\boldsymbol{R} + \textstyle\sum_{i,j\in[r]:\boldsymbol{M}[i,j]=1} \boldsymbol{E}\boldsymbol{R}_{(i,j)} + \boldsymbol{M}\boldsymbol{S}\boldsymbol{G} \\ &= \boldsymbol{E}\boldsymbol{R} + \textstyle\sum_{i,j\in[r]:\boldsymbol{M}[i,j]=1} \boldsymbol{E}\boldsymbol{R}_{(i,j)} + [\boldsymbol{M}(\boldsymbol{g}^T \otimes \boldsymbol{I}_r) \parallel -\boldsymbol{M}\boldsymbol{S}'(\boldsymbol{g}^T \otimes \boldsymbol{I}_n)] \end{aligned}$$

Because of $\|\boldsymbol{E}(\boldsymbol{R} + \sum_{i,j\in[r]:\boldsymbol{M}[i,j]=1} \boldsymbol{R}_{(i,j)})\|_\infty < q/8$ and $2^{\ell-2} \in [q/4, q/2)$, for all $i,j = 1,\dots,r$, $\langle \boldsymbol{s}_i, \boldsymbol{c}_{j\ell-1} \rangle \approx q/4$ if $m_{i,j} = 1$, and $\langle \boldsymbol{s}_i, \boldsymbol{c}_{j\ell-1} \rangle \approx 0$ otherwise.

Security of SecEnc directly holds from $\mathsf{DLWE}_{n,q,\chi}$. For a matrix $\boldsymbol{M} \in \{0,1\}^{r\times r}$, let $f_{\boldsymbol{M}}$ be a function from $\mathbb{Z}_q^{r\times(n+r)}$ to $\mathbb{Z}_q^{(n+r)\times N}$ such that for a matrix $\boldsymbol{S} \in \mathbb{Z}_q^{r\times(n+r)}$,

$$f_{\boldsymbol{M}}(\boldsymbol{S}) = \left(\frac{\boldsymbol{MS}}{\boldsymbol{0}}\right)\boldsymbol{G} \in \mathbb{Z}_q^{(n+r)\times N}.$$

The security of PubEnc directly holds by $\mathsf{DLWE}_{n,q,\chi}$ and assuming our scheme circular secure with respect to $f_{\boldsymbol{M}_{(i,j)}}$.

Lemma 4. *Let $\boldsymbol{B}, \boldsymbol{M}_{(i,j)}, \boldsymbol{R}_{(i,j)}, \boldsymbol{P}_{(i,j)}$ $(i,j = 1,\ldots,r)$ be the matrices generated in KeyGen, and $\boldsymbol{R}$ be the matrix generated in PubEnc. For every $i,j = 1,\ldots,r$, if our scheme is circular secure with respect to $f_{\boldsymbol{M}_{(i,j)}}$ and $\mathsf{DLWE}_{n,q,\chi}$ holds, then the joint distribution $(\boldsymbol{B}, \boldsymbol{BR}_{(i,j)}, \boldsymbol{P}_{(i,j)}, \boldsymbol{BR})$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{(n+r)\times m} \times \mathbb{Z}_q^{(n+r)\times N} \times \mathbb{Z}_q^{(n+r)\times N} \times \mathbb{Z}_q^{(n+r)\times N}$.*

We need to estimate the noise growth by the evaluation of homomorphic matrix addition and multiplication. Similar to [1], we employ the properties of subgaussian random variables for tight analysis. We collect the results of the estimation in the following lemma.

Lemma 5. *Let $\boldsymbol{S} \in \mathbb{Z}^{r\times(n+r)}$ be a secret key matrix. Let $\boldsymbol{C}_1 \in \mathbb{Z}_q^{(n+r)\times N}$ and $\boldsymbol{C}_2 \in \mathbb{Z}_q^{(n+r)\times N}$ be ciphertexts that encrypt $\boldsymbol{M}_1 \in \{0,1\}^{r\times r}$ and $\boldsymbol{M}_2 \in \{0,1\}^{r\times r}$ with noise matrices $\boldsymbol{E}_1 \in \mathbb{Z}^{r\times N}$ and $\boldsymbol{E}_2 \in \mathbb{Z}^{r\times N}$, respectively. Let $\boldsymbol{e}_{1,i}^T \in \mathbb{Z}^{1\times N}$ $(i = 1,\ldots,r)$ be the i-th row vector of $\boldsymbol{E}_1$. Let $\boldsymbol{C}_{add} := \boldsymbol{C}_1 \oplus \boldsymbol{C}_2$ and $\boldsymbol{C}_{mult} \overset{R}{\leftarrow} \boldsymbol{C}_1 \odot \boldsymbol{C}_2$. Then, we have*

$$\begin{aligned}\boldsymbol{SC}_{add} &= \boldsymbol{E}_{add} + (\boldsymbol{M}_1 + \boldsymbol{M}_2)\boldsymbol{SG} \in \mathbb{Z}_q^{r\times N},\\ \boldsymbol{SC}_{mult} &= \boldsymbol{E}_{mult} + (\boldsymbol{M}_1\boldsymbol{M}_2)\boldsymbol{SG} \in \mathbb{Z}_q^{r\times N},\end{aligned}$$

where $\boldsymbol{E}_{add} := \boldsymbol{E}_1 + \boldsymbol{E}_2$ and $\boldsymbol{E}_{mult} := \boldsymbol{E} + \boldsymbol{M}_1\boldsymbol{E}_2$. In particular, $\boldsymbol{E}$ has in the i-th row the independent subgaussian entries with parameter $O(\|\boldsymbol{e}_{1,i}\|_2)$.

Proof. We can immediately prove the statements for $\boldsymbol{C}_{add}$. For $\boldsymbol{C}_{mult}$, we have

$$\begin{aligned}\boldsymbol{SC}_{mult} &= \boldsymbol{SC}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)\\ &= (\boldsymbol{E}_1 + \boldsymbol{M}_1\boldsymbol{SG})\boldsymbol{G}^{-1}(\boldsymbol{C}_2)\\ &= \boldsymbol{E}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2) + \boldsymbol{M}_1\boldsymbol{E}_2 + \boldsymbol{M}_1\boldsymbol{M}_2\boldsymbol{SG}.\end{aligned}$$

From the subgaussian properties and Lemma 2, we can see that the i-th row entries of $\boldsymbol{E} := \boldsymbol{E}_1\boldsymbol{G}^{-1}(\boldsymbol{C}_2)$ are independent subgaussian with parameter $O(\|\boldsymbol{e}_{1,i}\|_2)$.

Similar to the original GSW scheme, our scheme also has the asymmetric noise growth property, and thereby computing a polynomial length chain of homomorphic multiplications incurs the noise growth by a multiplicative polynomial factor. For ease of analyzing our optimized bootstrapping procedure described in the next section, we set the following corollary immediately proven from Lemma 5 and the properties of subgaussian random variables. This corollary includes the fixed ciphertext $\boldsymbol{G} \in \mathbb{Z}^{(n+r)\times N}$ of the message $\boldsymbol{I}_r$ with noise $\mathbf{0}$. This makes the noise in the output ciphertext subgaussian and independent from the noise in the input ciphertexts.

Corollary 2. *For $i = 1, \ldots, k$, let $\boldsymbol{C}_i \in \boldsymbol{Z}^{(n+r)\times N}$ be a ciphertext that encrypts a message matrix $\boldsymbol{M}_i \in \{0,1\}^{r\times r}$ such that for a matrix $\boldsymbol{E} \in \mathbb{Z}^{r\times N}$, $\|(\boldsymbol{M}_i\boldsymbol{E})^T\|_2 \leq \|\boldsymbol{E}^T\|_2$ with noise matrix $\boldsymbol{E}_i \in \mathbb{Z}^{r\times N}$. Let*

$$\boldsymbol{C} \overset{R}{\leftarrow} \bigodot_{i=1}^{k} \boldsymbol{C}_i \odot \boldsymbol{G} = \boldsymbol{C}_1 \odot (\boldsymbol{C}_2 \odot (\cdots (\boldsymbol{C}_{k-1} \odot (\boldsymbol{C}_k \odot \boldsymbol{G}))) \cdots).$$

For $i = 1, \ldots, k$, let $\boldsymbol{e}_i^T$ be a row vector of $\boldsymbol{E}_i$ whose norm is equal to $\|\boldsymbol{E}_i^T\|_2$, and $\boldsymbol{e}^T := [\boldsymbol{e}_1^T \parallel \boldsymbol{e}_2^T \parallel \cdots \parallel \boldsymbol{e}_k^T] \in \mathbb{Z}^{1\times kN}$. Then the noise matrix of $\boldsymbol{C}$ has in every row the independent subgaussian entries with parameter $O(\|\boldsymbol{e}\|_2)$.

Proof. The ciphertext $\boldsymbol{C}$ encrypts the message $\prod_{i=1}^{k} \boldsymbol{M}_i$ with noise $\boldsymbol{E}_1\boldsymbol{X}_1 + \sum_{i=2}^{k}(\prod_{j=1}^{i-1} \boldsymbol{M}_j)\boldsymbol{E}_i\boldsymbol{X}_i$, where $\boldsymbol{X}_i$ is the matrix used in the evaluation of each $\odot$. By Lemma 5, the elements of $\boldsymbol{E}_1\boldsymbol{X}_1$ in every row are independent and subgaussian with parameter $O(\|\boldsymbol{e}_1\|_2)$. Since we have $\|(\boldsymbol{M}_i\boldsymbol{E})^T\|_2 \leq \|\boldsymbol{E}^T\|_2$, $(\prod_{j=1}^{i-1} \boldsymbol{M}_j)\boldsymbol{E}_i\boldsymbol{X}_i$ has in its every row the independent subgaussian entries with parameter $O(\|\boldsymbol{e}_i\|_2)$. By the Pythagorean additivity of subgaussian random variables, $\boldsymbol{E}_1\boldsymbol{X}_1+\sum_{i=2}^{k}(\prod_{j=1}^{i-1} \boldsymbol{M}_j)\boldsymbol{E}_i\boldsymbol{X}_i$ has in every row the independent subgaussian entries with parameter $O(\|\boldsymbol{e}\|_2)$.

3.2 Relation to Packed FHE

The matrix GSW-FHE above is a natural extension of packed FHE. Plaintext slots in packed FHE correspond to diagonal entries of plaintext matrices in the matrix GSW-FHE scheme. It is easy to see that we can correctly compute homomorphic slot-wise addition and multiplication. In applications of packed FHE such as in [16], we may want to permute plaintext slots. This can be achieved by multiplying the encryptions of a permutation and its inverse from left and right. Security and correctness of the following algorithms clearly holds from Lemmas 4 and 5.

- $\mathsf{SwitchKeyGen}(\boldsymbol{S}, \sigma)$: Given a secret key matrix $\boldsymbol{S} \in \mathbb{Z}_q^{r\times(n+r)}$ and a permutation σ, let $\Sigma \in \{0,1\}^{r\times r}$ be a matrix corresponding to σ, and generate

$$\boldsymbol{W}_\sigma \overset{R}{\leftarrow} \mathsf{SecEnc}_{\boldsymbol{S}}(\Sigma),$$
$$\boldsymbol{W}_{\sigma^{-1}} \overset{R}{\leftarrow} \mathsf{SecEnc}_{\boldsymbol{S}}(\Sigma^T).$$

  Output the switch key $\mathsf{ssk}_\sigma := (\boldsymbol{W}_\sigma, \boldsymbol{W}_{\sigma^{-1}})$.
- $\mathsf{SlotSwitch}_{\mathsf{ssk}_\sigma}(\boldsymbol{C})$: Take as input a switch key ssk_σ and a ciphertext $\boldsymbol{C}$, output

$$\boldsymbol{C}_\sigma \overset{R}{\leftarrow} \boldsymbol{W}_\sigma \odot (\boldsymbol{C} \odot (\boldsymbol{W}_{\sigma^{-1}} \odot \boldsymbol{G})),$$

  where $\boldsymbol{G} \in \mathbb{Z}^{(n+r)\times N}$ is the fixed encryption of $\boldsymbol{I}_r$ with noise zero.

One nice feature of our plaintext-slot switching is that it does not suffer from the inconvenience of the security as in [4]: we do not have to use a larger modulus than the matrix GSW-FHE scheme. Brakerski et al. [4] made use of a larger modulus $Q = 2^{\ell} q$ to suppress noise growth when switching decryption keys, so the security of the plaintext-slot switching in [4] must have related to Q. The larger modulus leads the larger modulus-to-noise ratio. To obtain the same security level as the SIMD scheme of [4], it was required to select a larger dimension. As opposed to this, our plaintext-slot switching can use the same modulus as the matrix GSW-FHE scheme.

4 Optimizing Bootstrapping

We describe how to optimize the bootstrapping procedure of [1] by using our scheme. In Section4.1, we present the optimized bootstrapping procedure outlined in Section 1.2, whose correctness and security are discussed in Section 4.2.

4.1 Optimized Procedure

Let Q be the modulus of the ciphertext to be refreshed. Using the dimension-modulus reduction technique [7, 9], we can publicly switch the modulus and the dimension to the arbitrary and possibly smaller ones $q, d = \tilde{O}(\lambda)$. Here, q has the form $q := \prod_{i=1}^{t} r_i$, where r_i are small and powers of distinct primes (and hence pairwise coprime). The following lemma allows us to choose a sufficiently large q that the correctness of the dimension-modulus reduction holds by letting it be the product of all maximal prime powers r_i bounded by $O(\log \lambda)$, and then there exists $t = O(\log \lambda / \log \log \lambda)$.

Lemma 6 ([1]). *For all $x \geq 7$, the product of all maximal prime powers $r_i \leq x$ is at least* $\exp(3x/4)$.

By CRT, $\mathbb{Z}_q^+$ is isomorphic to the direct product $\mathbb{Z}_{r_1}^+ \times \cdots \times \mathbb{Z}_{r_t}^+$. For all $i \in [t]$, $x \in \mathbb{Z}_{r_i}^+$ corresponds to a cyclic permutation that can be represented by a indicator vector with 1 in the x-th position. Let $\phi_i : \mathbb{Z}_q \to \{0, 1\}^r$ be the isomorphism of an element in $\mathbb{Z}_q$ into the cyclic permutation that corresponds to an element in $\mathbb{Z}_{r_i}$, where $r := \max_i \{r_i\}$.

Our optimized bootstrapping procedure consists of two algorithms, $\mathsf{BootKeyGen}$ and $\mathsf{Bootstrap}$. The procedure can be used to refresh ciphertexts of all known standard LWE-based FHE. We achieve the input ciphertext $\boldsymbol{c} \in \{0, 1\}^d$ for $\mathsf{Bootstrap}$ from the dimension-modulus reduction and bit-decomposition of the ciphertext to be bootstrapped, and let $\boldsymbol{s} \in \mathbb{Z}_q^d$ be a secret key that corresponds to $\boldsymbol{c}$. This pre-processing is the same as that in [1], so see for further details.

- $\mathsf{BootKeyGen}(\mathsf{sk}, \boldsymbol{s})$: given a secret key sk for our scheme and the secret key $\boldsymbol{s} \in \mathbb{Z}_q^d$ for ciphertexts to be refreshed, output a bootstrapping key. For every

$i \in [t]$ and $j \in [d]$, let $\pi_{\phi_i(s_j)}$ be the permutation corresponding to $\phi_i(s_j)$, and compute

$$\tau_{i,j} \xleftarrow{R} \mathsf{SecEnc}_{\mathsf{sk}}(\mathsf{diag}(\phi_i(s_j))),$$
$$\mathsf{ssk}_{i,j} \xleftarrow{R} \mathsf{SwitchKeyGen}(\mathsf{sk}, \pi_{\phi_i(s_j)}),$$

where for a vector $\boldsymbol{x} \in \mathbb{Z}^r$, $\mathsf{diag}(\boldsymbol{x}) \in \mathbb{Z}^{r \times r}$ is the square integer matrix that has $\boldsymbol{x}$ in its diagonal entries and 0 in the others. In addition, we generate hints to check the equality on packed indicator vectors. For every $i \in [t]$, and $x \in \mathbb{Z}_q$ such that $\lfloor x \rceil_2 = 1$ [4], generate

$$\mathsf{ssk}_{\phi_i(x)} \xleftarrow{R} \mathsf{SwitchKeyGen}(\mathsf{sk}, \pi_{\phi_i(x)}),$$

where $\pi_{\phi_i(x)}$ is the cyclic permutation that maps the $(x \bmod r_i)$-th row to the first row in the matrix. To mask the first plaintext slot, generate an encryption of $(1, 0, \dots, 0)$:

$$\boldsymbol{P}_{(1,0,\dots,0)} \xleftarrow{R} \mathsf{SecEnc}_{\mathsf{sk}}(\mathsf{diag}((1, 0, \dots, 0))).$$

Output the bootstrapping key

$$\mathsf{bk} := \{(\tau_{i,j}, \mathsf{ssk}_{i,j}, \boldsymbol{P}_{(1,0,\dots,0)}, \mathsf{ssk}_{\phi_i(x)})\}_{i \in [t], j \in [d], x \in \mathbb{Z}_q : \lfloor x \rceil_2 = 1}.$$

- $\mathsf{Bootstrap}_{\mathsf{bk}}(\boldsymbol{c})$: Given a bootstrapping key bk and a ciphertext $\boldsymbol{c} \in \mathbb{Z}_q^d$, output the refreshed ciphertext $\boldsymbol{C}^*$. The decryption of all FHE based on the standard LWE computes $\lfloor \langle \boldsymbol{c}, \boldsymbol{s} \rangle \rceil_2$. The algorithm $\mathsf{Bootstrap}$ consists of two phases that evaluate the inner product and rounding.

  Inner Product: For every $i \in [t]$, homomorphically compute an encryption of $\phi_i(\langle \boldsymbol{c}, \boldsymbol{s} \rangle)$. Let $h := \min\{j \in [d] : c_j = 1\}$. For $i = 1, \dots, t$, set $\boldsymbol{C}_i^* := \tau_{i,h}$, and iteratively compute

  $$\boldsymbol{C}_i^* \xleftarrow{R} \mathsf{SlotSwitch}_{ssk_{i,j}}(\boldsymbol{C}_i^*)$$

  for $j = h + 1, \dots, d$ such that $c_j = 1$.

  Rounding: For each $x \in \mathbb{Z}_q$ such that $\lfloor x \rceil_2 = 1$, homomorphically check the equality between x and $\langle \boldsymbol{c}, \boldsymbol{s} \rangle$, and sum their results. The refreshed ciphertext is comuted as:

  $$\boldsymbol{C}^* \xleftarrow{R} \bigoplus_{x \in \mathbb{Z}_q \,:\, \lfloor x \rceil_2 = 1} \left(\bigodot_{i \in [t]} \left(\mathsf{SlotSwitch}_{\mathsf{ssk}_{\phi_i(x)}}(\boldsymbol{C}_i^*) \right) \odot \boldsymbol{P}_{(1,0,\dots,0)} \right). \quad (1)$$

The post-processing is almost the same as that in [1] except for the way to extract a matrix ciphertext. When finishing the bootstrapping procedure, we have a ciphertext $\boldsymbol{C}^*$ that encrypts in the first slot the same plaintext as the ciphertext $\boldsymbol{c}$. A vector ciphertext like [5, 6, 8] can be obtained to just take the

[4] Obviously, our procedure can work on not only the rounding function $\lfloor \cdot \rceil_2$ but also some arbitrary functions $f : \mathbb{Z}_q \to \{0, 1\}$.

$\ell-1$-th column vector of $\boldsymbol{C}^*$, and a matrix ciphertext like [17] can be obtained by removing from the second row to the r-th row and from the $l+1$-th column to rl-th column, and aggregating the remainders. We can utilize the key-switching procedure [5, 8] for switching from $\boldsymbol{s}_1$ back to the original secret key $\boldsymbol{s}$. This requires us to assume circular security.

Our bootstrapping procedure is more time- and space- efficient than that of [1]. The procedure [1] encrypts every elements of the permutation matrices corresponding to the secret key elements, and homomorphically evaluates naive matrix multiplications to obtain encryptions of compositions of permutations. In our procedure, a permutation is encrypted in one ciphertext, and a composition is computed by two homomorphic multiplications. This makes our procedure time-efficient by roughly a $O(\log^2\lambda)$ factor, and space-efficient by a $O(\log\lambda)$ factor.

4.2 Correctness and Security

From the security of our scheme, it is easy to see that our bootstrapping procedure can be secure by assuming the circular security and DLWE. Correctness holds as the following lemma.

Lemma 7. *Let* sk *be the secret key for our scheme. Let* $\boldsymbol{c}$ *and* $\boldsymbol{s}$ *be a ciphertext and secret key described in our bootstrapping procedure. Then, for* $\mathsf{bk}\overset{R}{\leftarrow}\mathsf{BootKeyGen}(\mathsf{sk},\boldsymbol{s})$*, the refreshed ciphertext* $\boldsymbol{C}^*\overset{R}{\leftarrow}\mathsf{Bootstrap}_{\mathsf{bk}}(\boldsymbol{c})$ *encrypts* $\lfloor\langle\boldsymbol{s},\boldsymbol{c}\rangle\rceil_2\in\{0,1\}$ *in the first slot.*

Proof. From Lemma 5 and group homomorphism of ϕ_i, $\boldsymbol{C}_i^*$ encrypts $\phi_i([\langle\boldsymbol{s},\boldsymbol{c}\rangle]_q)$. Since $\mathbb{Z}_q$ is isomorphic to $\mathbb{Z}_{r_1}\times\cdots\times\mathbb{Z}_{r_t}$ by CRT, $\bigodot_{i\in[t]}(\mathsf{SlotSwitch}_{\mathsf{ssk}_{\phi_i(x)}}(\boldsymbol{C}_i^*))\odot\boldsymbol{P}_{(1,0,\ldots,0)}$ encrypts 1 in the first slot if and only if $x=\langle\boldsymbol{s},\boldsymbol{c}\rangle \bmod q$. Finally, $\boldsymbol{C}^*$ encrypts 1 if and only if $\lfloor\langle\boldsymbol{s},\boldsymbol{c}\rangle\rceil_2=1$.

Here, we let s be the Gaussian parameter. Recall that n is the LWE dimension, r is the number of encrypted bits, $\ell=\lceil\log Q\rceil$, $N=(n+r)\cdot\ell$, $t=O(\log\lambda/\log\log\lambda)$, $d=\tilde{O}(\lambda)$ and $q=\tilde{O}(\lambda)$. We estimate the noise growth by our optimized bootstrapping procedure.

Lemma 8. *For any ciphertext* $\boldsymbol{c}\in\{0,1\}^d$ *described in our bootstrapping procedure, the noise in the refreshed ciphertext* $\boldsymbol{C}^*\overset{R}{\leftarrow}\mathsf{Bootstrap}_{\mathsf{bk}}(\boldsymbol{c})$ *has independent subgaussian entries with parameter* $O(s\sqrt{n\ell dtq})$*, except with probability* $2^{-\Omega((n+r)ldt)}$ *over the random choice of* bk *and* $\mathsf{Bootstrap}$*.*

Proof. Since the parenthesized part before the additions in Eq. (1) can be broken down into a sequence of $O(dt)$ homomorphic multiplications, Corollary 2 and Lemma 1 tell us that the term has subgaussian noise with parameter $O(s\sqrt{Ndt})$, except with probability $2^{-\Omega(Ndt)}$. From the Pythagorean additivity of subgaussian random variables and $N=(n+r)\cdot\ell$, the noise in $\boldsymbol{C}^*$ are subgaussian with parameter $O(s\sqrt{(n+r)\ell dtq})$, and so $O(s\sqrt{n\ell dtq})$ by the fact $n>r$.

From the above lemma, we can see that our procedure refreshes ciphertexts with error growth by the $O(\sqrt{nldtq})$ factor. Our scheme can evaluate its augmented decryption circuit by choosing a larger modulus than the final noise, and thus be pure FHE by the Gentry's bootstrapping theorem (Theorem 2) and the circular security assumption.

Theorem 3. *Our optimized bootstrapping scheme can be correct and secure assuming*

- *the quantum worst-case hardness of approximating* $\mathsf{GapSVP}_{\tilde{O}(n^{1.5}\lambda)}$ *and* $\mathsf{SIVP}_{\tilde{O}(n^{1.5}\lambda)}$,
- *or the classical worst-case hardness of approximating* $\mathsf{GapSVP}_{\tilde{O}(n^{2}\lambda)}$

on any n dimensional lattice.

Proof. By Lemma 1, to rely on the quantum worst-case hardness, we choose $s = \Theta(\sqrt{n})$. From Lemma 8, for correctness we only have to select $Q = \tilde{\Omega}(n\lambda \log Q)$, which satisfies $Q = \tilde{O}(n\lambda)$. Since the LWE inverse error rate is $1/\alpha = Q/s = \tilde{O}(\sqrt{n}\lambda)$, the security of our bootstrapping scheme is reduced to $\mathsf{GapSVP}_{\tilde{O}(n^{1.5}\lambda)}$ and $\mathsf{SIVP}_{\tilde{O}(n^{1.5}\lambda)}$.

In the case of reducing to the classical hardness of the lattice problem, since $1/\alpha = \tilde{\Omega}(\lambda\sqrt{n \log Q})$ and we must take $Q \approx 2^{n/2}$, the LWE inverse error rate satisfies $1/\alpha = \tilde{\Omega}(\lambda \cdot n)$. Therefore, the security of our optimized bootstrapping scheme is reduced to the classical hardness of $\mathsf{GapSVP}_{\tilde{O}(n^{2}\lambda)}$.

Since all known algorithms that approximate GapSVP and SIVP on any n dimensional lattices to within a poly(n)-factor run in time $2^{\Omega(n)}$, the 2^{λ} hardness requires us to choose $n = \Theta(\lambda)$. This makes the problems to which the security is reduced in the quantum case have the approximation factor $\tilde{O}(n^{2.5})$, which is smaller than $\tilde{O}(n^{3})$, the one of [1]'s bootstrapping scheme. In the classical case, the LWE inverse error rate is $1/\alpha = \tilde{\Omega}(n^{2})$ and hence our approximation factor is $\tilde{O}(n^{3})$. Furthermore, by selecting a larger dimension $n = \lambda^{1/\epsilon}$ for $\epsilon > 0$ (so at the cost of efficiency), the approximation factor can be $\tilde{O}(n^{1.5+\epsilon})$, which is comparable to the one of [9] and so the best known factor of standard lattice-based PKE. Consequently, our optimized bootstrapping scheme can be as secure as any other standard lattice-based PKE *without* successive dimension-modulus reduction, which is essential in all the known bootstrapping procedures [1, 9] provided recently.

References

1. Jacob Alperin-Sheriff and Chris Peikert. Faster Bootstrapping with Polynomial Error. In *CRYPTO*, pages 297-314, 2014.
2. David A. Mix Barrington. Bounded-Width Polynomial-Size Branching Programs Recognize Exactly Those Launguages in NC^1. In *STOC*, pages 1-5, 1986.
3. Boaz Barak. Cryptography course - Lecture Notes, COS 433. Princeton University, Computer Science Department, 2010. Available at `http://www.cs.princeton.edu/courses/archive/spring10/cos433`.

4. Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed Ciphertexts in LWE-based Homomorphic Encryption. In *PKC*, pages 1-13, 2013.
5. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) Fully Homomorphic Encryption without Bootstrapping. In *ITCS*, pages 309-325, 2012.
6. Zvika Brakerski. Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. In *CRYPTO*, pages 868-886, 2012.
7. Zvika Brakerski and Vinod Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. In *FOCS*, pages 97-106, 2011.
8. Zvika Brakerski and Vinod Vaikuntanathan. Fully Homomorphic Encryption from Ring-LWE and Security for Key Depedent Messages. In *CRYPTO*, pages 505-524, 2011.
9. Zvika Brakerski and Vinod Vaikuntanathan. Lattice-Based FHE as Secure as PKE. In *ITCS*, pages 1-12, 2014.
10. Jean-Sébastian Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical Multilinear Maps over the Integers. In *CRYPTO*, pages 476-493, 2013.
11. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully Homomorphic Encryption over the Integers. In *EUROCRYPT*, pages 24-43, 2010.
12. Craig Gentry. A FULLY HOMOMORPHIC ENCRYPTION SCHEME. PhD thesis, Stanford University, 2009. Available at `http://crypto.stanford.edu/craig`.
13. Craig Gentry. Fully Homomorphic Encryption using Ideal Lattices. In STOC, pages 169-178, 2009.
14. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate Multilinear Maps from Ideal Lattices. In *EUROCRYPT*, pages 1-17, 2013.
15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-Induced Multilinear Maps from Lattices. *IACR Cryptology ePrint Archive*, 2014:645, 2014.
16. Craig Gentry, Shai Halevi, and Nigel P. Smart. Better Bootstrapping in Fully Homomorphic Encryption. In *PKC*, pages 1-16, 2012.
17. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In *CRYPTO*, pages 75-92, 2013.
18. Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-Fly Multiparty Computation on the Cloud via Multikey Fully Homomorphic Encryption. In *STOC*, pages 1219-1234, 2012.
19. Daniele Micciancio and Petros Mol. Pseudorandom Knapsacks and the Sample Complexity of LWE Search-to-Decision Reductions. In *CRYPTO*, pages 465-484, 2011.
20. Daniele Micciancio and Chris Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *EUROCRYPT*, pages 700-718, 2012.
21. Chris Peikert. Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem. In *STOC*, pages 333-342, 2009.
22. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A Framework for Efficient and Composable Oblivious Transfer. In *CRYPTO*, pages 554-571, 2008.
23. Oded Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In *STOC*, pages 84-93, 2005.
24. Ron Rothblum. Homomorphic Encryption: from Private-Key to Public-Key. In *TCC*, pages 219-234, 2011.
25. Nigel P. Smart and Frederik Vercauteren. Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. In *PKC*, pages 420-443, 2010.

26. Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In Yonina C. Eldar and Gitta Kutyniok, editors, Compressed Sensing, Theory and Applications, chapter 5, pages 210-268. Cambridge University Press, `http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`, 2012.