Efficiently Processing Complex-Valued Data in Homomorphic Encryption

Carl Bootland[1], Wouter Castryck[1,2], Ilia Iliashenko[1], and Frederik Vercauteren[1]

[1]imec-COSIC, Dept. Electrical Engineering, KU Leuven
firstname.lastname@esat.kuleuven.be
[2]Department of Mathematics, KU Leuven

Abstract. We introduce a new homomorphic encryption scheme that is natively capable of computing with complex numbers. This is done by generalizing recent work of Chen, Laine, Player and Xia, who modified the Fan–Vercauteren scheme by replacing the integral plaintext modulus t by a linear polynomial $X - b$. Our generalization studies plaintext moduli of the form $X^m + b$. Our construction significantly reduces the noise growth in comparison to the original FV scheme, so much deeper arithmetic circuits can be homomorphically executed.

1 Introduction

The goal of homomorphic encryption is to allow for arbitrary arithmetic operations on encrypted data, such that the decrypted result equals the outcome of the same calculation carried out in the clear. Since the publication of Gentry's seminal Ph.D. work [15], this research area has evolved rapidly and is on the verge of reaching a first degree of maturity, as was recently demonstrated e.g. by practical implementations of privacy-enhanced electricity load forecasting [3, 2], digital image processing [1, 10], and medical data management [12, 18, 7]. Most of the current focus lies on *somewhat* homomorphic encryption (SHE), where the schemes are capable of homomorphically evaluating an arithmetic circuit having a certain predetermined computational depth. The leading proposals for realizing this goal are the Brakerski-Gentry-Vaikunthanathan (BGV) scheme [4] and the Fan-Vercauteren (FV) scheme [13].

In actual applications, the input to the homomorphic evaluation of an arithmetic circuit $\mathcal{C}$ needs to be preprocessed in two steps. The first step is encoding, where one's task is to represent the actual 'real world data' as elements of the plaintext space of the envisaged SHE scheme. This plaintext space is a certain commutative ring, and the encoding should be such that real world arithmetic

The first author is supported by a PhD fellowship of the Research Foundation - Flanders (FWO). The third author has been supported in part by ERC Advanced Grant ERC-2015-AdG-IMPaCT.

agrees with the corresponding ring operations, up to the anticipated computational depth.

In the original descriptions of BGV and FV, the plaintext space is a ring of the form $R_t = \mathbb{Z}[X]/(t, f(X))$ where $t \geq 2$ is an integer and $f(X) \in \mathbb{Z}[X]$ is a monic irreducible polynomial. Throughout this paper we will stick to the common choice of 2-power cyclotomics $f(X) = X^n + 1$, where $n = 2^k$ for some integer $k \geq 1$. Encoding numerical input is typically done by taking an integer-digit expansion with respect to some base b, then replacing b by X and finally reducing the digits modulo t. Decoding then amounts to lifting the coefficients back to $\mathbb{Z}$, for instance by choosing representatives in $(-t/2, t/2]$, and evaluating the result at $X = b$. Thanks to the relation $X^{-1} \equiv -X^{n-1}$ it is possible to allow the expansions to have a fractional part. In this case the decoding step must be preceded by replacing the monomials X^i of degree $i > B$ by $-X^{i-n}$, for some appropriate point of separation B. All these parameters need to be chosen in such a way that the evaluation of $\mathcal{C}$ on the encoded data decodes to the right outcome. At the same time one wants t to be as small as possible, because its size highly affects the efficiency of the resulting SHE computation. Selecting optimal parameters is a tedious application-dependent balancing act to which a large amount of recent literature has been devoted, see e.g. [20, 12, 8, 6, 18, 11, 2].

Because in practice n is of size at least 1024, the plaintext spaces R_t can a priori host an enormous range of data, even for very small values of t. Unfortunately this is hindered by their structure, which is not a great match with numerical input data types like integers, rationals or floats. For example, if $t = 2$ then it is not even possible to add a non-zero element to itself without incorrect decoding. Because of such phenomena, values of t are required that typically consist of dozens of decimal digits, badly affecting the efficiency. An idea to remedy this situation has been around for a while [17, 4, 14] and uses a polynomial plaintext modulus, rather than just an integer. Recently the first detailed instantiation of this idea was given by Chen, Laine, Player and Xia [6], who adapted the FV scheme to plaintext moduli $t = X - b$ for some $b \in \mathbb{Z}_{\geq 2}$. In this case the plaintext space becomes $R_t = \mathbb{Z}[X]/(X - b, X^n + 1) = \mathbb{Z}[X]/(X - b, b^n + 1) \cong \mathbb{Z}_{b^n+1}$, whose structure is a *much* better match with the common numerical input data types. This allows for much smaller plaintext moduli (norm-wise), with beneficial consequences for the efficiency, or for the depth of the circuits $\mathcal{C}$ that can be handled [6, Section 7.2].

This paper further explores the paradigm that the structure of the plaintext space R_t should match the input data type as closely as possible. Concretely, we focus on *complex-valued* data types, such as cyclotomic integers and floating point complex numbers. We study this setting mainly in its own right, but note that complex input data has been considered in homomorphic encryption before, e.g., in the homomorphic evaluation of the Discrete Fourier Transform studied by Costache, Smart and Vivek [10] in the context of digital image processing, where the input consists of cyclotomic integers.

Representing complex numbers. One naive way to encode a complex number z would be to view it as a pair of real numbers, for instance using Cartesian or polar coordinates. These can be fed separately to the SHE scheme, which is now used to evaluate two circuits. A more direct way is to use a complex base b. For instance, one could take $b = e^{\pi \mathbf{i}/n}$, as was done by Cheon, Kim, Kim and Song [8], albeit in a somewhat different context. This choice has the additional feature that $f(b) = 0$, so that wrapping around modulo $f(X) = X^n + 1$ does not lead to incorrect decoding. However, finding an integer-digit base b expansion with small norm which approximates z sufficiently well is an n-dimensional lattice problem, which is practically infeasible. To get around this Costache, Smart and Vivek [10] instead use $b = \zeta := e^{\pi \mathbf{i}/m}$ for some divisor $m \mid n$, which is small enough for finding short base ζ approximations, while preserving the feature that wrapping around modulo $f(X)$ is unharmful. But in their approach, a huge portion of plaintext space is left *unused*. Indeed, the encoding map is

$$\mathbb{Z}[\zeta] \to R_t : z = \sum_{i=0}^{m-1} z_i b^i \mapsto \sum_{i=0}^{m-1} \overline{z}_i Y^i,$$

where $Y = X^{n/m}$, $t \geq 2$ is an integral plaintext modulus and $\overline{z}_i$ is the reduction of z_i mod t, so that all plaintext computations are carried out in the subring $\mathbb{Z}[Y]/(t, Y^m + 1)$, which is of index t^{n-m} in R_t. Our proposal is to resort to a plaintext modulus of the form $t = X^m + b$ for some small integer b, with $|b| \geq 2$. In this case, for $m < n$, we have $R_{X^m+b} = \mathbb{Z}[X]/(X^m + b, X^n + 1) = \mathbb{Z}[X]/(b^{n/m} + 1, X^m + b)$. An additional assumption (which is discussed in more detail in the next section), is that

$$\text{there exists an } \overline{\alpha} \in \mathbb{Z}_{b^{n/m}+1} \text{ such that } \overline{b} = \overline{\alpha}^m, \tag{1}$$

where $\overline{b}$ denotes the reduction of b modulo $b^{n/m} + 1$. Throughout we fix such an $\overline{\alpha}$ and let $\overline{\beta}$ be its multiplicative inverse, which necessarily exists. This implies that $(\bar{\beta} X)^m + 1 = 0$, therefore we have a well-defined ring homomorphism

$$\mathbb{Z}[\zeta] \to R_{X^m+b} : \sum_{i=0}^{m-1} z_i \zeta^i \mapsto \sum_{i=0}^{m-1} \overline{z}_i \overline{\beta}^i X^i \tag{2}$$

which is surjective with kernel $(b^{n/m} + 1)$. In other words, while Costache, Smart and Vivek restrict their computations to an injective copy of $\mathbb{Z}[\zeta]/(t)$ inside R_t, we can view R_{X^m+b} as an *isomorphic* copy of $\mathbb{Z}[\zeta]/(b^{n/m} + 1)$. Essentially, our approach transfers the unused part of the plaintext space coming from the large dimension n into a larger integral modulus, reflected in the exponent n/m.

In the remainder of this paper, we explain how this observation can be used to efficiently process complex-valued input data in homomorphic encryption. First, in Section 2 we explain how to encode and decode elements of the ring $\mathbb{Z}[\zeta]$ of $2m^{\text{th}}$ cyclotomic integers and discuss the assumption (1), with special attention to the case $m = 2$ where $\mathbb{Z}[\zeta] = \mathbb{Z}[\mathbf{i}]$ is the ring of Gaussian integers. Next in Section 3 we explain how this can be used to encode other data types such as cyclotomic rationals or complex floats, either by resorting to LLL as in [10] or by using Chen et al.'s fractional encoder from [6]. In Section 4 we discuss how to adapt the FV scheme so that it can cope with plaintext spaces of the form R_{X^m+b}. Finally, in Section 6 we discuss the performance of this adaptation in comparison with previous approaches. In short we can reach a depth at least 5 times that of the best approach which directly encrypts encodings of complex numbers [10]. We can also reach very similar depths to the state of the art where one encrypts the real and imaginary parts separately [6]. However, since we natively encrypt complex numbers our ciphertexts are two times smaller and hence our approach is more efficient by roughly a factor two in time and three in space.

2 Encoding and decoding elements of $\mathbb{Z}[\zeta]$

Encoding Encoding an element of $\mathbb{Z}[\zeta]$ happens in two steps. The first step applies the map (2) yielding a polynomial of degree less than m which typically has very large coefficients. The second step is comparable to the *hat encoder* of Chen et al. [6] and switches to another representant by spreading this polynomial across the range $1, X, \ldots, X^{n-1}$ while making the coefficients a lot smaller. The result will then be lifted to $R = \mathbb{Z}[X]/(X^n + 1)$ and fed to our adaptation of the FV scheme, where the smaller coefficients are important to keep the noise growth bounded.

Here is how this second step is carried out in practice: we think of the coefficients $\overline{z}_i\overline{\beta}^i$ as being represented by integers between $-\lfloor b^{n/m}/2\rfloor$ and $\lceil b^{n/m}/2\rceil$. We then expand these integers to base b using digits $a_{i,j}$ from the range $-\lfloor b/2\rfloor$, $\ldots$, $\lfloor b/2\rfloor$ to find

$$\overline{z}_i\overline{\beta}^i = \overline{a}_{i,n/m-1}\overline{b}^{n/m-1} + \ldots + \overline{a}_{i,1}\overline{b} + \overline{a}_{i,0}.$$

There is a minor caveat here, namely if b is odd then there are more integers modulo $b^{n/m} + 1$ than there are balanced b-ary expansions of length at most n/m. This is easily resolved by allowing the last digit to be one larger. For even b the situation is opposite: since $\overline{z}_i\overline{\beta}^i$ is represented by an integer of size at most $b^{n/m}/2 = b/2 \cdot b^{n/m-1}$ we have a surplus of base-b expansions. Here it makes sense to choose an expansion with the shortest Hamming weight (e.g., if $b = 2$ then we simply pick the non-adjacent form). We denote the maximal number of non-zero coefficients that can appear in a fresh encoding by N_b.

Given such base-b expansions of the coefficients, we replace each occurrence of $\overline{b}$ by $-X^m$ and then substitute the results in the image of (2). We end up with an expansion $\sum_{i=0}^{n-1} \overline{c}_i X^i$ where the $\overline{c}_i$ are represented by integers of absolute value at most $\lfloor b/2\rfloor$, or in fact $\lfloor (b+1)/2\rfloor$ if we take into account the caveat.

Decoding In order to decode a given expansion $\sum_{i=0}^{n-1} \overline{c}_i X^i$ we walk through the same steps in reverse order. First we pick another representant by reducing the expansion modulo $X^m + b$, in order to end up with

$$\sum_{i=0}^{m-1} \overline{c}_i' X^i \in \mathbb{Z}[X]/(b^{n/m} + 1, X^m + b).$$

This can be rewritten as $\sum_{i=0}^{m-1} \overline{c}_i' \overline{\alpha}^i \overline{\beta}^i X^i$ so we decode as $\sum_{i=0}^{m-1} z_i \zeta^i \in \mathbb{Z}[\zeta]$ where z_i is a representant of $\overline{c}_i' \overline{\alpha}^i$ taken from the range $-\lfloor b^{n/m}/2 \rfloor, \ldots, \lceil b^{n/m}/2 \rceil$.

On the assumption (1) Usually n and m are determined by security considerations and the concrete application. To apply our encoding method we want to find a small value of b for which condition (1) is met. This is easiest if n/m is small or m is small. If no satisfactory value of b can be found then one can try to enlarge m and view $\mathbb{Z}[\zeta]$ as a subring of a higher degree cyclotomic ring. Below we give two lemmas constraining the possible choices for b given m and n; still assuming we are working with 2-power cyclotomic f.

One choice for b which is always possible is $2^{m/2}$, since defining α as

$$\alpha = 2^{n/8} \left(2^{n/4} - 1 \right), \tag{3}$$

then it easy to verify that $\alpha^2 \equiv 2 \bmod 2^{n/2} + 1$ and hence

$$\alpha^m \equiv 2^{m/2} \bmod 2^{\frac{m}{2}\frac{n}{m}} + 1.$$

If m is small then this results in a reasonably slow coefficient growth. On the other hand if m is large compared to n then the modulus $b^{n/m} + 1$ is smaller and it is apparently easier to have condition (1) satisfied, as is confirmed by experiment.

Lemma 1. *Let $n > m > 1$. A necessary condition for* (1) *is that for every odd prime $p \mid b^{n/m} + 1$ we have $2n \mid p - 1$.*

Proof. First we show that b has multiplicative order $2n/m$ in $\mathbb{Z}_{b^{n/m}+1}$. Clearly we have $b^{n/m} \equiv -1 \mod b^{n/m} + 1$ so that $b^{2n/m} \equiv 1 \mod b^{n/m} + 1$. This shows that the order of b divides $2n/m$ so is a power of 2 and hence it is equal to $2n/m$.

Since $2 \mid n/m$ and $x^2 \equiv 1 \mod 4$ for any odd x we have that if b is odd $b^{n/m} + 1 \equiv 2 \mod 4$ while if b is even $b^{n/m} + 1$ is odd. Thus that we can write

$$b^{n/m} + 1 = 2^\rho p_1^{e_1} \ldots p_j^{e_j}$$

where the p_i, $1 \leq i \leq j$ are distinct odd primes and $\rho = b \bmod 2$.

Now we can see via the Chinese Remainder Theorem that there exists an α such that $\alpha^m \equiv b \mod b^{n/m} + 1$ if and only if there exist α_i such that $\alpha_i^m \equiv b \bmod p_i^{e_i}$ for every i. Further we must have $b^{n/m} \equiv -1 \mod p_i^{e_i}$ so that b has order $2n/m$ modulo $p_i^{e_i}$. This implies α_i has order $m \cdot 2n/m = 2n$ modulo $p_i^{e_i}$ and since $(\mathbb{Z}/p_i^{e_i}\mathbb{Z})^\times$ is cyclic of order $p_i^{e_i - 1}(p_i - 1)$ we see that $2n \mid (p_i - 1)$ by Lagrange's Theorem for each $1 \leq i \leq j$.

Lemma 2. *Let g be an element of order n in $\mathbb{Z}_{4n}^{\times}$ and let t be an element of order 2 not in $\langle g \rangle$ so that $\mathbb{Z}_{4n}^{\times} = \langle t \rangle \times \langle g \rangle$. If condition* (1) *is satisfied for odd $b > 1$ and $m > 1$ then $b \bmod 4n$ is an element of the subgroup $\langle t \rangle \times \langle g^m \rangle$. In particular this implies that $b \equiv \pm 1 \bmod 4m$.*

In fact, one may always take $g = 3$ and $t = -1$ in the above lemma.

Proof. Using Lemma 1 and the notation from its proof we can write each p_i as $2nc_i + 1$ for some natural number c_i. This implies that

$$b^{n/m} + 1 = 2 \prod_{i=1}^{j} (2nc_i + 1)^{e_i} \equiv 2 \bmod 4n$$

and hence $b^{n/m} \equiv 1 \bmod 4n$. Therefore the order of b as an element of $\mathbb{Z}_{4n}^{\times}$ divides n/m.

Now we have $\mathbb{Z}_{4n}^{\times} = \langle t \rangle \times \langle g \rangle$ so that for $b \bmod 4n$ to have an order dividing n/m it must be an element of the subgroup $\langle t \rangle \times \langle g^m \rangle$. This is because this subgroup certainly only contains elements whose order divides n/m. Further, $\mathbb{Z}_{4n}^{\times}$ has exactly $2n/m$ such elements but this is the size of the subgroup so the subgroup is exactly all such elements.

For the final part we note, as stated after the lemma, that $g = 3$ and $t = -1$ can be taken and that $3^m \equiv 1 \bmod 4m$ which gives the desired result. We remark that for any $b \equiv \pm 1 \bmod 4m$ it is always the case that $b^{n/m} \equiv 1 \bmod 4n$ so from this condition we cannot determine anything more about b modulo $4m$ but the condition given modulo $4n$ is stronger.

Lemma 3. *Suppose b, n and m satisfy* (1)*, then so does $-b, n, m$.*

Proof. Since $(-b)^{n/m} + 1 = b^{n/m} + 1$ when n is a power of two and $m < n$, we must show that -1 has an mth root modulo $b^{n/m} + 1$; we show that $\alpha^{n/m}$ is such an mth root. We have $(\alpha^{n/m})^m = (\alpha^m)^{n/m} \equiv b^{n/m} \equiv -1 \bmod b^{n/m} + 1$ as required. Hence we see that $(\alpha^{n/m+1})^m = \alpha^n \alpha^m = -1 \cdot b$ as required.

We note that the above proof only required n/m to be even and not equal to a power of two so applies somewhat more generally.

We give some examples of both odd and even b which satisfy Equation (1) in Appendix A. However it seems to be more fruitful to consider the case of even b.

Our method is particularly friendly towards Gaussian integers. Indeed if $m = 2$ then one can always take $b = 2$, as we have seen that $\overline{\alpha}^2 = \overline{2}$ where α is as in (3). The map (2) then defines an isomorphism between R_{X^2+2} and $\mathbb{Z}[\mathbf{i}]/(2^{n/2} + 1)$. If this ring is not large enough to ensure correct decoding, then one can move to slightly larger values of b. The next choice which always works is $b = 4$, where one can simply take $\alpha = 2$. Here the ring becomes $\mathbb{Z}[\mathbf{i}]/(2^n + 1)$.

3 Encoding complex-valued input data

In this section we look at the more general problem of encoding floating point complex numbers. Our approach will be to approximate these complex numbers by suitable cyclotomic rationals and then proceed as in Section 2. We have many choices for such approximations including the choice of m which defines which root of unity we are working with. We also have the choice between using integer or rational coefficients for the approximation. Perhaps the most obvious and straightforward approach is to consider our complex number z written in terms of its real and imaginary parts, say $z = x + y\mathbf{i}$ for some real numbers x and y. We can then approximate x and y by rationals depending on how much precision we require. This leads us to considering the case $m = 2$ and the question then arises of how to encode fractional coefficients.

3.1 Fractional encoding

Here we consider how to encode a rational number into the space $\mathbb{Z}/p\mathbb{Z}$ for some integer p, so that it can then be expanded using the technique in Section 2. This problem was considered by Chen, Laine, Player and Xia in [6, Section 6]. Their approach is to define a finite subset $\mathcal{P}$ of $\mathbb{Q}$ along with an encoding map $\texttt{Enc}\colon \mathcal{P} \to \mathbb{Z}/p\mathbb{Z}$ and a decoding map $\texttt{Dec}\colon \texttt{Enc}(\mathcal{P}) \to \mathcal{P}$. The maps should satisfy, firstly, correctness: $\texttt{Dec}(\texttt{Enc}(x/y)) = x/y$ for $x/y \in \mathcal{P}$ and secondly, $\texttt{Enc}$ should be both additively and multiplicatively homomorphic so long as it still encodes an element of $\mathcal{P}$. The natural choice for the map $\texttt{Enc}$ is $\texttt{Enc}(x/y) = xy^{-1}$ mod p where the inverse of y is computed modulo p. Care thus needs to be taken to ensure that y has such an inverse, which is ensured with a careful choice of $\mathcal{P}$.

In our setting the coefficient modulus p is of the form $b^{n/2} + 1$, thus if one wants roughly the same precision for the integer and fractional parts one can take for an odd base b

$$\mathcal{P} = \left\{ c + \frac{d}{b^{n/4}} \ :\ c, d \in \left[-\frac{b^{n/4}-1}{2}, \frac{b^{n/4}-1}{2} \right] \cap \mathbb{Z} \right\};$$

while for even b one can choose

$$\mathcal{P} = \left\{ c + \frac{d}{b^{n/4-\delta}} : |c| \leq \frac{(b^{n/4+\delta-1}-1)b}{2(b-1)}; |d| \leq \frac{(b^{n/4-\delta}-1)b}{2(b-1)}; c, d \in \mathbb{Z} \right\},$$

where $\delta \in \{0, 1\}$ depending on whether you want one more base-b digit in the fractional ($\delta = 0$) or integer ($\delta = 1$) part.

The encoding of an element $e \in \mathcal{P}$ is then computed as $-eb^{n/2}$ mod $b^{n/2} + 1$. The important thing to note about using this encoding is that for decoding to work the result of the computations must lie in $\mathcal{P}$. If your input data are complex numbers and you approximate them using $n/4$ fractional b-ary digits then it is likely that after one multiplication the result is no longer in $\mathcal{P}$. Thus one must appropriately choose the precision with which to encode the data, depending primarily on the depth of the circuit to be evaluated and the final precision required. The only constraint is that the precision should be a divisor of $b^{n/4}$ so that $-eb^{n/2}$ is an integer.

We note that the fractional encoder need not require m to be 2. However in this case there appears to be no straightforward way to find a good rational approximation with small numerators and denominators except when the denominators are all equal, in this case if this denominator is r then we simply require an approximation of rz in $\mathbb{Z}[\zeta]$ subject to some constraint on the coefficients. However, the problem of finding such an approximation to our complex number itself, rather than a scaling, is interesting in its own right as it avoids the need for encoding fractional values and tracking the denominator inherently present in such encodings.

3.2 Integer coefficient approximation

The task of finding a cyclotomic integer closely approximating an arbitrary complex number was considered by Costache, Smart and Vivek in [10]. Here the idea is to solve an instance of the closest vector problem (CVP) in the (scaled) lattice $\mathbb{Z}[\zeta]$, where the power basis is scaled and split into real and complex part, which are approximated by integers. In detail: we choose a scaling constant $C > 0$, and define the constants a_i and b_i for $i = 0, \dots, m-1$, where $a_i = \lceil \Re(C\zeta^i) \rfloor$ and $b_i = \lceil \Im(C\zeta^i) \rfloor$. The lattice we then consider is given by the m rows of the matrix

$$\begin{pmatrix} 1 & & 0 & a_0 & b_0 \\ & \ddots & & \vdots & \vdots \\ 0 & & 1 & a_{m-1} & b_{m-1} \end{pmatrix}.$$

The target vector in our CVP instance will then be the appropriately scaled real and complex parts of the complex number z we wish to approximate. Concretely, this vector is $(0, \dots, 0, \lceil \Re(Cz) \rfloor, \lceil \Im(Cz) \rfloor)$.

If $(z_0, \dots, z_{m-1}, A, B)$ is a solution to the CVP instance then we must have

$$\lceil \Re(Cz) \rfloor \approx A = \sum_{i=0}^{m-1} z_i a_i \approx \Re\left(C \sum_{i=0}^{m-1} z_i \zeta^i \right)$$

and similarly for the imaginary part. We therefore see that $\sum_{i=0}^{m-1} z_i \zeta^i$ is a good approximation to z. Further, C gives some control over the quality of the approximation, larger C gives a finer-grained lattice but also increases the size of the last two coefficients of the basis vectors which may lead to a larger distance between the target vector and the closest lattice point, which in turn makes solving the CVP instance harder and negatively affects the quality of our approximation of Cz.

In [10] the authors solve this CVP instance using the embedding technique. Namely they attempt to solve the shortest vector problem in the lattice spanned by the rows of

$$\begin{pmatrix} 1 & & 0 & a_0 & b_0 & 0 \\ & \ddots & & \vdots & \vdots & \vdots \\ 0 & & 1 & a_{m-1} & b_{m-1} & 0 \\ 0 & \cdots & 0 & \lceil \Re(Cz) \rfloor & \lceil \Im(Cz) \rfloor & T \end{pmatrix}$$

for some non-zero constant T. With suitable parameter choices, performing LLL reduction on this lattice will return a basis of short vectors for this lattice, among which at least one has $\pm T$ in the final coordinate. The remaining coefficients then give plus or minus the target vector minus a close vector.

One issue with the embedding technique is that each new instance of the CVP problem requires performing lattice reduction which for large m is rather time-consuming. In typical applications we want to approximate many different complex numbers, using the same C so only the target vector changes. A more efficient approach therefore is to perform lattice reduction on the CVP lattice itself and since this is independent of the target vector it needs only to be done once so we can spend significantly more time in this step to find a good basis of this lattice. We can then apply a technique such as Babai's nearest plane algorithm, or Babai's rounding algorithm, with this reduced basis to find an approximate closest vector.

4 Adapting the Fan-Vercauteren SHE scheme

In this section we construct a variant of the FV scheme [13] with plaintext modulus $X^m + b$ following the blueprint given in [6]. We prove correctness of this scheme (Theorem 1) and analyze the noise growth induced by homomorphic arithmetic operations (Lemma 6, Theorem 2).

4.1 Basic scheme

Writing $R = \mathbb{Z}[X]/(X^n + 1)$, the ciphertext space is defined by $R_q = R/(q)$ for some positive integer q, while the plaintext space is $R_{X^m+b} = R/(X^m + b)$. We will assume that $b \ll q$. Recall that in the original FV scheme the plaintext space is $R/(t)$ for some positive integer $t \ll q$. We define the scaling parameter Δ_b as

$$\Delta_b = \left\lfloor \frac{q}{X^m + b} \mod (X^n + 1) \right\rceil = \left\lfloor -\frac{q}{b^{n/m} + 1} \sum_{i=1}^{n/m} (-b)^{i-1} X^{n-im} \right\rceil .$$

Obviously, Δ_b is the analogue of the scalar $\Delta = \lfloor q/t \rfloor$ in the original FV scheme. Other parameters are the error distribution $\chi_e = \mathcal{D}(\sigma^2)$ on R (coefficient-wise with respect to the power basis, with standard deviation σ) and the key distribution $\chi_k = \mathcal{U}_3$ which uniformly generates elements of R with ternary coefficients (with respect to the power basis). We also define the decomposition base w and denote $\ell = \lfloor \log_w q \rfloor$.

The new encryption scheme `ComFV` is then defined in the same way as `FV` where t and Δ are replaced by $X^m + b$ and Δ_b, respectively.

- `ComFV.KeyGen( )`: Let $s \leftarrow \chi_k$ and $e, e_0, \dots e_\ell \leftarrow \chi_e$. Uniformly sample random $a, a_0, \dots, a_\ell \in R_q$ and compute $b_i = \left[-(a_i s + e_i) + w^i s^2\right]_q$. Output the secret key $\mathtt{sk} = s$, the public key $\mathtt{pk} = \Big([-(as+e)]_q \,, a \Big)$ and the evaluation key $\mathtt{evk} = \{(b_i, a_i)\}_{i=0}^{\ell}$.
- $\mathtt{ComFV.Encrypt}(\mathtt{pk}, \mathsf{msg})$: Sample $u \leftarrow \chi_k$ and $e_0, e_1 \leftarrow \chi_e$. Set $p_0 = \mathtt{pk}[0]$ and $p_1 = \mathtt{pk}[1]$, and compute $c_0 = [\Delta_b \cdot \mathsf{msg} + p_0 u + e_0]_q$ and $c_1 = [p_1 u + e_1]_q$. Output $\mathtt{ct} = (c_0, c_1)$.
- $\mathtt{ComFV.Decrypt}(\mathtt{sk}, \mathtt{ct})$: Return $\mathsf{msg}' = \left\lfloor \frac{X^m+b}{q} [c_0 + c_1 s]_q \right\rceil \mod (X^m + b)$.

The security of this scheme is based on the same argument as of the original FV scheme. In particular, it is hard to distinguish the public key `pk` and ciphertext pairs from uniform tuples according to the decision version of the Ring-LWE problem [19]. The evaluation key `evk` does not leak any information about the secret key as long as a circular security assumption holds [13].

For an element $a \in K := \mathbb{Q}[x]/(f(x))$ the canonical (infinity) norm of a is defined as

$$\|a\|_\infty^{\mathrm{can}} = \left\| \left(a(\zeta), a(\zeta^3), \dots, a(\zeta^{2n-1}) \right) \right\|_\infty .$$

In Appendix A we state some properties of the canonical norm which will be used throughout this section. To verify correctness we use the notion of invariant noise introduced in [6]. The *invariant noise* of a ciphertext $\mathtt{ct} = (c_0, c_1)$ encrypting a plaintext $\mathsf{msg} \in R_{X^m+b}$ is an element $v \in K$ with the smallest canonical norm such that

$$\frac{X^m + b}{q} \cdot [c_0 + c_1 s]_q = \mathsf{msg} + v + g(X^m + b) \tag{4}$$

for some $g \in R$. Then decryption works correctly when $\|v\|_\infty^{\mathrm{can}} < 1/2$ that is supported by the following lemma.

Lemma 4 (Decryption noise). *Let* `ct` *be an encryption of the plaintext element* $\mathsf{msg} \in R_{X^m+b}$ *such that its invariant noise* v *satisfies* $\|v\|_\infty^{\mathrm{can}} < 1/2$. *Then* $\mathtt{ComFV.Decrypt}(\mathtt{sk}, \mathtt{ct}) = \mathsf{msg}$.

Proof. Computing $\mathtt{ComFV.Decrypt}(\mathtt{sk}, \mathtt{ct})$, we have using the definition of the invariant noise

$$\begin{aligned} \mathsf{msg}' &= \left\lfloor \frac{X^m + b}{q} \left[\mathtt{ct}[0] + \mathtt{ct}[1] \cdot s\right]_q \right\rceil \mod (X^m + b) \\ &= \lfloor \mathsf{msg} + v + g(X^m + b) \rceil \mod (X^m + b) \\ &= \mathsf{msg} + \lfloor v \rceil \end{aligned}$$

for some $g \in R$ and since $\|v\|_\infty \le \|v\|_\infty^{\mathrm{can}} < 1/2$ we have $\lfloor v \rceil = 0$. Thus $\mathsf{msg}' = \mathsf{msg}$.

To show that v is small enough, we need an upper bound on the initial invariant noise size depending on the scheme parameters. To proceed we need the next lemma.

Lemma 5 (Scaling noise). *With Δ_b defined as above,*

$$\frac{\Delta_b(X^m+b)}{q} = 1 + \frac{\rho}{q} \in K,$$

for some $\rho \in K$ satisfying $\|\rho\|_\infty^{\mathrm{can}} \leq (b+1)\sqrt{3n}$ with very high probability.

Proof. For some polynomial $g \in K$ with $\|g\|_\infty \leq 1/2$,

$$\frac{\Delta_b(X^m+b)}{q} = \left(\frac{q}{X^m+b} + g\right) \cdot \frac{X^m+b}{q} = 1 + \frac{g(X^m+b)}{q}.$$

Thus we can take $\rho = g(X^m+b) \in K$ and

$$\|\rho\|_\infty^{\mathrm{can}} = \|g(X^m+b)\|_\infty^{\mathrm{can}} \leq (b+1)\sqrt{3n},$$

where the last inequality is due to $g(X) \leftarrow \mathcal{U}_{\mathrm{rnd}}$; see Appendix A.

Recall that the Hamming weight of a plaintext $\mathsf{msg} \in R_{X^m+b}$ is bounded by N_b. In addition, $\|\mathsf{msg}\|_\infty \leq b/2$ for even b and $\|\mathsf{msg}\|_\infty \leq (b+1)/2$ for odd b with at most one coefficient reaching this bound. Hence, $\|\mathsf{msg}\|_\infty^{\mathrm{can}} \leq N_b(b+1)/2$. Now, we have all the ingredients to define the scheme parameters supporting correct decryption.

Theorem 1 (Fresh noise). *Let $\mathtt{ct} = \mathtt{ComFV.Encrypt}(\mathtt{pk}, \mathtt{msg})$ be a fresh ciphertext, then the invariant noise of $\mathtt{ct}$ is bounded with very high probability by*

$$\frac{b+1}{q}\left(\frac{\sqrt{3n}}{2}(b+1)N_b + 2\sigma n\sqrt{12+\frac{9}{n}}\right),$$

where N_b is the number of non-zero coefficients that can appear in a fresh encoding and σ is the standard deviation of the error distribution χ_e.

Proof. Set $c_0 = \mathtt{ct}[0]$, $c_1 = \mathtt{ct}[1]$, and $p_0 = \mathtt{pk}[0], p_1 = \mathtt{pk}[1]$. We have, working modulo $(X^m+b)R$, that

$$\frac{X^m+b}{q} \cdot [c_0 + c_1 s]_q = \frac{X^m+b}{q} \cdot (\Delta_b \cdot \mathsf{msg} + p_0 u + e_0 + p_1 u s + e_1 s)$$

Applying Lemma 5, we obtain

$$\begin{aligned}
\frac{X^m+b}{q} \cdot [c_0 + c_1 s]_q &= \mathsf{msg} \cdot \left(1 + \frac{\rho}{q}\right) + \frac{X^m+b}{q} \cdot (p_0 u + e_0 + p_1 u s + e_1 s) \\
&= \mathsf{msg} + \frac{\rho}{q} \cdot \mathsf{msg} + \frac{X^m+b}{q} \cdot ((-as - e)u + aus + e_1 s) \\
&= \mathsf{msg} + \frac{\rho}{q} \cdot \mathsf{msg} + \frac{X^m+b}{q} \cdot (-eu + e_1 + e_2 s)
\end{aligned}$$

Here, the noisy term is $v = (\rho \cdot \mathsf{msg} + (X^m + b) \cdot (-eu + e_1 + e_2 s))/q$. Given Lemmas 5 and 9, it follows that

$$\begin{aligned}\|v\|_\infty^{\mathrm{can}} &\le \frac{1}{q} \cdot \left((b+1) N_b \sqrt{3n} \cdot \frac{b+1}{2} + 6(b+1)\sqrt{n}\sqrt{\sigma^2(4n/3+1)} \right) \\ &= \frac{b+1}{q} \cdot \left(\frac{\sqrt{3n}}{2} \cdot (b+1) N_b + 2\sigma n \sqrt{12 + \frac{9}{n}} \right).\end{aligned}$$

4.2 Homomorphic operations

In this section we show how homomorphic addition and multiplication are performed in the new scheme. We prove correctness of these operations and estimate the invariant noise growth. Throughout this section, $\mathtt{Ct}(\mathsf{msg}, v)$ denotes a ciphertext encrypting message $\mathsf{msg} \in R_{X^m+b}$ with invariant noise v.

Addition is the coordinate-wise sum of corresponding ciphertext components:

- $\mathtt{ComFV.Add}(\mathtt{ct}_0, \mathtt{ct}_1)$: Return $\left([\mathtt{ct}_0[0] + \mathtt{ct}_1[0]]_q, [\mathtt{ct}_0[1] + \mathtt{ct}_1[1]]_q\right)$.

It follows immediately from (4) that the invariant noise grows additively as in the lemma below.

Lemma 6 (Addition noise). *Given two ciphertexts* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_1, v_1)$ *and* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_2, v_2)$*, the function* $\mathtt{ComFV.Add}(\mathtt{ct}_1, \mathtt{ct}_2)$ *returns a ciphertext* $\mathtt{ct}_{\mathtt{Add}} = \mathtt{Ct}(\mathsf{msg}_1 + \mathsf{msg}_2, v_{\mathtt{Add}})$ *with* $\|v_{\mathtt{Add}}\|_\infty^{\mathrm{can}} \le \|v_1\|_\infty^{\mathrm{can}} + \|v_2\|_\infty^{\mathrm{can}}$.

Multiplication consists of two steps. The first one, denoted $\mathtt{ComFV.BMul}$, returns the coefficients of the ciphertext product when expressed as of a polynomial in s, namely of $(\mathtt{ct}_0[0] + \mathtt{ct}_0[1]s)(\mathtt{ct}_1[0] + \mathtt{ct}_1[1]s)$. The second step then maps the degree two term back to degree one using the relinearization technique.

- $\mathtt{ComFV.BMul}(\mathtt{ct}_0, \mathtt{ct}_1)$: Compute $c_0 = \left[\left\lfloor \frac{X^m+b}{q} \cdot \mathtt{ct}_0[0] \cdot \mathtt{ct}_1[0] \right\rceil\right]_q$,
  $c_1 = \left[\left\lfloor \frac{X^m+b}{q} \cdot (\mathtt{ct}_0[0] \cdot \mathtt{ct}_1[1] + \mathtt{ct}_0[1] \cdot \mathtt{ct}_1[0]) \right\rceil\right]_q$
  and $c_2 = \left[\left\lfloor \frac{X^m+b}{q} \cdot \mathtt{ct}_0[1] \cdot \mathtt{ct}_1[1] \right\rceil\right]_q$.
  Return $\mathtt{ct}_{\mathtt{BMul}} = (c_0, c_1, c_2)$.
- $\mathtt{ComFV.Relin}(\mathtt{ct}_{\mathtt{BMul}}, \mathtt{evk})$: Writing $\mathtt{ct}_{\mathtt{BMul}} = (c_0, c_1, c_2)$, expand c_2 in base w, namely $c_2 = \sum_{i=0}^{\ell} c_{2,i} w^i$ with $c_{2,i} \in R_w$. Compute

  $$c'_0 = \left[c_0 + \sum_{i=0}^{\ell} \mathtt{evk}[i][0] \cdot c_{2,i} \right]_q, \qquad c'_1 = \left[c_1 + \sum_{i=0}^{\ell} \mathtt{evk}[i][1] \cdot c_{2,i} \right]_q$$

  and output $c_{\mathtt{Relin}} = (c'_0, c'_1)$.
- $\mathtt{ComFV.Mul}(\mathtt{ct}_0, \mathtt{ct}_1, \mathtt{evk})$: Return $c_{\mathtt{Mul}} = \mathtt{ComFV.Relin}(\mathtt{ComFV.BMul}(\mathtt{ct}_0, \mathtt{ct}_1), \mathtt{evk})$.

To estimate the noise growth of multiplication, we analyze each step above separately. First, we provide an upper bound on the noise introduced by $\mathtt{ComFV.BMul}$.

Lemma 7 (Noise after `ComFV.BMul`). *Given two ciphertexts* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_1, v_1)$ *and* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_2, v_2)$*, the function* $\mathtt{ComFV.BMul}(\mathtt{ct}_1, \mathtt{ct}_2)$ *returns a triple* $\mathtt{ct}_{\mathtt{BMul}} = (c_0, c_1, c_2)$ *such that*

$$\frac{X^m + b}{q} \cdot \left[c_0 + c_1 s + c_2 s^2\right]_q = \mathsf{msg}_1 \cdot \mathsf{msg}_2 + v_{\mathtt{BMul}} + g(X^m + b)$$

for some $g \in R$ *and the noise* $v_{\mathtt{BMul}}$ *satisfying*

$$\begin{aligned}\|v_{\mathtt{BMul}}\|_\infty^{\mathrm{can}} \leq (b+1)\sqrt{3n + 2n^2}\,(\|v_1\|_\infty^{\mathrm{can}} + \|v_2\|_\infty^{\mathrm{can}}) + 3\,\|v_1\|_\infty^{\mathrm{can}} \cdot \|v_2\|_\infty^{\mathrm{can}} \\ + \frac{b+1}{q}\sqrt{3n + 2n^2 + 4n^3/3}\end{aligned}$$

Proof. According to the description of `ComFV.BMul`, every component c_i of $\mathtt{ct}_{\mathtt{BMul}}$ contains a rounding error r_i, $\|r_i\|_\infty \leq 1/2$. Thus, decrypting $\mathtt{ct}_{\mathtt{BMul}}$ leads to

$$\frac{X^m + b}{q} \cdot \left[c_0 + c_1 s + c_2 s^2\right]_q = \left(\frac{X^m + b}{q}\right)^2 \cdot \mathtt{ct}_1(s) \cdot \mathtt{ct}_2(s) + r + g(X^m + b),$$

where $r = (X^m + b)(r_0 + r_1 s + r_2 s^2)/q$ and $g \in R$. According to Appendix A, the variance of $\left\|r_0 + r_1 s + r_2 s^2\right\|_\infty^{\mathrm{can}}$ is equal to $n/12 + n^2/18 + n^3/27$. It follows that

$$\begin{aligned}\|r\|_\infty^{\mathrm{can}} &\leq \frac{b+1}{q} 6\sqrt{n/12 + n^2/18 + n^3/27} \\ &= \frac{b+1}{q}\sqrt{3n + 2n^2 + 4n^3/3}\end{aligned}$$

Since $(X^m + b) \cdot \mathtt{ct}_i(s)/q = \mathsf{msg}_i + v_i + g_i(X^m + b)$ for some $g_i \in R$, expanding the previous expression results in

$$\begin{aligned}\frac{X^m + b}{q} \cdot \left[c_0 + c_1 s + c_2 s^2\right]_q &= \mathsf{msg}_1 \cdot \mathsf{msg}_2 + v_2(\mathsf{msg}_1 + g_1(X^m + b)) \\ &\quad + v_1(\mathsf{msg}_2 + g_2(X^m + b)) \\ &\quad + v_1 v_2 + r \\ &\quad + (\mathsf{msg}_1 \cdot g_2 + \mathsf{msg}_2 \cdot g_1 + g)(X^m + b) \\ &\quad + g_1 g_2 (X^m + b)^2 \\ &= \mathsf{msg}_1 \cdot \mathsf{msg}_2 + v_{\mathtt{BMul}} + h(X^m + b).\end{aligned}$$

Notice that $\mathtt{ct}_i[0]$ and $\mathtt{ct}_i[1]$ should be indistinguishable from samples generated by $\mathcal{U}_q$ according to the decision Ring-LWE problem. The variance of $\mathtt{ct}_i[0] + \mathtt{ct}_i[1] \cdot s$ is thus $q^2 n/12 + q^2 n^2/18$. Hence, it follows

$$\begin{aligned}\|\mathsf{msg}_i + g_i(X^m + b)\|_\infty^{\mathrm{can}} &= \left\|\frac{X^m + b}{q} \cdot \mathtt{ct}_i(s) - v_i\right\|_\infty^{\mathrm{can}} \\ &\leq \frac{b+1}{q} \cdot q\sqrt{3n + 2n^2} + \|v_i\|_\infty^{\mathrm{can}} \\ &= (b+1)\sqrt{3n + 2n^2} + \|v_i\|_\infty^{\mathrm{can}}.\end{aligned}$$

Hence, the noisy term $v_{\mathtt{BMul}}$ satisfies

$$\begin{aligned}\|v_{\mathtt{BMul}}\|_\infty^{\mathrm{can}} \leq\ & \|v_2\|_\infty^{\mathrm{can}} \cdot \left((b+1)\sqrt{3n+2n^2} + \|v_1\|_\infty^{\mathrm{can}}\right)\\ & + \|v_1\|_\infty^{\mathrm{can}} \cdot \left((b+1)\sqrt{3n+2n^2} + \|v_2\|_\infty^{\mathrm{can}}\right)\\ & + \|v_1\|_\infty^{\mathrm{can}} \cdot \|v_2\|_\infty^{\mathrm{can}} + \frac{b+1}{q}\sqrt{3n+2n^2+4n^3/3}\\ =\ & (b+1)\sqrt{3n+2n^2}\,(\|v_1\|_\infty^{\mathrm{can}} + \|v_2\|_\infty^{\mathrm{can}}) + 3\,\|v_1\|_\infty^{\mathrm{can}} \cdot \|v_2\|_\infty^{\mathrm{can}}\\ & + \frac{b+1}{q}\sqrt{3n+2n^2+4n^3/3}\end{aligned}$$

The next lemma provides an upper bound on the noise introduced after relinearization.

Lemma 8 (Noise after $\mathtt{ComFV.Relin}$). *Given a triple* $\mathtt{ct} = (c_0, c_1, c_2)$ *encrypting a message* msg *and containing noise* v*, the relinearization function returns a ciphertext* $\mathtt{ct}_{\mathtt{Relin}} = \mathtt{Ct}(\mathsf{msg}, v_{\mathtt{Relin}})$ *with*

$$\|v_{\mathtt{Relin}}\|_\infty^{\mathrm{can}} \leq \|v\|_\infty^{\mathrm{can}} + \frac{b+1}{q} \cdot \sigma n w \sqrt{3(\ell+1)}.$$

Proof. As in the proof of Lemma 7, we scale down the output of relinearization

$$\begin{aligned}\frac{X^m+b}{q} \cdot [\mathtt{ct}_{\mathtt{Relin}}(s)]_q &= \frac{X^m+b}{q} \cdot [c_0' + c_1' s]_q\\ &= \frac{X^m+b}{q} \cdot \left(c_0 + c_1 s + c_{2,i}\sum_{i=0}^{\ell} \mathtt{evk}[i][0] + \mathtt{evk}[i][1] \cdot s\right)\\ &\quad + g(X^m+b)\\ &= \frac{X^m+b}{q} \cdot \left(c_0 + c_1 s - \sum_{i=0}^{\ell} e_i c_{2,i} + s^2 \sum_{i=0}^{\ell} w^i c_{2,i}\right)\\ &\quad + \left(\sum_{i=0}^{\ell} g_i c_{2,i} + g\right)(X^m+b)\end{aligned}$$

Recall that by definition $\sum_i w^i c_{2,i} = c_2$. Thus, replacing $\sum_i g_i c_{2,i} + g$ by $\tilde{g}$, we obtain for some $h \in R$

$$\begin{aligned}\frac{X^m+b}{q} \cdot [\mathtt{ct}_{\mathtt{Relin}}(s)]_q &= \frac{X^m+b}{q} \cdot \left(c_0 + c_1 s + c_2 s^2 - \sum_{i=0}^{\ell} e_i c_{2,i}\right) + \tilde{g}(X^m+b)\\ &= \mathsf{msg} + v - \frac{X^m+b}{q} \cdot \sum_{i=0}^{\ell} e_i c_{2,i} + (\tilde{g}+h)(X^m+b)\end{aligned}$$

As a result, $v_{\mathtt{Relin}} = v - \frac{X^m+b}{q} \cdot \sum_{i=0}^{\ell} e_i c_{2,i}$. Given that $c_{2,i}$'s look uniformly random in R_w, the variance of $\sum_{i=0}^{\ell} e_i c_{2,i}$ is equal to $(\ell+1)(w\sigma n)^2/12$. Hence, we obtain

$$\begin{aligned}
\|v_{\mathtt{Relin}}\|_\infty^{\mathrm{can}} &\le \|v\|_\infty^{\mathrm{can}} + \frac{b+1}{q} \cdot \sum_{i=0}^{\ell} \|e_i c_{2,i}\|_\infty^{\mathrm{can}} \\
&\le \|v\|_\infty^{\mathrm{can}} + \frac{b+1}{q} \cdot 6\sigma n w \sqrt{\frac{\ell+1}{12}} \\
&= \|v\|_\infty^{\mathrm{can}} + \frac{b+1}{q} \cdot \sigma n w \sqrt{3(\ell+1)}.
\end{aligned}$$

Combining the two previous lemmas, we deduce the total noise growth after homomorphic multiplication in the following Theorem.

Theorem 2 (Multiplication noise). *Given two ciphertexts* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_1, v_1)$ *and* $\mathtt{ct}_1 = \mathtt{Ct}(\mathsf{msg}_2, v_2)$*, the function* $\mathtt{ComFV.Mul}(\mathtt{ct}_1, \mathtt{ct}_2, \mathtt{evk})$ *outputs a ciphertext* $\mathtt{ct}_{\mathtt{Mul}} = \mathtt{Ct}(\mathsf{msg}_1 \cdot \mathsf{msg}_2, v_{\mathtt{Mul}})$ *with*

$$\begin{aligned}
\|v_{\mathtt{Mul}}\|_\infty^{\mathrm{can}}&(b+1)\sqrt{3n+2n^2}\left(\|v_1\|_\infty^{\mathrm{can}} + \|v_2\|_\infty^{\mathrm{can}}\right) + 3\,\|v_1\|_\infty^{\mathrm{can}} \cdot \|v_2\|_\infty^{\mathrm{can}} \\
&+ \frac{b+1}{q}\sqrt{3n+2n^2+4n^3/3} + \frac{b+1}{q} \cdot \sigma n w \sqrt{3(\ell+1)}
\end{aligned}$$

with very high probability.

We note that the dominating term here is the first term and not the term containing the product of the canonical norms of the multiplicands since the canonical norms are smaller than 1/2 when the ciphertext can be decrypted correctly.

5 Application to Image Processing

In this section we apply the ComFV scheme to the image processing use case [10]. For this application, as with any other, we need to take into account two constraints regarding computation correctness. Firstly, the coefficients of encrypted encodings can increase in absolute value after arithmetic operations and reach some bound, say, B. To decode these resulting encodings, B must be smaller than $(b^{n/m}+1)/2$ as described in Section 3. Secondly, the invariant noise of encryptions grows as well according to the heuristic estimates of Section 4. To decrypt the resulting output, this noise should be smaller than 1/2 as shown in Lemma 4.

Homomorphic Discrete Fourier Transform. We calculate the parameters of the new scheme which are compatible with the image processing pipeline given in [10].

The circuit takes input images as 8-bit integer vectors $\mathbf{a} \in \mathbb{Z}^d$ for some $d \mid m$. Then, it performs the discrete Fourier transform (DFT), $\mathcal{F}$, that maps

$\mathbf{a} = (a_0, \ldots, a_{d-1})$ to a vector $\mathbf{a}' \in \mathbb{Z}^d$ such that $\mathbf{a}'[j] = \sum_{i=0}^{d-1} a_i \zeta_d^{ij}$, where ζ_d is a primitive d-th root of unity. The resulting vector is then multiplied coordinatewise by some encrypted 8-bit integers and mapped back to $\mathbb{Z}^d$ via the inverse DFT.

Using the ComFV scheme, decoding is correct as long as $b^{n/m} + 1 > 2^{17} d^2$, for details see [10]. Notably, scalar multiplication by a root of unity is no longer noise preserving as in [10], where ζ_m^i is encoded by some power of X. According to (2), ζ_m^i is mapped to some polynomials $z(X)$ such that $\|z\|_\infty^{\text{can}} \leq bn/2m$. Therefore, the canonical norm of the invariant noise is increasing after every multiplication by ζ_m^i.

Computing $\mathcal{F}$ and $\mathcal{F}^{-1}$, we resort to the mixed Fourier transform (MFT) method that combines both the fast Fourier transform (FFT) and the naive Fourier transform (NFT). In the NFT, the input vector is multiplied by a matrix $F = \left(\zeta_d^{ij}\right)_{i,j}$ that needs $O(d^2)$ multiplications and only one multiplicative level. The FFT method calls recursively smaller size DFT's such that the ith coordinate of the DFT output is then given as

$$\mathcal{F}(\mathbf{a})[i] = \mathcal{F}(a_0, \ldots, a_{d/2-1}) + \zeta_d^i \cdot \mathcal{F}(a_{d/2}, \ldots, a_{d-1}).$$

The FFT reduces the number of multiplications to $O(d \log d)$ but needs $O(\log d)$ multiplicative levels. Thus, the FFT introduces more noise than the NFT but it is computationally faster. The MFT approach consists in computing the FFT recursion up to some dimension $\tilde{d} \leq d$ and then computing NFT.

We applied the ComFV scheme to 6 DFT dimensions d given in [10]. As shown in Table 1, the ciphertext size is reduced in all cases. However, only the FFT method was used in [10] while we resort sometimes to a slower MFT circuit for $d \in 2^8, 2^{12}, 2^{13}$.

Table 1. Ciphertext size comparison between our encoding and [10]. All parameters are taken to be compatible with a d-dimensional DFT circuit and the security level λ.

d	$\tilde{d}$	b	n	$\log q$	λ	ct size	ct size[10]
2^4	1	30	2^{12}	149	119	149 kB	300 kB
2^6	1	30	2^{12}	149	119	149 kB	300 kB
2^8	2^4	30	2^{13}	147	438	294 kB	300 kB
2^{10}	1	132	2^{13}	222	206	444 kB	768 kB
2^{12}	2^8	472	2^{14}	180	1004	720 kB	768 kB
2^{13}	2^{13}	$\simeq 2^{22}$	2^{14}	172	1082	688 kB	768 kB

6 Comparison with FV: regular circuits

To estimate the performance of ComFV in a general setting and fairly compare it with the original FV scheme and the work of [6], we resort to regular circuits as introduced in [11]. These circuits have already been used in [6] for the same purpose.

A regular circuit consists of D computational levels where each level contains $A \in \{0, 3, 10\}$ addition levels, requiring 2^A inputs, followed by one multiplication. Therefore in total the number of inputs required is $2^{D(A+1)}$. Each circuit input is given by a complex number with real and imaginary parts from $(-U, U)$ for some $U \in \{2^8, 2^{16}, 2^{32}, 2^{64}\}$. We will always use a precision of 16 fractional bits in this paper which in the case of a complex number refers to both the real and complex parts independently.

Our aim is to compare ComFV to the previously best known scheme allowing native complex inputs as well as to the state of the art when encoding the real and imaginary parts separately [6]. We will compare this method with our method where we use the same encoding of the complex number as a cyclotomic integer. We chose $m = 4$ as this is the minimal m for which $\mathbb{Z}[\zeta]$ is dense in $\mathbb{C}$ and it allows us to use $b = 4^h$ for some $h \in \mathbb{N}$, taking $\alpha = 2^{h/2}$ if h is even and $\alpha = 2^{(h(n+4)-4)/8}(2^{hn/4} - 1)$ if h is odd. We also use $m = 4$ when using FV and one may wonder if taking a larger m is better. However, we found that using larger m in this case gave the same depths and only increased the time to encode a complex number.

For the current state of the art we use the scheme of Chen et al. [6], which we call CLPX, and encode the real and imaginary parts of our complex number separately. Thus an encryption now consists of two ciphertext pairs and addition is performed component-wise while we use the Karatsuba algorithm to perform multiplication using only three calls to the multiplication algorithm of the underlying scheme. We use the same values for n and q for comparison so that ciphertexts will be twice as large compared to our work. The fractional encoder is used to encode the real and imaginary parts so we use $m = 2$ in this case. For the optimal value of b we restrict our search space to powers of 2, since we require a precision of 2^{-16}, the simplest way to ensure correct decoding at depth D is to require $2^{16D} \mid b^{n/4}$ so taking b a power of two looks a good fit. We again compare this approach with ours, in this case we also use the fractional encoder.

We computed the theoretical and heuristic maximal depth of a regular circuit which can be reached using FV, the CLPX approach of using plaintext modulus $X - b$ and our ComFV with parameters n, q, σ given in the SEAL library [5] and the relinearization base $w = 2^{32}$. Our results are presented in Tables 2 and 3. In the tables we also give a value for b (or t) which allows one to reach this maximal depth, this b is very often not unique and in this case we give the smallest b for which there is a decryption error at the next level. To find a heuristic estimate of the maximal depth that can be reached in each scheme we take a carefully chosen complex number and use this as the complex number given for all inputs of the circuit. One reason for this can be seen in the table of results, Table 3, where we see that for $A = 10$, depths of 14 can be achieved, this requires $2^{14 \cdot 11} = 2^{154}$ inputs, meaning using different inputs would be completely infeasible in practice. Another good reason for choosing all inputs to be the same is that during addition there is no cancellation occurring, indeed the A levels of addition simply become the worst case of scaling by 2^A. The precise complex number we chose depends on the encoding scheme but essentially one finds one with an encoding which has many large coefficients. If the fractional encoder is used then we take the complex number to be $(U - 2^{-16})(1 + \mathbf{i})$ while when using the cyclotomic integer approximation approach it is a matter of trial and error but this need only be done once for each U and m.

Table 2. Maximal theoretical regular circuit depths of `FV` (D_O) with the approximation encoding, the CLPX approach encrypting the real and imaginary parts separately (D_M), `ComFV` with the approximation encoding (D_A) and the fractional encoding (D_F) depending on input size (U), number of additions per level (A), n and q. Corresponding t and b's are provided.

	n	4096			8192			16384			32768		
	$\log q$	116			226			435			889		
	A	0	3	10	0	3	10	0	3	10	0	3	10
$U = 2^8$	D_O	1	0	0	1	1	1	2	2	1	3	3	2
	t_O	2^{34}	–	–	2^{34}	2^{40}	2^{54}	2^{68}	2^{86}	2^{54}	2^{135}	2^{177}	2^{128}
	D_M	4	3	3	9	8	6	12	12	11	15	14	14
	b_M	2	2	2	2^3	2^2	2	2^9	2^9	2^5	2^{33}	2^{17}	2^{17}
	D_A	5	4	3	9	8	6	11	11	10	14	13	12
	b_A	2^2	2^2	2^2	2^6	2^4	2^2	2^{10}	2^{12}	2^{10}	2^{34}	2^{24}	2^{20}
	D_F	5	4	3	9	8	7	11	11	10	14	14	13
	b_F	2	2	2	2^5	2^3	2^2	2^9	2^9	2^8	2^{33}	2^{33}	2^{29}
$U = 2^{16}$	D_O	1	0	0	1	1	1	2	2	1	3	3	2
	t_O	2^{34}	–	–	2^{34}	2^{40}	2^{54}	2^{67}	2^{85}	2^{54}	2^{134}	2^{176}	2^{127}
	D_M	4	3	3	9	8	6	12	12	11	14	14	14
	b_M	2	2	2	2^3	2^2	2	2^9	2^9	2^5	2^{18}	2^{18}	2^{18}
	D_A	5	4	3	9	8	6	11	11	10	14	13	12
	b_A	2^2	2^2	2^2	2^6	2^4	2^2	2^{10}	2^{12}	2^{10}	2^{34}	2^{24}	2^{20}
	D_F	5	4	3	9	8	7	11	11	10	14	13	12
	b_F	2	2	2	2^5	2^3	2^3	2^9	2^{12}	2^{10}	2^{34}	2^{23}	2^{19}
$U = 2^{32}$	D_O	0	0	0	1	1	1	1	1	1	2	2	2
	t_O	–	–	–	2^{65}	2^{71}	2^{85}	2^{65}	2^{71}	2^{85}	2^{130}	2^{148}	2^{190}
	D_M	4	3	3	8	8	6	11	11	10	14	14	13
	b_M	2	2	2	2^3	2^3	2	2^9	2^9	2^5	2^{34}	2^{34}	2^{17}
	D_A	5	4	3	8	8	6	11	10	9	13	13	12
	b_A	2^2	2^2	2^2	2^6	2^6	2^2	2^{18}	2^{10}	2^8	2^{34}	2^{40}	2^{28}
	D_F	5	4	3	8	8	6	11	10	9	13	13	12
	b_F	2^2	2	2	2^5	2^5	2^2	2^{17}	2^{10}	2^7	2^{33}	2^{39}	2^{27}
$U = 2^{64}$	D_O	–	–	–	0	0	0	1	1	1	2	1	1
	t_O	–	–	–	–	–	–	2^{129}	2^{135}	2^{149}	2^{258}	2^{135}	2^{149}
	D_M	4	3	3	8	7	6	10	10	10	13	13	12
	b_M	2	2	2	2^5	2^3	2^2	2^9	2^9	2^9	2^{33}	2^{33}	2^{17}
	D_A	4	4	3	7	7	6	10	10	9	12	12	11
	b_A	2^2	2^2	2^2	2^6	2^6	2^4	2^{18}	2^{18}	2^{12}	2^{34}	2^{36}	2^{22}
	D_F	4	4	3	7	7	6	10	10	9	12	12	11
	b_F	2^2	2^2	2	2^5	2^5	2^3	2^{17}	2^{18}	2^{11}	2^{33}	2^{36}	2^{22}

From Table 3 we see that in all cases our methods greatly outperform the best scheme natively encrypting complex numbers. At a minimum we can achieve 5 times the depth and for larger n our method becomes even more efficient as the amount of plaintext space not being efficiently used only grows in the current

Table 3. Maximal heuristic regular circuit depths of the original FV scheme with native complex inputs (D_O), the CLPX approach encrypting the real and imaginary parts separately (D_M), ComFV with the approximation encoding (D_A) and the fractional encoding (D_F) depending on input size (U), number of additions per level (A), n and q. A corresponding t or b is provided.

	n	4096			8192			16384			32768		
	$\log q$	116			226			435			889		
	A	0	3	10	0	3	10	0	3	10	0	3	10
$U = 2^8$	D_O	1	1	0	1	1	1	2	2	2	3	3	2
	t_O	2^{35}	2^{41}	2^{18}	2^{35}	2^{41}	2^{55}	2^{70}	2^{88}	2^{130}	2^{164}	2^{182}	2^{202}
	D_M	6	5	4	10	9	8	13	12	11	15	15	14
	b_M	2	2	2	2^5	2^4	2^2	2^{16}	2^{14}	2^{10}	2^{37}	2^{34}	2^{31}
	D_A	6	5	4	9	9	7	12	11	10	14	13	13
	b_A	2^2	2^2	2^2	2^6	2^6	2^6	2^{18}	2^{18}	2^{10}	2^{40}	2^{40}	2^{38}
	D_F	6	5	4	9	9	7	12	12	10	14	14	13
	b_F	2	2	2	2^4	2^4	2^2	2^{16}	2^{15}	2^8	2^{32}	2^{33}	2^{33}
$U = 2^{16}$	D_O	1	1	0	1	1	1	2	2	2	3	3	2
	t_O	2^{35}	2^{41}	2^{18}	2^{35}	2^{41}	2^{55}	2^{70}	2^{88}	2^{130}	2^{164}	2^{173}	2^{201}
	D_M	6	5	4	10	9	7	12	12	11	15	14	13
	b_M	2	2	2	2^5	2^4	2^2	2^{17}	2^{14}	2^{10}	2^{37}	2^{38}	2^{35}
	D_A	6	5	4	9	9	7	12	11	10	14	13	13
	b_A	2^2	2^2	2^2	2^6	2^6	2^6	2^{18}	2^{18}	2^{10}	2^{40}	2^{40}	2^{38}
	D_F	6	5	4	9	9	7	12	11	10	14	13	13
	b_F	2^2	2	2	2^5	2^6	2^3	2^{17}	2^{15}	2^{10}	2^{33}	2^{41}	2^{37}
$U = 2^{32}$	D_O	0	0	0	1	1	1	1	1	1	2	2	2
	t_O	2^{33}	2^{33}	2^{33}	2^{65}	2^{71}	2^{84}	2^{65}	2^{71}	2^{85}	2^{206}	2^{205}	2^{198}
	D_M	5	5	4	9	9	7	12	11	10	14	14	13
	b_M	2^2	2	2	2^7	2^5	2^2	2^{17}	2^{16}	2^{13}	2^{40}	2^{39}	2^{35}
	D_A	5	5	4	8	8	7	11	10	10	13	13	12
	b_A	2^2	2^2	2^2	2^6	2^6	2^6	2^{18}	2^{18}	2^{14}	2^{40}	2^{40}	2^{40}
	D_F	5	5	4	9	8	7	11	10	10	13	13	12
	b_F	2^2	2^2	2	2^9	2^6	2^4	2^{17}	2^{15}	2^{14}	2^{33}	2^{41}	2^{38}
$U = 2^{64}$	D_O	—	—	—	0	0	0	1	1	1	2	1	1
	t_O	—	—	—	2^{65}	2^{65}	2^{65}	2^{129}	2^{135}	2^{149}	2^{258}	2^{266}	2^{262}
	D_M	5	5	4	8	8	7	11	11	10	13	13	12
	b_M	2^2	2^2	2	2^9	2^6	2^3	2^{19}	2^{18}	2^{13}	2^{44}	2^{41}	2^{39}
	D_A	5	4	4	8	7	7	10	10	9	12	12	12
	b_A	2^4	2^4	2^2	2^{10}	2^6	2^6	2^{18}	2^{18}	2^{14}	2^{40}	2^{40}	2^{44}
	D_F	5	5	4	8	8	7	10	10	9	12	12	12
	b_F	2^3	2^3	2^2	2^9	2^9	2^6	2^{17}	2^{18}	2^{14}	2^{33}	2^{41}	2^{43}

solution. The CLPX method on the other hand is able to achieve slightly larger depths than our scheme, at most one more for the largest n we consider. Where our method improves is on efficiency, we effectively halve the ciphertext size and are expected to be roughly three times faster due to the fact that we can use one multiplication operation per level whereas the CLPX approach requires three.

7 Conclusion

We constructed a new encoding algorithm for complex data values and a corresponding somewhat homomorphic encryption scheme by utilizing a polynomial plaintext modulus of the form $X^m + b$. This choice allows for a much better use of the available plaintext space and much slower noise growth compared to existing solutions encrypting complex numbers. As a result, for the same ciphertext modulus q and degree n, we can homomorphically evaluate between 5 and 12 times deeper circuits compared to existing solutions based on FV and natively encoding complex numbers. In comparison to the state of the art, which encrypts the real and imaginary parts of the complex numbers separately, our method reduces the size of ciphertexts by a factor of 2 making our scheme at least twice as efficient in time and three times more efficient in space.

References

1. Barnett, A., Santokhi, J., Simpson, M., Smart, N.P., Stainton-Bygrave, C., Vivek, S., Waller, A.: Image classification using non-linear support vector machines on encrypted data (2017), cryptology ePrint Archive: Report 2017/857
2. Bonte, C., Bootland, C., Bos, J.W., Castryck, W., Iliashenko, I., Vercauteren, F.: Faster homomorphic function evaluation using non-integral base encoding. In: CHES 2017. LNCS, vol. 10529, pp. 579–600. Springer, Heidelberg (Sep 2017)
3. Bos, J.W., Castryck, W., Iliashenko, I., Vercauteren, F.: Privacy-friendly forecasting for the smart grid using homomorphic encryption and the group method of data handling. In: AFRICACRYPT 17. LNCS, vol. 10239, pp. 184–201. Springer, Heidelberg (May 2017)
4. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS 2012. pp. 309–325. ACM (Jan 2012)
5. Chen, H., Laine, K., Player, R.: Simple encrypted arithmetic library - SEAL v2.1. In: FC 2017. vol. 10323, pp. 3–18. Springer, Heidelberg (2017)
6. Chen, H., Laine, K., Player, R., Xia, Y.: High-precision arithmetic in homomorphic encryption. In: Smart, N.P. (ed.) CT-RSA 2018. LNCS, vol. 10808, pp. 116–136. Springer, Heidelberg (2018)
7. Cheon, J.H., Jeong, J., Lee, J., Lee, K.: Privacy-preserving computations of predictive medical models with minimax approximation and non-adjacent form. In: FC 2017. vol. 10323, pp. 53–74. Springer, Heidelberg (2017)
8. Cheon, J.H., Kim, A., Kim, M., Song, Y.S.: Homomorphic encryption for arithmetic of approximate numbers. In: ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 409–437. Springer, Heidelberg (Dec 2017)
9. Costache, A., Smart, N.P.: Which ring based somewhat homomorphic encryption scheme is best? In: CT-RSA 2016. LNCS, vol. 9610, pp. 325–340. Springer, Heidelberg (Feb / Mar 2016)
10. Costache, A., Smart, N.P., Vivek, S.: Faster homomorphic evaluation of discrete Fourier transforms. In: FC 2017. LNCS, vol. 10322, pp. 517–529 (2017)
11. Costache, A., Smart, N.P., Vivek, S., Waller, A.: Fixed-point arithmetic in SHE schemes. In: SAC 2016. LNCS, vol. 10532, pp. 401–422. Springer, Heidelberg (Aug 2016)
12. Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: Manual for using homomorphic encryption for bioinformatics. Tech. rep., MSR-TR-2015-87, Microsoft Research (2015)
13. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144 (2012), `http://eprint.iacr.org/2012/144`

14. Geihs, M., Cabarcas, D.: Efficient integer encoding for homomorphic encryption via ring isomorphisms. In: LATINCRYPT 2014. LNCS, vol. 8895, pp. 48–63. Springer, Heidelberg (Sep 2015)
15. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: 41st ACM STOC. pp. 169–178. ACM Press (May / Jun 2009)
16. Gentry, C., Halevi, S., Smart, N.P.: Homomorphic evaluation of the AES circuit. In: CRYPTO 2012. LNCS, vol. 7417, pp. 850–867. Springer, Heidelberg (Aug 2012)
17. Hoffstein, J., Pipher, J., Silverman, J.H.: Ntru: A ring-based public key cryptosystem. In: Algorithmic Number Theory, Third International Symposium, ANTS-III. pp. 267–288. Springer, Heidelberg (1998)
18. Lauter, K., López-Alt, A., Naehrig, M.: Private computation on encrypted genomic data. In: LATINCRYPT 2014. LNCS, vol. 8895, pp. 3–27 (Sep 2015)
19. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (May 2010)
20. Naehrig, M., Lauter, K.E., Vaikuntanathan, V.: Can homomorphic encryption be practical? In: ACM Cloud Computing Security Workshop – CCSW. pp. 113–124. ACM (2011)

A The canonical norm

This appendix closely follows Appendix A.5 of the ePrint version of [6].

Let $K = \mathbb{Q}[X]/(f(X))$ be a cyclotomic number field where, as usual, $f(X) = X^n + 1$ is the $2n$-cyclotomic polynomial, n a power of two. We denote the ring of integers of K by R, i.e. $R = \mathbb{Z}[X]/(f(X))$. Let R_a be the reduction of R modulo an ideal (a). If a is a natural number $R_a = \mathbb{Z}_a[X]/(f(X))$ and we take representatives of $\mathbb{Z}/a\mathbb{Z}$ from the half-open interval $[-a/2, a/2)$.

For any $a = \sum_i a_i X^i \in K$, *the infinity norm* $\|a\|_\infty$ is defined as $\max_i |a_i|$. We denote by δ_R the upper bound on $\|ab\|_\infty / \|a\|_\infty \cdot \|b\|_\infty$ for any $a, b \in R$. This bound is called *the expansion factor* of R. For a our ring of cyclotomic integers R, the expansion factor is $\delta_R = n$. Let ζ is a complex primitive $2n$-th root of unity. We define *the canonical norm* as

$$\|a\|_\infty^{\text{can}} = \left\| \left(a(\zeta), a(\zeta^3), \ldots, a(\zeta^{2n-1}) \right) \right\|_\infty .$$

It is easy to check that the canonical norm satisfies

$$\|a\|_\infty \leq \|a\|_\infty^{\text{can}}, \qquad \|a+b\|_\infty^{\text{can}} \leq \|a\|_\infty^{\text{can}} + \|b\|_\infty^{\text{can}}, \qquad \|ab\|_\infty^{\text{can}} \leq \|a\|_\infty^{\text{can}} \cdot \|b\|_\infty^{\text{can}} .$$

The last inequality implies that the canonical norm leads to tighter bounds than the infinity norm [19].

Canonical norm of random polynomials We will need to bound the canonical norm of random polynomials whose coefficients are generated from a discrete Gaussian or uniform distributions. We follow a heuristic approach given in [16, A.5], which was already used in [9, 5, 6] for an analysis of the FV scheme.

Let $a \in R$ be a polynomial such that its coefficients are chosen independently from some zero-mean distribution with standard deviation σ. For this purpose, we use the following distributions

– a discrete Gaussian distribution $\mathcal{D}(\sigma^2)$ with PMF proportional to $\exp(-\left|x\right|^2/2\sigma^2)$,
– the uniform distribution $\mathcal{U}_3$ over the ternary set $\{-1,0,1\}$,
– the uniform distribution $\mathcal{U}_q$ over $\mathbb{Z}_q$,
– the uniform distribution $\mathcal{U}_{\mathrm{rnd}}$ over the interval $(-1/2,1/2]$.

By the definition of the canonical norm, we need to compute $a(\zeta_{2n}^i)$. The evaluation $a(\zeta^i)$ is the inner product between the coefficient vector of a and the fixed vector $\left(1,\zeta^i,\ldots,\zeta^{i(n-1)}\right)$, which has Euclidean norm $\sqrt{n}$. Hence, the random variable $a(\zeta_{2n}^i)$ has variance $V=\sigma^2 n$ by the Cauchy-Schwartz inequality.

When $a_i \leftarrow \mathcal{D}(\sigma^2)$ then the coefficients have variance $\simeq \sigma^2$ and thus the variance of $a(\zeta_{2n}^i)$ is $V_{\mathcal{D}} \simeq \sigma^2 n$. If $a_i \leftarrow \mathcal{U}_3$ then the coefficients have variance $2/3$ and thus the total variance is $V_{\mathcal{U}_3}=2n/3$. By analogy, $V_{\mathcal{U}_q} \lesssim q^2 n/12$ as the a_i has variance roughly $q^2/12$. Finally, the variance of $a_i \leftarrow \mathcal{U}_{\mathrm{rnd}}$ is equal to $1/12$, so $V_{\mathcal{U}_{\mathrm{rnd}}}=n/12$.

Since $a(\zeta_{2n}^i)$ is the sum of independently distributed complex variables, by the law of large numbers it is distributed similarly to a complex Gaussian random variable of variance V. Therefore, given that $\mathrm{erfc}(6)\simeq 2^{-55}$, we can use $6\sqrt{V}$ as a high-probability bound on $a(\zeta_{2n}^i)$. Since in practice $n\geq 2^{12}$, this bound is good enough to claim that $\|a\|_\infty^{\mathrm{can}} \leq 6\sqrt{V}$ with very high probability. For the distributions above, we get

$$\begin{aligned}
\|a\|_\infty^{\mathrm{can}} &\leq 6\sigma\sqrt{n}, \quad a_i \leftarrow \mathcal{D}(\sigma^2),\\
\|a\|_\infty^{\mathrm{can}} &\leq 2\sqrt{6n}, \quad a_i \leftarrow \mathcal{U}_3,\\
\|a\|_\infty^{\mathrm{can}} &\leq q\sqrt{3n}, \quad a_i \leftarrow \mathcal{U}_q,\\
\|a\|_\infty^{\mathrm{can}} &\leq \sqrt{3n}, \quad a_i \leftarrow \mathcal{U}_{\mathrm{rnd}}.
\end{aligned}$$

We also need to bound the canonical norm of a product of two random polynomials a and b whose coefficients are independently sampled from zero-mean distributions with variances σ_1^2,σ_2^2, respectively. Writing the product ab mod (X^n+1) with relation to the power basis of R, we obtain

$$\begin{pmatrix} a_0 & -a_{n-1} & \ldots & -a_1 \\ \vdots & \vdots & & \vdots \\ a_{n-1} & a_{n-2} & \ldots & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ \vdots \\ b_{n-1} \end{pmatrix} = \begin{pmatrix} g_0 \\ \vdots \\ g_{n-1} \end{pmatrix}.$$

Hence, the product coefficients are equal to

$$g_k = \sum_{i=0}^{k} a_i b_{k-i} - \sum_{i=k+1}^{n-1} a_i b_{k+n-i}.$$

for any $k\in[0,\ldots,n-1]$. Since the coefficient distributions are independent and have zero mean, the product of any pair a_i, b_j has variance $\sigma_1^2\sigma_2^2$ and zero mean. Hence, the variance of each coefficient g_k is equal to $n\sigma_1^2\sigma_2^2$. Following the above reasoning, the canonical norm of $g(\zeta^i)$ is thus bounded by

$$\|ab\|_\infty^{\mathrm{can}} \leq 6n\sigma_1\sigma_2.$$

This means that the variance of the coefficients of ue where $u \leftarrow \chi_k$ and $e \leftarrow \chi_e$ is approximately $2\sigma^2 n/3$. We can now give the variance of the term appearing in the proof of Theorem 1.

Lemma 9. *Let $e, e_1, e_2 \leftarrow \chi_e$ and $u, s \leftarrow \chi_k$. Then the variance of the coefficients of $-eu + e_1 + e_2 s$ is very close to $\sigma^2(4n/3 + 1)$.*

Proof. We have just seen that the variance of the coefficients of both $-eu$ and $e_2 s$ is approximately $2\sigma^2 n/3$ while the variance of the coefficients of e_1 is approximately σ^2. Because they are independent, we can sum the variances to give the result.