Indistinguishability Obfuscation Without Maps: Attacks and Fixes for Noisy Linear FE

Shweta Agrawal[1] and Alice Pellet-Mary[2]

[1] IIT Madras, India
shweta.a@gmail.com
[2] imec-COSIC, KU Leuven, Belgium
alice.pelletmary@kuleuven.be

Abstract. Candidates of *Indistinguishability Obfuscation* (iO) can be categorized as "direct" or "bootstrapping based". Direct constructions rely on high degree multilinear maps [27, 28] and provide heuristic guarantees, while bootstrapping based constructions [38, 35, 37, 7, 2, 32] rely, in the best case, on *bilinear* maps as well as new variants of the Learning With Errors (LWE) assumption and pseudorandom generators. Recent times have seen exciting progress in the construction of indistinguishability obfuscation (iO) from bilinear maps (along with other assumptions) [37, 7, 32, 2].

As a notable exception, a recent work by Agrawal [2] provided a construction for iO without using *any* maps. This work identified a new primitive, called *Noisy Linear Functional Encryption* (NLinFE) that provably suffices for iO and gave a direct construction of NLinFE from new assumptions on lattices. While a preliminary cryptanalysis for the new assumptions was provided in the original work, the author admitted the necessity of performing significantly more cryptanalysis before faith could be placed in the security of the scheme. Moreover, the author did not suggest concrete parameters for the construction.

In this work, we fill this gap by undertaking the task of thorough cryptanalytic study of NLinFE. We design two attacks that let the adversary completely break the security of the scheme. Our attacks are completely new and unrelated to attacks that were hitherto used to break other candidates of iO. To achieve this, we develop new cryptanalytic techniques which (we hope) will inform future designs of the primitive of NLinFE.

From the knowledge gained by our cryptanalytic study, we suggest modifications to the scheme. We provide a new scheme which overcomes the vulnerabilities identified before. We also provide a thorough analysis of all the security aspects of this scheme and argue why plausible attacks do not work. We additionally provide concrete parameters with which the scheme may be instantiated. We believe the security of NLinFE stands on significantly firmer footing as a result of this work.

1 Introduction

Indistinguishability Obfuscation (iO) is one of the most sought-after primitives in modern cryptography. While introduced in a work by Barak et al. in 2001 [11], the first candidate construction for this primitive was only provided in 2013 [28]. In this breakthrough work, the authors not only gave the first candidate for iO but also demonstrated its power by using it to construct the first full fledged functional encryption (FE) scheme. This work led to a deluge of ever more powerful applications of iO, ranging from classic to fantastic [33, 44, 18, 17, 13, 34, 36, 14]. Few years later, iO is widely acknowledged to be (almost) "crypto complete". We refer the reader to [35] for a detailed discussion.

However, constructions of iO have been far from perfect. The so called "first generation" constructions relied on the existence of *multilinear maps* of polynomial degree [28, 29, 46, 9], "second generation" relied on multilinear maps of constant degree [38, 35, 37], and in a sequence of exciting recent works, "third generation" candidates rely only on multilinear maps of degree 2 (i.e. bilinear maps) along with assumptions on the complexity of certain special types of pseudorandom generators and new variants of the Learning With Errors (LWE) assumption [7, 32, 2]. It is well known that degree 2 maps can be instantiated on elliptic curve groups, so this brings us closer to realizing iO from believable assumptions than ever before.

iO Without Maps: All the above constructions rely on multilinear maps of degree ≥ 2. While there exist candidates for multilinear maps of degree ≥ 3, they have been subject to many attacks [22, 24, 31, 26, 26, 42, 25, 19, 8, 43, 23, 21] and their security is poorly understood. On the other hand, bilinear maps are well understood and considered safe to use (at least in the pre-quantum world). Recent works [7, 2, 32] have come tantalizingly close to basing iO on bilinear maps while minimizing the additional assumptions required. There is hope that these efforts will converge to a candidate whose security we may trust.

While realizing iO from degree 2 maps (along with other plausible assumptions) is a very worthy goal, it is nevertheless only one approach to take. Any cryptographic primitive, especially one of such central importance, deserves to be studied from different perspectives and based on diverse mathematical assumptions. Two works (that we are aware of) attempt to construct iO without using *any* maps – one by Gentry, Jutla and Keane [30] and another by Agrawal [2]. The work by Gentry et al. [30] constructs obfuscation schemes for matrix product branching programs that are purely algebraic and employ matrix groups and tensor algebra over a finite field. They prove security of their construction against a restricted class of attacks. On the other hand, the work of Agrawal formalizes a "minimal" (as per current knowledge) primitive called "Noisy Linear Functional Encryption" (NLinFE) which is showed to imply iO and provides a direct construction for this using new assumptions on NTRU lattices, which are quite different from assumptions used so far for building multilinear maps or iO.

Comparison with Other Approaches. The instantiation of iO via Agrawal's direct construction of NLinFE (henceforth referred to simply as NLinFE) has both advantages and disadvantages compared to other cutting-edge constructions. For instance, [30] has the advantage that it constructs full fledged iO directly, while NLinFE has the advantage that untested assumptions are used to construct a much *simpler* primitive. Next, consider constructions that use bilinear maps [7, 2, 32]. On the positive side, NLinFE has potential to be quantum secure, which evidently is not a property that bilinear map based constructions can hope to achieve. Additionally, the NLinFE supports outputs of *super-polynomial* size, while bilinear map based constructions can support only polynomially sized outputs. In particular, this leads to the latter constructions relying on a complicated and inefficient (albeit cool) "security amplification" step in order to be useful for iO. Moreover, there is a qualitative advantage to Agrawal's direct construction: while bilinear map based constructions use clever methods to compute a PRG output *exactly*, the direct construction of NLinFE relaxes correctness and settles for computing the PRG output only *approximately* – this allows for the usage of encodings that are not powerful enough for exact computation.

On the other hand, Agrawal's encodings are new, while assumptions over bilinear maps have stood the test of time (in the pre-quantum world). While bilinear map based constructions must also make new, non-standard assumptions, these constructions come with a clean proof from the non-standard assumptions. Meanwhile, Agrawal's NLinFE came with a proof in a very weak security game that only permits the adversary to request a *single* ciphertext, and that too from a non-standard assumption. Moreover, the author did not suggest concrete parameters for the construction, and admitted the necessity of substantially more cryptanalysis before faith could be placed in these new assumptions.

Our Results. In this work, we undertake the task of thorough cryptanalytic study of Agrawal's NLinFE scheme. We design two attacks that let the adversary completely break the security of the scheme. To achieve this, we develop new cryptanalytic techniques which (we hope) will inform future designs of the primitive of NLinFE.

As mentioned above, Agrawal proved the security of her NLinFE in a weak security game where the attacker is only permitted to request a single ciphertext. Our first attack shows that this is not a co-incidence: an attacker given access to many ciphertexts can manipulate them to recover a (nonlinear) equation in secret terms, which, with some effort, can be solved to recover the secret elements. We emphasize that this attack is very different in nature from the annihilation attacks [42] studied in the context of breaking other constructions of iO. We refer to this attack as the *multiple ciphertext attack*. To demonstrate our attack, we formalize an assumption implicitly made by [2], and design an attack that breaks this assumption – this in turn implies an attack on the scheme. We implement this attack and provide the code as supplementary material with this work.

Our second attack, which we call the *rank attack* exploits a seemingly harmless property of the output of decryption in NLinFE. Recall that the primitive of

NLinFE enables an encryptor to compute a ciphertext $\mathsf{CT}(\mathbf{z})$, a key generator to compute a secret key $\mathsf{SK}(\mathbf{v})$ and the decryptor, given $\mathsf{CT}(\mathbf{z})$ and $\mathsf{SK}(\mathbf{v})$ to recover $\langle \mathbf{z},\ \mathbf{v}\rangle + \mathsf{Nse}$, where Nse must satisfy some weak pseudorandomness properties.

A detail that is important here is that for NLinFE to be useful for iO, the term Nse above must be a linear combination of noise terms, each multiplied with a different (public) modulus. In more detail, the noise term Nse output by NLinFE has the structure $\sum_i p_i \mu_i$ where p_i for $i \in [0, D-2]$ are a sequence of increasing moduli and μ_i are unstructured noise terms. Moreover, for decryption to succeed, these moduli must be public.

The NLinFE construction takes great care to ensure that the noise terms computed via NLinFE are high degree polynomials in values that are spread out over the entire ring, and argues (convincingly, in our opinion) that these may not be exploited easily. However, while some of the μ_i in the above equation are indeed "strong" and difficult to exploit, we observe that some of them are not. Moreover, since the moduli p_i are public, the μ_i can be "separated" into different "levels" according to the factor p_i. Hence, it is necessary that the noise at *each* "level" be "strong", but NLinFE fails to enforce this. Therefore, while there exist strong terms in some levels, the existence of a weak noise term in even one other level enables us to isolate them and use them to construct a matrix, whose rank reveals whether the message bit is 0 or 1.

From the knowledge gained by our cryptanalytic study, we suggest fixes to the scheme. The first attack can be overcome by disabling meaningful manipulation between different encodings. We achieve this by making the encodings non-commutative. The second attack can be overcome by ensuring that the noise terms for all levels are equally strong. We then provide a new scheme which overcomes the vulnerabilities described above. We also provide a thorough analysis of all the security aspects of this scheme and argue why plausible attacks do not work. We additionally provide concrete parameters with which the scheme may be instantiated.

Comparison with other attacks on iO. While Agrawal's NLinFE construction is quite different from previous iO constructions needing fresh cryptanalysis, there are still some high-level similarities between the rank attack we propose and previous attacks on candidate obfuscators [22, 25, 19, 20]. In more detail, these attacks also combine public elements in a clever way to obtain a matrix, and computing the eigenvalues or the rank of this matrix then enables an attacker to break the scheme. We note however that while the main idea of the attack is the same (we compute a matrix and its rank leaks some secret information), the way we obtain the matrix is completely different from [25, 19, 20].

1.1 Our Techniques

We proceed to describe our techniques. We begin by defining the primitive of noisy linear functional encryption.

Noisy Linear Functional Encryption. Noisy linear functional encryption (NLinFE) is a generalization of linear functional encryption (LinFE) [1, 3]. Recall that in linear FE, the encryptor provides a $\mathsf{CT}_{\mathbf{z}}$ which encodes vector $\mathbf{z} \in R^n$, the key generator provides a secret key $\mathsf{SK}_{\mathbf{v}}$ which encodes vector $\mathbf{v} \in R^n$ and the decryptor combines them to recover $\langle \mathbf{z}, \mathbf{v} \rangle$. NLinFE is similar to linear FE, except that the function value is recovered only up to some bounded additive noise term, and indistinguishability holds even if the challenge messages evaluated on any function key are only "approximately" and not exactly equal. The functionality of NLinFE is as follows: given a ciphertext $\mathsf{CT}_{\mathbf{z}}$ and a secret key $\mathsf{SK}_{\mathbf{v}}$, the decryptor recovers $\langle \mathbf{z},\ \mathbf{v} \rangle + \mathsf{noise}_{\mathbf{z},\mathbf{v}}$ where $\mathsf{noise}_{\mathbf{z},\mathbf{v}}$ is specific to the message and function being evaluated.

It is well known that functional encryption (FE) for the function class NC_1 which achieves *sublinear*[3] ciphertext is sufficient to imply iO [6, 15]. Agrawal [2] additionally showed the following "bootstrapping" theorem.

Theorem 1.1 ([2]). *(Informal) There exists an* FE *scheme for the circuit class* NC_1 *with sublinear ciphertext size and satisfying indistinguishability based security, assuming:*

- *A noisy linear FE scheme* NLinFE *with sublinear ciphertext size satisfying indistinguishability based security and supporting superpolynomially large outputs.*
- *The Learning with Errors (*LWE*) Assumption.*
- *A pseudorandom generator (*PRG*) computable in* NC_0.

Since the last two assumptions are widely believed, it suffices to construct an NLinFE scheme to construct the all-powerful iO.

The NLinFE *Construction.* Agrawal provided a direct construction of NLinFE which supports superpolynomially large outputs, based on new assumptions that are based on the Ring Learning With Errors (RLWE) and NTRU assumptions (we refer the reader to Section 2 for a refresher on RLWE and NTRU).

The starting point of Agrawal's NLinFE scheme is the LinFE scheme of [3], which is based on LWE (or RLWE). NLinFE inherits the encodings and secret key structure of LinFE verbatim to compute inner products, and develops new techniques to compute the desired noise. Since the noise must be computed using a high degree polynomial for security [10, 39], the work of [2] designs new encodings that are amenable to multiplication as follows.

Let $R = \mathbb{Z}[x]/\langle x^n + 1 \rangle$ and $R_{p_1} = R/(p_1 \cdot R)$, $R_{p_2} = R/(p_2 \cdot R)$ for some primes $p_1 < p_2$. Then, for $i \in \{1, \ldots, w\}$, sample f_{1i}, f_{2i} and g_1, g_2 from a discrete Gaussian over ring R. Set

$$h_{1i} = \frac{f_{1i}}{g_1}, \quad h_{2j} = \frac{f_{2j}}{g_2} \in R_{p_2} \ \ \forall \ \ i, j \in [w]$$

[3] Here "sublinear" refers to the property that the ciphertext size is sublinear in the number of keys requested by the FE adversary.

Thus, [2] assumes that the samples $\{h_{1i}, h_{2j}\}$ for $i, j \in [w]$ are indistinguishable from random, even though multiple samples share the same denominator.

Additionally, [2] assumes that RLWE with small secrets remains secure if the noise terms live in some secret ideal. The motivation for choosing such structured secrets is that they can be multiplied with well chosen NTRU terms such as the $\{h_{1i}, h_{2j}\}$ above, to cancel the denominator and obtain a small element which can be absorbed in noise.

In more detail, for $i \in [w]$, let $\widehat{\mathcal{D}}(\Lambda_2), \widehat{\mathcal{D}}(\Lambda_1)$ be discrete Gaussian distributions over lattices Λ_2 and Λ_1 respectively. Then, sample

$$\begin{aligned} e_{1i} &\leftarrow \widehat{\mathcal{D}}(\Lambda_2), \quad \text{where } \Lambda_2 \triangleq g_2 \cdot R. \text{ Let } \quad e_{1i} = g_2 \cdot \xi_{1i} \in \mathsf{small}, \\ e_{2i} &\leftarrow \widehat{\mathcal{D}}(\Lambda_1), \quad \text{where } \Lambda_1 \triangleq g_1 \cdot R. \text{ Let } \quad e_{2i} = g_1 \cdot \xi_{2i} \in \mathsf{small}, \end{aligned}$$

Here, small is used to collect terms whose norms may be bounded away from the modulus. Note that for $i, j \in [w]$, it holds that:

$$h_{1i} \cdot e_{2j} = f_{1i} \cdot \xi_{2j}, \quad h_{2j} \cdot e_{1i} = f_{2j} \cdot \xi_{1i} \quad \in \ \mathsf{small}$$

Now, sample small secrets t_1, t_2 and for $i \in [w]$, compute

$$\begin{aligned} d_{1i} &= h_{1i} \cdot t_1 + p_1 \cdot e_{1i} \in R_{p_2} \\ d_{2i} &= h_{2i} \cdot t_2 + p_1 \cdot e_{2i} \in R_{p_2} \end{aligned}$$

Then, note that the products $d_{1i} \cdot d_{2j}$ do not suffer from large cross terms for any $i, j \in [w]$. As discussed above, due to the fact that the error of one sample is chosen to "cancel out" the large denominator in the other sample, the product yields a well behaved RLWE sample whose label is a product of the original labels. In more detail,

$$\begin{aligned} d_{1i} \cdot d_{2j} &= \big(h_{1i} \cdot h_{2j}\ \big) \cdot (t_2\ t_2) + p_1 \cdot \mathsf{noise} \\ \text{where } \mathsf{noise} &= p_1 \cdot \big(f_{1i} \cdot \xi_{2j} \cdot t_1 + f_{2j} \cdot \xi_{1i} \cdot t_2 + p_1 \cdot g_1 \cdot g_2 \cdot \xi_{1i} \cdot \xi_{2j}\big) \in \mathsf{small} \end{aligned}$$

The encoding $d_{1i} \cdot d_{2j}$ can be seen an an RLWE encoding under a public label – this enables the noise term $p_1 \cdot \mathsf{noise}$ above to be added to the inner product computed by LinFE, yielding the desired NLinFE. The actual construction [2] does several more tricks to ensure that the noise term is high entropy and spread across the ring – we refer the reader to Section 3 for details.

Exploiting Correlated Noise across Multiple Ciphertexts. As discussed above, Agrawal [2] provided a proof of security for the NLinFE construction (under a non-standard assumption) in a very weak security model where the adversary is only allowed to request a single ciphertext. In this work, we show that the construction is in fact insecure if the adversary has access to multiple ciphertexts. To do so, we first formally define a variant of the RLWE problem, which we call the RLWE problem with correlated noise. The distribution of the elements in this problem are similar to the one obtained by the encryption procedure of the NLinFE described above. We then show that this problem can be solved in polynomial time by an attacker, which in turn translates to an attack on Agrawal's NLinFE construction.

The key vulnerability exploited by the attack is that the noise terms across multiple ciphertexts are correlated. In more detail, we saw above that $d_{1i} = h_{1i} \cdot t_1 + p_1 \cdot e_{1i}$ where e_{1i} lives in the ideal $g_2 \cdot R$. Now, consider the corresponding element in another ciphertext: $d'_{1i} = h_{1i} \cdot t'_1 + p_1 \cdot e'_{1i}$ where e'_{1i} is also in the ideal $g_2 \cdot R$. The key observation we make is that the noise e_{1i} does not only annihilate the requisite large terms in the encodings of its own ciphertext namely $\{d_{2i}\}$ – it also annihilates large terms in the encodings of other ciphertexts, namely $\{d'_{2i}\}$.

This allows us to perform mix and match attacks, *despite* the fact that each encoding is randomized with fresh randomness. Consider the large terms in the following two products:

$$\begin{aligned} d_{1i}d'_{2j} &= \big(h_{1i}h_{2j}\big) \cdot (t_1 t'_2) + p_1 \cdot \mathsf{small} \\ d_{2j}d'_{1i} &= \big(h_{2j}h_{1i}\big) \cdot (t_2 t'_1) + p_1 \cdot \mathsf{small} \end{aligned}$$

We see above that the labels $h_{1i}h_{2j}$ can be computed in two different ways (but the secrets are different). In a symmetric manner, if we consider other indices i' and j' for the ciphertext elements above, we can obtain

$$\begin{aligned} d_{1i}d_{2j} &= \big(h_{1i}h_{2j}\big) \cdot (t_1 t_2) + p_1 \cdot \mathsf{small} \\ d_{2j'}d_{1i'} &= \big(h_{2j'}h_{1i'}\big) \cdot (t_2 t_1) + p_1 \cdot \mathsf{small}. \end{aligned}$$

Now, the secret is the same but the labels are changing. By playing on these symmetries, we can combine the products above (and the symmetric ones) so that all large terms are canceled and we are left with only small terms.

Intrinsically, what happens here is that in an element $d_{1i} = h_{1i} \cdot t_1 + p_1 \cdot e_{1i}$, we can change the h_{1i} and t_1 elements independently (the secret t_1 changes with the ciphertext and the label h_{1i} changes with the index of the element in the ciphertext). By varying these two elements independently, one can obtain 2×2 encodings (for 2 different choices of h_{1i} and 2 different choices of t_1), and consider the 2×2 matrix associated. More formally, let us write

$$\begin{aligned} d_{1i} &= h_{1i} \cdot t_1 + p_1 \cdot e_{1i} \\ d_{1i'} &= h_{1i'} \cdot t_1 + p_1 \cdot e_{1i'} \\ d'_{1i} &= h_{1i} \cdot t'_1 + p_1 \cdot e'_{1i} \\ d'_{1i'} &= h_{1i'} \cdot t'_1 + p_1 \cdot e'_{1i'} \end{aligned}$$

these encodings. We consider the matrix

$$\begin{pmatrix} d_{1i} & d_{1i'} \\ d'_{1i} & d'_{1i'} \end{pmatrix} = \begin{pmatrix} t_1 \\ t'_1 \end{pmatrix} \cdot \begin{pmatrix} h_{1i} & h_{1i'} \end{pmatrix} + p_1 \cdot \begin{pmatrix} e_{1i} & e_{1i'} \\ e'_{1i} & e'_{1i'} \end{pmatrix}.$$

This matrix is the sum of a matrix of rank 1 with large coefficients plus a full rank matrix with small coefficients that are multiples of g_2. These properties ensure that its determinant will be of the form $g_2/g_1 \cdot \mathsf{small}$. By doing the same thing with the encodings d_{2i}, we can also create an element of the form $g_1/g_2 \cdot \mathsf{small}$. By multiplying these two elements, we finally obtain a linear combination of the encodings which is small. We can then distinguish whether the encodings are random or are RLWE with correlated noise elements. For more details, please see Section 4.

Unravelling the structure of the Noise. Our second attack, the so called "rank attack" exploits the fact that for the NLinFE noise to be useful for bootstrapping, it needs to be linear combination of noise terms, each of which is multiple of a fixed and public modulus p_i, for $i \in [0, D-2]$. As discussed above, the noise terms that are multiples of distinct p_i may be separated from each other and attacked individually. In these piece-wise noise terms, we first isolate the noise term that encodes the message, which is 0 or m (say). Thus, our isolated noise term is of the form Nse or $\mathsf{Nse} + m$ depending on the challenge. Here, Nse is a complicated high degree multivariate polynomial, but we will find a way to learn the challenge bit *without* solving high degree polynomial equations.

To do so, we examine the noise term more carefully. As mentioned above, this term is a high degree, multivariate polynomial which looks difficult to analyze. However, we observe that each variable in this polynomial may be categorized into one of three "colours" – blue if it is fixed across all ciphertexts and secret keys, red if it is dependent only on the secret key and black if it is dependent only on the ciphertext. Next, we observe that if the challenge is 0, then the above polynomial may be expressed as a sum of scalar products, where in every scalar product one vector depends only on the secret key and the other one depends only on the cipher text. Concatenating all these vectors, one obtains a term

$$\langle \mathbf{a}, \mathbf{b} \rangle,$$

where $\mathbf{a}$ depends only on the secret key and $\mathbf{b}$ depends only on the ciphertext (and they are both secret). The dimension of $\mathbf{a}$ and $\mathbf{b}$ is the sum of the dimension of all the vectors involved in the sum above, let us denote this dimension by N.

Assume that we can make $N+1$ requests for secret keys and ciphertexts. Now, in NLinFE, the message m itself depends on *both* the secret key and the ciphertext[4] – we denote by m_{ij} the message corresponding to the i-th secret key and the j-th ciphertext, and note that m_{ij} is known to the NLinFE adversary. We write $c_{i,j} = \langle \mathbf{a}_i, \mathbf{b}_j \rangle + (0 \text{ or } m_{ij})$ the noise term obtained when computing decryption with the i-th secret key and the j-th ciphertext. Define $\mathbf{C}$ and $\mathbf{M}$ the $N \times N$ matrices $(c_{i,j})_{i,j}$ and $(m_{ij})_{i,j}$ respectively. Similarly, let $\mathbf{A}$ be the matrix whose rows are the $\mathbf{a}_i$ and $\mathbf{B}$ be the matrix whose columns are the $\mathbf{b}_j$.

Then, depending on the challenge, we claim that $\mathbf{C}$ or $\mathbf{C} - \mathbf{M}$ is of rank at most N. To see this, note that we have $\mathbf{C} = \mathbf{A} \cdot \mathbf{B} + (0 \text{ or } M)$, where $\mathbf{A}$ has dimension $(N+1) \times N$ and $\mathbf{B}$ has dimension $N \times (N+1)$, so that $\mathbf{A} \cdot \mathbf{B}$ has

[4] This is created by the bootstrapping step. Intuitively m_{ij} is itself a noise term, which depends on both SK and CT, and we seek to "flood" this term using NLinFE. Please see [2] for more details.

rank at most N. On the other hand, the other matrix is of the form $\mathbf{A} \cdot \mathbf{B} \pm M$, which has full rank with good probability. We finish the attack by arguing that the adversary is indeed allowed to make $N + 1$ requests for secret keys and ciphertexts. Thus, by computing the rank of $\mathbf{C}$ and $\mathbf{C} - \mathbf{M}$, we can learn the challenge bit. For details, please see Section 5.

Fixing the Construction. In light of the attacks described above, we propose a variant of Agrawal's NLinFE construction [2], designed to resist these attacks.

Recall that for the multi-ciphertexts attack, we used the commutativity of the elements to ensure that, when multiplying elements in a certain way, the labels and secrets were the same. Hence, we prevent this attack by replacing the product of scalars $h_{1i} \cdot t_1$ in the encodings by an inner product $\langle \mathbf{h}_{1i}\, , \mathbf{t}_1 \rangle$, where the elements h_{1i} and t_1 have been replaced by vectors of dimension κ (the security parameter). This fix does not completely prevent the multi-ciphertexts attack, but the generalization of this attack to this non commutative setting requires a very large modulus, and is therefore not applicable to the range of parameters required for correctness.

To fix the rank attack, we first observe that we do not need to construct directly an NLinFE scheme with structured noise. Indeed, assume first that we have an NLinFE scheme with arbitrary noise, and we would like to have a noise term which is a multiple of p_0. Then, when we want to encode a vector $\mathbf{z}$, we simply encode $\mathbf{z}/p_0$ with our NLinFE with arbitrary noise. By decrypting the message, one would then recover $1/p_0 \cdot \langle \mathbf{z}, \mathbf{v} \rangle + \mathsf{noise}$, and by multiplying this by p_0, we obtain $\langle \mathbf{z}, \mathbf{v} \rangle + p_0 \cdot \mathsf{noise}$, with the desired noise shape. More generally, if we want a noise term which is a sum of multiples of p_i's, we could use an additive secret sharing of $\mathbf{z}$, i.e., compute random vectors $\mathbf{z}_i$ such that $\sum_i \mathbf{z}_i = \mathbf{z}$, and then encode $\mathbf{z}_i/p_i$ with the NLinFE scheme with arbitrary noise. By decrypting every ciphertexts, one could then recover $1/p_i \cdot \langle \mathbf{z}_i, \mathbf{v} \rangle + \mathsf{noise}$ for all i's, and by scaling and summing them, one will have a noise term of the desired shape.

Once we have made this observation that an NLinFE scheme with arbitrary noise is sufficient for our purpose, we can prevent the rank attack by removing the moduli p_i from Agrawal's construction. This means that the noise term we obtain at the end cannot be split anymore into smaller noise terms by looking at the "levels" created by the moduli. We now only have one big noise term, which contains noise terms of high degree and so seems hard to exploit. For technical reasons, we in fact have to keep one modulus, but the general intuition is the same as the one given here. For more details, please see Section 6.

1.2 Organization of the paper

In Section 2, we provide the definitions and preliminaries we require. In Section 3, we recap the NLinFE scheme provided in [2]. In Section 4, we provide our multiple ciphertext attack against the scheme in Section 3. In Section 5, we provide our rank attack. In Section 6, we provide a modified construction which is secure against all the attacks we find. In Section 7 we provide a concrete setting of parameters and discuss why we believe other potential attacks do not apply.

2 Preliminaries

In this section, we provide the notation and preliminaries we require.

2.1 Noisy Linear Functional Encryption (NLinFE)

Let R be a ring, instantiated either as the ring of integers $\mathbb{Z}$ or the ring of polynomials $\mathbb{Z}[x]/f(x)$ where $f(x) = x^n+1$ for n a power of 2. We let $R_{p_i} = R/p_iR$ for some prime p_i, $i \in [0, d]$ for some constant dLet $B_1, B_2 \in \mathbb{R}^+$ be bounding values, where $\frac{B_2}{B_1} = \mathsf{superpoly}(\kappa)$.

We define the symmetric key variant below.

Definition 2.1. *A (B_1, B_2, N)-noisy linear functional encryption scheme FE is a tuple $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ of four probabilistic polynomial-time algorithms with the following specifications:*

- $\mathsf{FE.Setup}(1^\kappa, R^\ell_{p_{d-1}})$ *takes as input the security parameter κ and the space of message and function vectors $R^\ell_{p_{d-1}}$ and outputs the public key and the master secret key pair $(\mathsf{PK}, \mathsf{MSK})$.*
- $\mathsf{FE.Keygen}(\mathsf{MSK}, \mathbf{v})$ *takes as input the master secret key MSK and a vector $\mathbf{v} \in R^\ell_{p_{d-1}}$ and outputs the secret key $\mathsf{SK}_\mathbf{v}$.*
- $\mathsf{FE.Enc}(\mathsf{MSK}, \mathbf{z})$ *takes as input the public key PK and a message $\mathbf{z} \in R^\ell_{p_{d-1}}$ and outputs the ciphertext $\mathsf{CT}_\mathbf{z}$.*
- $\mathsf{FE.Dec}(\mathsf{SK}_\mathbf{v}, \mathsf{CT}_\mathbf{z})$ *takes as input the secret key of a user $\mathsf{SK}_\mathbf{v}$ and the ciphertext $\mathsf{CT}_\mathbf{z}$, and outputs $y \in R_{p_{d-1}} \cup \{\perp\}$.*

Definition 2.2 (Approximate Correctness). *A noisy linear functional encryption scheme FE is correct if for all $\mathbf{v}, \mathbf{z} \in R^\ell_{p_{d-1}}$,*

$$\Pr\left[\begin{array}{l}(\mathsf{PK}, \mathsf{MSK}) \leftarrow \mathsf{FE.Setup}(1^\kappa);\\ \mathsf{FE.Dec}\Big(\mathsf{FE.Keygen}(\mathsf{MSK}, \mathbf{v}), \mathsf{FE.Enc}(\mathsf{MSK}, \mathbf{z})\Big) = \langle \mathbf{v}, \mathbf{z}\rangle + \mathsf{noise}_{\mathsf{fld}}\end{array}\right] = 1 - \mathrm{negl}(\kappa)$$

where $\mathsf{noise}_{\mathsf{fld}} \in R$ with $\|\mathsf{noise}_{\mathsf{fld}}\| \leq B_2$ and the probability is taken over the coins of $\mathsf{FE.Setup}$, $\mathsf{FE.Keygen}$, and $\mathsf{FE.Enc}$.

Security. Next, we define the notion of $\mathsf{Noisy\text{-}IND}$ security and admissible adversary.

Definition 2.3 ($\mathsf{Noisy\text{-}IND}$ Security Game). *We define the security game between the challenger and adversary as follows:*

1. **Public Key:** *Challenger returns PK to the adversary.*
2. **Pre-Challenge Queries:** Adv *may adaptively request keys for any functions $\mathbf{v}_i \in R^\ell_{p_{d-1}}$. In response, Adv is given the corresponding keys $\mathsf{SK}(\mathbf{v}_i)$.*

3. **Challenge Ciphertexts:** Adv *outputs the challenge message pairs* $(\mathbf{z}_0^i, \mathbf{z}_1^i) \in R_{p_{d-1}}^{\ell} \times R_{p_{d-1}}^{\ell}$ *for* $i \in [Q]$*, where* Q *is some polynomial, to the challenger. The challenger chooses a random bit* b*, and returns the ciphertexts* $\{\mathsf{CT}(\mathbf{z}_b^i)\}_{i\in[Q]}$.
4. **Post-Challenge Queries:** Adv *may request additional keys for functions of its choice and is given the corresponding keys.* Adv *may also output additional challenge message pairs which are handled as above.*
5. **Guess.** Adv *outputs a bit* b'*, and succeeds if* $b' = b$.

The advantage *of* Adv *is the absolute value of the difference between the adversary's success probability and* $1/2$.

Definition 2.4 (Admissible Adversary). *We say an adversary is admissible if it makes at most* N *key requests and if for any pair of challenge messages* $\mathbf{z}_0, \mathbf{z}_1 \in R_{p_{d-1}}^{\ell}$ *and any queried key* $\mathbf{v}_i \in R_{p_{d-1}}^{\ell}$*, it holds that* $|\langle \mathbf{v}_i, \mathbf{z}_0 - \mathbf{z}_1 \rangle| \le B_1$
.

Structure of Noise. The bootstrapping step in [2] requires that

$$|\langle \mathbf{v}_i, \mathbf{z}_0 - \mathbf{z}_1 \rangle| = \sum_{i=0}^{d-2} p_i \cdot \mathsf{noise}_{\mathsf{ch,i}}$$

for some noise terms $\mathsf{noise}_{\mathsf{ch,i}}$. Hence the flooding noise $\mathsf{noise}_{\mathsf{fld}}$ that is added by the NLinFE must also be structured as $\sum_{i=0}^{d-2} p_i \cdot \mathsf{noise}_{\mathsf{fld,i}}$.

Definition 2.5 (Noisy-IND security).
A (B_1, B_2, N) *noisy linear FE scheme* NLinFE *is* $\mathsf{Noisy\text{-}IND}$ *secure if for all admissible probabilistic polynomial-time adversaries* Adv*, the advantage of* Adv *in the* $\mathsf{Noisy\text{-}IND}$ *security game is negligible in the security parameter* κ.

The works of [6, 15, **?**, 2] show that as long as the size of the ciphertext is sublinear in N, a $(B_1, B_2, N) - \mathsf{NLinFE}$ scheme implies indistinguishability obfuscation.

2.2 Sampling and Trapdoors

Ajtai [4] showed how to sample a random lattice along with a trapdoor that permits sampling short vectors from that lattice. Recent years have seen significant progress in refining and extending this result [45, 5, 41].

Let $R = \mathbb{Z}[x]/(f)$ where $f = x^n + 1$ and n is a power of 2. Let $R_q \triangleq R/qR$ where q is a large prime satisfying $q = 1 \mod 2n$. For $r \in R$, we use $\|r\|$ to refer to the Euclidean norm of r's coefficient vector.

We will make use of the following algorithms from [41]:

1. $\mathsf{TrapGen}(n, m, q)$: The $\mathsf{TrapGen}$ algorithm takes as input the dimension of the ring n, a sufficiently large integer $m = O(n \log q)$ and the modulus size q and outputs a vector $\mathbf{w} \in R_q^m$ such that the distribution of $\mathbf{w}$ is negligibly far from uniform, along with a "trapdoor" $\mathbf{T_w} \in R^{m\times m}$ for the lattice $\Lambda_q^{\perp}(\mathbf{w}) = \big\{\mathbf{x} \;: \langle \mathbf{w},\, \mathbf{x}\rangle = 0 \mod q\big\}$.

2. $\mathsf{SamplePre}(\mathbf{w}, \mathbf{T_w}, a, \sigma)$: The $\mathsf{SamplePre}$ algorithm takes as input a vector $\mathbf{w} \in R_q^m$ along with a trapdoor $\mathbf{T_w}$ and a syndrome $a \in R_q$ and a sufficiently large $\sigma = O(\sqrt{n \log q})$ and outputs a vector $\mathbf{e}$ from a distribution within negligible distance to $\mathcal{D}_{\Lambda_q^a(\mathbf{w}), \sigma \cdot \omega(\sqrt{\log n})}$ where $\Lambda_q^a(\mathbf{w}) = \{\mathbf{x} \ : \langle \mathbf{w},\ \mathbf{x} \rangle = a \mod q\}$.

2.3 Random matrices over $\mathbf{Z}_q$

Lemma 2.6. *Let q be a prime integer and $\mathbf{A}$ be sampled uniformly in $(\mathbf{Z}/(q\mathbf{Z}))^{m \times m}$. Then*

$$\mathbf{P}\left(\det(\mathbf{A}) \in (\mathbf{Z}/(q\mathbf{Z}))^\times\right) = \prod_{i=1}^{m} \left(1 - \frac{1}{q^i}\right) \geq \frac{4\ln(2)}{q}.$$

Proof. The first equality is obtained by counting the number of invertible $m \times m$ matrices in $\mathbf{Z}/(q\mathbf{Z})$. For the lower bound, we observe that $1 - 1/q^i \geq 1/2$ for all $1 \leq i \leq m$. By concavity of the logarithm function, this implies that $\log(1 - 1/q^i) \geq -2/q^i$ for all $i \geq 1$ (recall that the logarithm is taken in base 2). We then have

$$\begin{aligned} \log \prod_{i=1}^{m} \left(1 - \frac{1}{q^i}\right) &= \sum_{i=1}^{m} \log\left(1 - \frac{1}{q^i}\right) \\ &\geq \sum_{i=1}^{m} \frac{-2}{q^i} \\ &\geq \frac{-2}{q} \cdot \frac{1}{1 - 1/q} \\ &\geq \frac{-4}{q}. \end{aligned}$$

Taking the exponential we obtain that $\mathbf{P}\left(\det(\mathbf{A}) \in (\mathbf{Z}/(q\mathbf{Z}))^\times\right) \geq 2^{-4/q} \geq 1 - \frac{4\ln(2)}{q}$ as desired.

Lemma 2.7 (Corollary 2.2 of [16]). *Let q be a prime integer and A be sampled uniformly in $(\mathbf{Z}/(q\mathbf{Z}))^{m \times m}$. For any $x \in (\mathbf{Z}/(q\mathbf{Z}))^\times$, we have*

$$\mathbf{P}\left(\det(\mathbf{A}) = x \mid \det(A) \in (\mathbf{Z}/(q\mathbf{Z}))^\times\right) = \frac{1}{|(\mathbf{Z}/(q\mathbf{Z}))^\times|} = \frac{1}{q-1}.$$

In other words, $\det(A)$ is uniform in $(\mathbf{Z}/(q\mathbf{Z}))^\times$ when conditioned on being invertible.

Corollary 2.2 of [16] even gives explicit values for the probability $\mathbf{P}(\det(\mathbf{A}) = x)$ for any x. Here, we only use the fact that these values are the same whenever the gcd of x and q is constant (in our case, the gcd is always 1 because x is invertible). Observe also that Corollary 2.2 of [16] is stated for a prime power q, and can be extended to any modulus q by Chinese reminder theorem (but we only use it here in the case of a prime modulus q).

3 Agrawal's Construction of Noisy Linear FE

We begin by recapping the construction of NLinFE by Agrawal [2]. The construction uses two prime moduli p_1 and p_2 with $p_1 \ll p_2$. The message and function vectors will be chosen from R_{p_1} while the public key and ciphertext are from R_{p_2}. The construction will make use of the fact that elements in R_{p_1} as well as elements sampled from a discrete Gaussian distribution denoted by $\mathcal{D}$, are small in R_{p_2}.

$\mathsf{NLinFE.Setup}(1^\kappa, 1^w)$: On input a security parameter κ, a parameter w denoting the length of the function and message vectors, do the following:

1. Sample prime moduli $p_0 < p_1 < p_2$ and standard deviation σ for discrete Gaussian distributions $\mathcal{D}$, $\hat{\mathcal{D}}$ and $\hat{\mathcal{D}}'$ according to the parameter specification of [2].
2. Sample $\mathbf{w} \leftarrow R_{p_2}^m$ with a trapdoor $\mathbf{T_w}$ using the algorithm TrapGen as defined in Section 2.2.
3. Sample $\mathbf{E} \in \mathcal{D}^{m\times w}$ and set $\mathbf{a} = \mathbf{E}^\intercal \mathbf{w} \in R_{p_2}^w$.
4. For $i \in \{1, \ldots, r\}$, $\ell \in \{1, \ldots, k\}$, sample $f_{1i}^\ell, f_{2i}^\ell \leftarrow \mathcal{D}$ and $g_1^\ell, g_2^\ell \leftarrow \mathcal{D}$. If g_1^ℓ, g_2^ℓ are not invertible over R_{p_2}, resample. Set
$$h_{1i}^\ell = \frac{f_{1i}^\ell}{g_1^\ell}, \quad h_{2i}^\ell = \frac{f_{2i}^\ell}{g_2^\ell} \in R_{p_2}$$
5. Sample a PRF seed, denoted as seed.

Output
$$\mathsf{MSK} = \Big(\mathbf{w}, \mathbf{T_w}, \mathbf{a}, \mathbf{E}, \{f_{1i}^\ell, f_{2i}^\ell\}_{i\in[r],\ell\in[k]}, \{g_1^\ell, g_2^\ell\}_{\ell\in[k]}\}, \mathsf{seed} \Big)$$

$\mathsf{NLinFE.Enc}(\mathsf{MSK}, \mathbf{z})$: On input public key MSK, a message vector $\mathbf{z} \in R_{p_1}^w$, do:

1. **Construct Message Encodings.** Sample $\boldsymbol{\nu} \leftarrow \mathcal{D}^m$, $\boldsymbol{\eta} \leftarrow \mathcal{D}^w$ and $t_1, t_2 \leftarrow \mathcal{D}$. Set $s = t_1 \cdot t_2$. Compute:
$$\mathbf{c} = \mathbf{w} \cdot s + p_1 \cdot \boldsymbol{\nu} \in R_{p_2}^m, \quad \mathbf{b} = \mathbf{a} \cdot s + p_1 \cdot \boldsymbol{\eta} + \mathbf{z} \in R_{p_2}^w$$
2. **Sample Structured Noise.** To compute encodings of noise, do the following:
   (a) Define lattices:
   $$\Lambda_1^\ell \stackrel{\text{def}}{=} g_1^\ell \cdot R, \quad \Lambda_2^\ell \stackrel{\text{def}}{=} g_2^\ell \cdot R$$
   (b) Sample noise terms from the above lattices as:
   $$e_{1i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_2^\ell), \tilde{e}_{1i}^\ell \leftarrow \widehat{\mathcal{D}'}(\Lambda_2^\ell), \quad e_{2i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_1^\ell), \tilde{e}_{2i}^\ell \leftarrow \widehat{\mathcal{D}'}(\Lambda_1^\ell) \quad \forall i \in [r], \ell \in [k]$$
   Here $\widehat{\mathcal{D}}(\Lambda_1^\ell), \widehat{\mathcal{D}'}(\Lambda_1^\ell)$ are discrete Gaussian distributions on Λ_1^ℓ and $\widehat{\mathcal{D}}(\Lambda_2^\ell), \widehat{\mathcal{D}'}(\Lambda_2^\ell)$ are discrete Gaussian distributions on Λ_2^ℓ.
3. **Compute Encodings of Noise.**

(a) Let

$$d^{\ell}_{1i} = h^{\ell}_{1i} \cdot t_1 + p_1 \cdot \tilde{e}^{\ell}_{1i} + p_0 \cdot e^{\ell}_{1i} \in R_{p_2} \quad \forall i \in [r], \ell \in [k].$$

Here, $p_1 \cdot \tilde{e}^{\ell}_{1i}$ behaves as noise and $p_0 \cdot e^{\ell}_{1i}$ behaves as the message. Let $\mathbf{d}^{\ell}_1 = (d^{\ell}_{1i})$.

(b) Similarly, let

$$d^{\ell}_{2i} = h^{\ell}_{2i} \cdot t_2 + p_1 \cdot \tilde{e}^{\ell}_{2i} + p_0 \cdot e^{\ell}_{2i} \in R_{p_2} \quad \forall i \in [r], \ell \in [k].$$

Here, $p_1 \cdot \tilde{e}^{\ell}_{2i}$ behaves as noise and $p_0 \cdot e^{\ell}_{2i}$ behaves as the message. Let $\mathbf{d}^{\ell}_2 = (d^{\ell}_{2i})$.

4. **Output Ciphertext.** Output message encodings $(\mathbf{c}, \mathbf{b})$ and noise encodings $(\mathbf{d}^{\ell}_1, \mathbf{d}^{\ell}_2)$ for $\ell \in [k]$.

$\mathsf{NLinFE.KeyGen}(\mathsf{MSK}, \mathbf{v}, \mathbf{v}^{\times})$: On input the master secret key MSK, a NLinFE function vector $\mathbf{v} \in R^{w}_{p_1}$ and its corresponding noise polynomial (represented here as a quadratic polynomial) $\mathbf{v}^{\times} \in R^{L}_{p_1}$, where $L = |1 \leq j \leq i \leq r|$, do the following.

1. **Sampling Basis Preimage vectors.**

(a) Sample short $\mathbf{e}_{ij} \in R^m$ using $\mathsf{SamplePre}$ (please see Section 2.2) with randomness $\mathsf{PRF}(\mathsf{seed}, ij)$ such that

$$\langle \mathbf{w}, \mathbf{e}_{ij} \rangle = h_{ij}, \text{ where } h_{ij} \overset{\text{def}}{=} \sum_{\ell \in [k]} h^{\ell}_{1i} h^{\ell}_{2j} + p_0 \cdot \mathit{\Delta}_{ij} + p_1 \cdot \tilde{\mathit{\Delta}}_{ij}$$

Above $\mathit{\Delta}_{ij}, \tilde{\mathit{\Delta}}_{ij} \leftarrow \mathcal{D} \in R$ for $1 \leq j \leq i \leq r$.

$$\text{Let } \mathbf{E}^{\times} = (\mathbf{e}_{ij}) \in R^{m \times L}, \quad \mathbf{h}^{\times} = (h_{ij}) \in R^{L}_{p_2}$$

where $L = |1 \leq j \leq i \leq r|$.

2. **Combining Basis Preimages to Functional Preimage.** Define

$$\mathbf{k_v} = \mathbf{E} \cdot \mathbf{v} + \mathbf{E}^{\times} \cdot \mathbf{v}^{\times} \quad \in R^m \tag{3.1}$$

3. Output $(\mathbf{k_v}, \mathbf{v})$.

$\mathsf{NLinFE.Dec}(\mathsf{CT_z}, \mathsf{SK_v})$: On input a ciphertext $\mathsf{CT_z} = \big(\mathbf{c}, \mathbf{b}, \{\mathbf{d}^{\ell}_1, \mathbf{d}^{\ell}_2\}_{\ell \in [k]} \big)$ and a secret key $\mathbf{k_v}$ for function $\mathbf{v}$, do the following

1. Compute encoding of noise term on the fly as:

$$\mathbf{d}^{\times} \overset{\text{def}}{=} (\sum_{\ell \in [k]} \mathbf{d}^{\ell}_1 \otimes \mathbf{d}^{\ell}_2) \in R^{L}_{p_2}$$

2. Compute functional ciphertext as:

$$b_{\mathbf{v}} = \mathbf{v}^{\mathsf{T}} \mathbf{b} + (\mathbf{v}^{\times})^{\mathsf{T}} \mathbf{d}^{\times} \in R_{p_2}$$

3. Compute $b_{\mathbf{v}} - \mathbf{k}^{\mathsf{T}}_{\mathbf{v}} \mathbf{c} \mod p_1$ and output it.

Remark on the parameters. In the above scheme, one should think of B_1 as being $\text{poly}(\kappa)$, $B_2 = 2^{O(\kappa)} \cdot B_1$ and $N = (kr)^{1+\varepsilon}$ for some $\varepsilon > 0$.

4 Multi-Ciphertext Attack on Agrawal's NLinFE

Agrawal [2] provided a proof of security for her construction (under a non-standard assumption) in a weak security game where the adversary may only request a single ciphertext. In this section, we show that her construction is in fact insecure if the adversary has access to multiple ciphertexts.

The problem appearing in Agrawal's NLinFE construction is a variant of the RLWE problem, where the random elements in RLWE samples are chosen from some NTRU-like distribution, are kept secret, and the noise terms are correlated to these elements. In this section, we first formally define a variant of the RLWE problem, which we call the RLWE problem with correlated noise. The distribution of the elements in this problem are similar to the one obtained by the encryption procedure of the NLinFE described above. We then show that this problem can be solved in polynomial time by an attacker, hence resulting in an attack on Agrawal's NLinFE construction.

Definition 4.1. *(RLWE with correlated noise). Let R be some ring isomorphic to $\mathbf{Z}^n$ (for instance $R = \mathbf{Z}[X]/(X^n+1)$ for n a power of two, and the isomorphism is the coefficient embedding). We define the RLWE problem with correlated noise as follows. Let m, k, q, σ, σ' be some parameters (q will be the modulus, m the number of samples and σ and σ' are small compared to q). We let $\mathcal{D}_\sigma$ be the discrete Gaussian distribution over R with parameter σ and $U(R_q)$ be the uniform distribution over R_q. Sample*

- $g_1, g_2 \leftarrow \mathcal{D}_\sigma$
- $f_{1i}, f_{2i} \leftarrow \mathcal{D}_\sigma$ *for all* $1 \leq i \leq k$
- $t_1[j], t_2[j] \leftarrow \mathcal{D}_\sigma$ *for all* $1 \leq j \leq m$
- $e_{1i}[j], e_{2i}[j] \leftarrow \mathcal{D}_{\sigma'}$ *for all* $1 \leq i \leq k$ *and* $1 \leq j \leq m$
- $u_{1i}[j], u_{2i}[j] \leftarrow U(R_q)$ *for all* $1 \leq i \leq k$ *and* $1 \leq j \leq m$.

The RLWE problem with correlated noise is to distinguish between

$$\left(\frac{f_{1i}}{g_1} t_1[j] + e_{1i}[j] \cdot g_2 \bmod q, \frac{f_{2i}}{g_2} t_2[j] + e_{2i}[j] \cdot g_1 \bmod q\right)_{i,j}$$

and

$$(u_{1i}[j], u_{2i}[j])_{i,j}.$$

Remark 4.2. This RLWE problem with correlated noise differs from the classical RLWE problem in 4 different ways:

- Instead of being uniform, the elements a are of the form $\frac{f_i}{g} \bmod q$ with f_i and g small modulo q,
- There are multiple secrets $t_1[j]$ and $t_2[j]$,

- The elements $\frac{f_i}{g}$ are secret,
- The noise is correlated with the elements $\frac{f_i}{g}$ (instead of following a small Gaussian distribution).

We observe that if we obtain m ciphertexts from the NLinFE construction described above, and if we only keep in each ciphertext the part corresponding to $\ell = 1$, then the elements obtained follow the RLWE distribution with correlated noise. The notation $[j]$ refers to the j-th ciphertext, and we dropped the index ℓ since we are only considering $\ell = 1$.

The next lemma explains how we can solve the RLWE problem with correlated noise in polynomial time, using 4 pairs of elements (obtained by varying i and j).

Lemma 4.3. *Assume $k, m \geq 2$ and that the modulus q is a prime integer congruent to 1 modulo $2n$. Let $(b_{1i}[j], b_{2i}[j])_{1\leq i,j\leq 2}$ be obtained from either the RLWE distribution with correlated noise or the uniform distribution over R_q. Let us define*

$$\begin{aligned} b :=&(b_{1,1}[1] \cdot b_{2,1}[1] \cdot b_{1,2}[2] \cdot b_{2,2}[2] + b_{1,1}[2] \cdot b_{2,1}[2] \cdot b_{1,2}[1] \cdot b_{2,2}[1] \\ &- b_{1,1}[2] \cdot b_{2,1}[1] \cdot b_{1,2}[1] \cdot b_{2,2}[2] - b_{1,1}[1] \cdot b_{2,1}[2] \cdot b_{1,2}[2] \cdot b_{2,2}[1]) \bmod q. \end{aligned}$$

If the $b_{\beta i}[j]$ come from the uniform distribution, then $\|b\|_\infty \geq q/4$ with high probability (over the random choice of the $(b_{1i}[j], b_{2i}[j])_{1\leq i,j\leq 2}$). Otherwise, $\|b\|_\infty$ is small compared to q.

Proof. Let us first consider the case where the $b_{\beta i}[j]$ are uniform modulo q and independent. Observe that b can be written as the determinant of a product of two matrices

$$\mathbf{M}_1 = \begin{pmatrix} b_{1,1}[1] & b_{1,1}[2] \\ b_{1,2}[1] & b_{1,2}[2] \end{pmatrix} \text{ and } \mathbf{M}_2 = \begin{pmatrix} b_{2,1}[1] & b_{2,1}[2] \\ b_{2,2}[1] & b_{2,2}[2] \end{pmatrix}.$$

These two matrices are uniform over R_q. Because $q \equiv 1 \bmod 2n$, we have that $x^n + 1 = \prod_i (x - \alpha_i) \bmod q$ and so $R_q \simeq \mathbf{Z}_q[x]/(x - \alpha_1) \times \cdots \times \mathbf{Z}_q[x]/(x - \alpha_1) \simeq (\mathbf{Z}_q)^n$. By Chinese reminder theorem, all the matrices $\mathbf{M}_b \bmod (x - \alpha_i)$ are uniform and independent matrices in $\mathbf{Z}_q$. Now, by Chinese reminder theorem and Lemma 2.6, we have that

$$\mathbf{P}(\det(\mathbf{M}_1) \notin R_q^\times) = \mathbf{P}(\exists i, \det(\mathbf{M}_1 \bmod (x - \alpha_i)) \notin \mathbf{Z}_q^\times) \leq O\left(\frac{n}{q}\right).$$

Because $n \ll q$, this implies that $\mathbf{M}_1$ and $\mathbf{M}_2$ are invertible with high probability. Recall from Lemma 2.7 that, when conditioned on being invertible, the determinant of $\mathbf{M}_1$ and $\mathbf{M}_2$ are uniformly distributed over the invertible elements of R_q. Hence, we conclude that with high probability, the product $\det(\mathbf{M}_1) \cdot \det(\mathbf{M}_2)$ is uniform in $R_q^\times$ and so is likely to have infinity norm larger than $q/4$.

Let us now assume that the $b_{\beta i}[j]$ come from the RLWE distribution with correlated noise. Then, we have

$$\begin{aligned}
b = {} & \\
& \left(\frac{f_{1,1}}{g_1}t_1[1] + e_{1,1}[1]\cdot g_2\right)\left(\frac{f_{2,1}}{g_2}t_2[1] + e_{2,1}[1]\cdot g_1\right)\left(\frac{f_{1,2}}{g_1}t_1[2] + e_{1,2}[2]\cdot g_2\right)\left(\frac{f_{2,2}}{g_2}t_2[2] + e_{2,2}[2]\cdot g_1\right) \\
& + \left(\frac{f_{1,1}}{g_1}t_1[2] + e_{1,1}[2]\cdot g_2\right)\left(\frac{f_{2,1}}{g_2}t_2[2] + e_{2,1}[2]\cdot g_1\right)\left(\frac{f_{1,2}}{g_1}t_1[1] + e_{1,2}[1]\cdot g_2\right)\left(\frac{f_{2,2}}{g_2}t_2[1] + e_{2,2}[1]\cdot g_1\right) \\
& - \left(\frac{f_{1,1}}{g_1}t_1[2] + e_{1,1}[2]\cdot g_2\right)\left(\frac{f_{2,1}}{g_2}t_2[1] + e_{2,1}[1]\cdot g_1\right)\left(\frac{f_{1,2}}{g_1}t_1[1] + e_{1,2}[1]\cdot g_2\right)\left(\frac{f_{2,2}}{g_2}t_2[2] + e_{2,2}[2]\cdot g_1\right) \\
& - \left(\frac{f_{1,1}}{g_1}t_1[1] + e_{1,1}[1]\cdot g_2\right)\left(\frac{f_{2,1}}{g_2}t_2[2] + e_{2,1}[2]\cdot g_1\right)\left(\frac{f_{1,2}}{g_1}t_1[2] + e_{1,2}[2]\cdot g_2\right)\left(\frac{f_{2,2}}{g_2}t_2[1] + e_{2,2}[1]\cdot g_1\right),
\end{aligned}$$

where the computations are performed modulo q. Observe that in the products and sums above, all the elements are small. The only things that can be large are the division modulo q by g_1 and g_2. We are going to show that if we develop the products above, then all the terms containing divisions by g_1 or g_2 are annihilated. So b will be a polynomial of degree 4 of small elements (with no denominator) and hence it will be small compared to q.

Let us consider the first line of the equation above

$$\left(\frac{f_{1,1}}{g_1}t_1[1] + e_{1,1}[1]\cdot g_2\right)\cdot\left(\frac{f_{2,1}}{g_2}t_2[1] + e_{2,1}[1]\cdot g_1\right)\cdot\left(\frac{f_{1,2}}{g_1}t_1[2] + e_{1,2}[2]\cdot g_2\right)\cdot\left(\frac{f_{2,2}}{g_2}t_2[2] + e_{2,2}[2]\cdot g_1\right).$$

When we develop this product, we are going to produce terms with denominators of degree 0, 1, 2, 3 and 4 in the g_β. Observe that the third line is the same as the first line, where we have exchanged $t_1[1]$ and $t_1[2]$ and the corresponding noises. So every term of the first line containing $\frac{f_{1,1}}{g_1}t_1[1]\cdot\frac{f_{1,2}}{g_1}t_1[2]$ will be the same as the analogue term in the third line, and so will be annihilated. Similarly, the fourth line is the same as the first line, where we have exchanged $t_2[1]$ and $t_2[2]$ and the corresponding noises. So every term of the first line containing $\frac{f_{2,1}}{g_2}t_2[1]\cdot\frac{f_{2,2}}{g_2}t_2[2]$ will be the same as the analogue term in the fourth line, and so will be annihilated. Using this remark, we argue below that all the terms with denominators in the first line are annihilated.

- The term of degree 4 contains $\frac{f_{1,1}}{g_1}t_1[1]\cdot\frac{f_{1,2}}{g_1}t_1[2]$ and so is annihilated by the third line.
- The terms of degree 3 have to contain 3 denominators out of the 4. So they contain either $\frac{f_{1,1}}{g_1}t_1[1]\cdot\frac{f_{1,2}}{g_1}t_1[2]$ or $\frac{f_{2,1}}{g_2}t_2[1]\cdot\frac{f_{2,2}}{g_2}t_2[2]$. In both cases, they are annihilated.
- The terms of degree 2 containing either $\frac{f_{1,1}}{g_1}t_1[1]\cdot\frac{f_{1,2}}{g_1}t_1[2]$ or $\frac{f_{2,1}}{g_2}t_2[1]\cdot\frac{f_{2,2}}{g_2}t_2[2]$ are annihilated. It remains the terms of degree 2 whose denominator is g_1g_2. But these terms are multiplied by a noise which is a multiple of g_1 and another noise which is a multiple of g_2. Hence the denominator is annihilated and these terms are just polynomials in the small elements.

– The terms of degree 1 have denominator g_1 or g_2. But they are multiplied by noises that are multiples of g_1 and g_2. Hence the denominator is annihilated and these terms are polynomials in the small elements.

To conclude, all the terms are either eliminated thanks to the symmetries, or the denominators are removed by multiplication by g_1 and g_2. Similarly, we can show that this holds for all the four lines. The sage code for the above attack is provided as supplementary material with the paper. So b is a polynomial of constant degree in the g_β, $f_{\beta i}$, $t_\beta[j]$ and $e_{\beta i}[j]$, which are all much smaller than q. Hence, b is also much smaller than q.

Concluding the attack. To conclude the attack on Agrawal's NLinFE scheme, let us now explain how the distinguishing attack described above can be used to recover the secret elements of the RLWE with correlated noise instance. We have seen in Lemma 4.3 that, from four instances of RLWE with correlated noise, one can compute a quantity b which is significantly smaller than the modulus q. This means that one can recover b over the ring R, without reduction modulo q. Let us consider such an element b, obtained from the four RLWE with correlated noise instances $(b_{1i}[j], b_{2i}[j]), (b_{1i'}[j], b_{2i'}[j]), (b_{1i}[j'], b_{2i}[j']), (b_{1i'}[j'], b_{2i'}[j'])$ (for simplicity, the lemma above is stated with $i, j = 1$ and $i', j' = 2$, but it can be generalized to any choice of (i, j, i', j'), with $i \neq i'$ and $j \neq j'$). Computing b as in Lemma 4.3, we obtain a polynomial over R of degree 8 in 16 variables (the g_β, the $t[j]$'s, the $f_{\beta,i}$ and the $e_{\beta,i}[j]$). More generally, if we consider all the equations one can create by computing b as above for i, j, i', j' varying in $\{1, \cdots, \ell\}$, with $i \neq i'$ and $j \neq j'$, then one can obtain $\ell^2(\ell-1)^2$ equations of degree 8 in $2 + 3\ell + 2\ell^2$ variables. Choosing $\ell = 3$ provides 36 equations in 29 variables, hence one may hope that this system has a unique solution, and that solving it would reveal the values of the secret parameters.

Recall that solving a system of polynomial equations is hard in general, but the hardness increases with the number of variables. Hence, if the number of variable is constant (here equal to 29), solving a polynomial system of equations should be easy. One way to solve such a system is by computing a Gröbner basis of the ideal generated by the multivariate polynomials. This can be done in the worst case in time doubly exponential in the number of variables (see for instance [40, 12]), which is constant in our case, as we have a constant number of variables.[5] Once we have a Gröbner basis corresponding to our system of equations, we can solve it by computing the roots of a constant number of univariate polynomials over K. Since we know that the solution of our system is in R^{29}, it is sufficient to compute the roots of the polynomials over K with precision $1/2$, and then round them to the nearest integer element. Solving these univariate polynomial equations can hence be done in polynomial time (in the size of the output).

[5] In all this discussion, we are interested in the theoretical complexity. In practice, solving an arbitrary overdetermined system with 29 variables could take a lot of time, but this time would not increase with the security parameter κ, hence, it is constant for our purposes.

Alternatively, to avoid numerical issues, we could choose a large prime number p, which we know is larger than all the noise terms arising in the equations, and then solve the system in $R/(pR)$. Hopefully, the system is still overdetermined modulo p, and so has a unique solution which corresponds to the solution over R. Thanks to the fact that p is larger than the noise terms, recovering them modulo p reveals them exactly, so we can recover the solution over R from the one over $R/(pR)$. This approach can also be done in time doubly exponential in the number of variables, and polynomial in the degree of K and in $\log(p)$.

To conclude, the elements b enables us to recover equations of degree 8 in a constant number of variables, which can then be solved efficiently. This means that we can recover the secret elements g_β, $t[j]$, $f_{\beta,i}$ and $e_{\beta,i}[j]$ of the RLWE with correlated noise instances in polynomial time (given sufficiently many instances).

5 Rank Attack on Agrawal's NLinFE

In this section, we present a novel "rank attack" against the NLinFE scheme. The attack exploits the property that the NLinFE scheme must compute a noise term with special structure: in detail, the noise term must be expressible as a linear combination of noise terms which are multiples of moduli p_i for $i \in [0, D-2]$. The moduli p_i in this case are *public* – this enables the attacker to recover noise terms at different "levels", namely, corresponding to different moduli. The attack exploits the fact that while the noise terms corresponding to some moduli are highly non-linear and difficult to exploit, those corresponding to some others are in fact linear and may be exploited by carefully arranging them into a matrix and computing its rank. We provide details below.

5.1 Exploiting the noise obtained after decrypting a message.

Let us first explicit the noise obtained after decryption. When computing $b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\intercal}\mathbf{c}$ for a valid ciphertext and secret key, one obtain something much smaller than p_2, which can hence be recovered exactly. This noise is the following

$$\begin{aligned}
&b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\intercal}\mathbf{c} \\
&= \mathbf{v}^T\mathbf{z} + p_1\mathbf{v}^T\boldsymbol{\eta} - p_0(\mathbf{v}^{\times})^T\boldsymbol{\Delta}\cdot s - p_1(\mathbf{v}^{\times})^T\widetilde{\boldsymbol{\Delta}}\cdot s - p_1(\mathbf{v}^T\mathbf{E} + (\mathbf{v}^{\times})^T\mathbf{E}^{\times})\boldsymbol{\nu} \\
&+ \sum_{\ell,i,j} v_{ij}^{\times}\ \Big[p_1\cdot\Big(p_1\cdot(g_2^\ell\cdot\tilde{\xi}_{1i}^\ell\cdot g_1^\ell\cdot\tilde{\xi}_{2j}^\ell) + p_0\cdot(g_2^\ell\cdot\tilde{\xi}_{1i}^\ell\cdot g_1^\ell\cdot\xi_{2j}^\ell + g_2^\ell\cdot\xi_{1i}^\ell\cdot g_1^\ell\cdot\tilde{\xi}_{2j}^\ell) \\
&+ (f_{1i}^\ell\cdot t_1\cdot\tilde{\xi}_{2j}^\ell + f_{2j}^\ell\cdot t_2\cdot\tilde{\xi}_{1i}^\ell)\Big) + p_0\cdot\Big(p_0\cdot(g_2^\ell\cdot\xi_{1i}^\ell\cdot g_1^\ell\cdot\xi_{2j}^\ell) + (f_{1i}^\ell\cdot t_1\cdot\xi_{2j}^\ell + \cdot f_{2j}^\ell\cdot t_2\cdot\xi_{1i}^\ell)\Big)\Big],
\end{aligned}$$

where $\boldsymbol{\Delta}$ and $\widetilde{\boldsymbol{\Delta}}$ are vectors of dimension L whose elements are respectively the Δ_{ij} and $\widetilde{\Delta}_{ij}$. This noise term is quite complicated, but since it involves multiples of p_1 and p_0, one can distinguish the noise terms that are multiples of p_0, p_0^2, p_1, p_1^2 and p_0p_1. Here, we assume that the noise terms that are multiplied to the p_i's are small enough so that the different multiples do not overlap. While this should be true for correctness that p_1 is much larger than the multiples of p_0 appearing in the term above, this might not be true for instance when splitting the multiple of p_0 from the multiple of p_0^2 (one could for instance think of $p_0 = 4$). As we should see below however, this will not be a problem for our attack. To see this, let us write $p_0 \cdot \mathsf{small}_1 + p_0^2 \cdot \mathsf{small}_2 + p_1 \cdot \mathsf{small}_3$ the noise term above. As we have said, for correctness, it should hold that, when reducing this term modulo p_1, we obtain $p_0 \cdot \mathsf{small}_1 + p_0^2 \cdot \mathsf{small}_2$ over R. Now, dividing by p_0 and reducing the obtained term modulo p_0, we recover $\mathsf{small}_1 \bmod p_0$. In the rank attack below, we exploit the noise term small_1, which we might know only modulo p_0 (and not over R). However, all we do on this noise terms is linear algebra, and does not depend on the ring in which we are considering the elements. Hence, we could as well perform the attack in R_{p_0} if we recovered only $\mathsf{small}_1 \bmod p_0$.

Recall also that in the distinguishing game, the adversary chooses two messages $\mathbf{z}_0$ and $\mathbf{z}_1$ with the constraint that $\mathbf{v}^T\mathbf{z}_0 = \mathbf{v}^T\mathbf{z}_1 + p_0 \cdot \mu$ for any vector $\mathbf{v}$ for which she has a secret key (with a small μ). She then gets back the encryption of one of the two messages and wants to know which one was encoded. In other words, if $\mathbf{z}$ is the encrypted message, the adversary knows that $\mathbf{v}^T\mathbf{z} = x$ or $x + p_0 \cdot \mu$ for some known values of x and μ (with $p_0 \cdot \mu$ smaller than some bound B_1), and wants to distinguish between these two cases. We can then assume that the adversary removes x from the noise term, and is left with either 0 or $p_0 \cdot \mu$. The adversary can then obtain the following noise terms

$$\sum_{\ell} \left(\sum_{i,j} v_{ij}^{\times} \cdot \tilde{\xi}_{1i}^{\ell} \tilde{\xi}_{2j}^{\ell} \right) g_2^{\ell} g_1^{\ell} \tag{5.1}$$

$$\sum_{\ell} \left(\sum_{i,j} v_{ij}^{\times} \cdot \xi_{1i}^{\ell} \xi_{2j}^{\ell} \right) g_2^{\ell} g_1^{\ell} \tag{5.2}$$

$$\sum_{\ell} \left(\sum_{i,j} v_{ij}^{\times} \cdot (\tilde{\xi}_{1i}^{\ell} \xi_{2j}^{\ell} + \tilde{\xi}_{1i}^{\ell} \xi_{2j}^{\ell}) \right) g_2^{\ell} g_1^{\ell} \tag{5.3}$$

$$\sum_{i,j,\ell} v_{ij}^{\times} \cdot (f_{1i}^{\ell} \cdot t_1 \cdot \xi_{2j}^{\ell} + f_{2j}^{\ell} \cdot t_2 \cdot \xi_{1i}^{\ell}) \quad + \quad (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu) \tag{5.4}$$

$$\sum_{i,j,\ell} v_{ij}^{\times} \cdot \left(f_{1i}^{\ell} \cdot t_1 \cdot \tilde{\xi}_{2j}^{\ell} + f_{2j}^{\ell} \cdot t_2 \cdot \tilde{\xi}_{1i}^{\ell} \right) \quad + \quad (\mathbf{v}^{\times})^T \widetilde{\boldsymbol{\Delta}} \cdot s + \mathbf{v}^T \boldsymbol{\eta} + (\mathbf{v}^T \mathbf{E} + (\mathbf{v}^{\times})^T \mathbf{E}^{\times}) \boldsymbol{\nu} \tag{5.5}$$

In the noise terms above, the blue elements are secret and are fixed for all ciphertexts and secret keys, the red elements are known and depend only on the secret key, the black elements are secret and depend only on the ciphertexts and the brown element is the challenge. The value μ of the challenge can be chosen by the adversary, and the adversary has to decide, given the above noise terms, whether (5.4) contains 0 or μ. Recall also that the vector $\mathbf{v}$ can be chosen by the adversary whereas the vector $\mathbf{v}^{\times}$ is chosen by the challenger as the polynomial that computes a PRG. The blue and red elements above can be modified independently, by considering another secret key or another ciphertext.

5.2 Rank Attack to Distinguish bit.

The rank attack focuses on the noise term (5.4). As this noise term contains the challenge, it suffices to distinguish between a noise term with 0 or a noise term with μ to break the NLinFE construction. Let us rewrite the equation in a more convenient way.

$$\begin{aligned}
&\sum_{i,j,\ell} v_{ij}^{\times} \cdot \left(f_{1i}^{\ell} \cdot t_1 \cdot \xi_{2j}^{\ell} + f_{2j}^{\ell} \cdot t_2 \cdot \xi_{1i}^{\ell}\right) \quad + \quad (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu) \\
&\quad = \sum_{\ell} \Big(\sum_{j} \Big(\sum_{i} v_{ij}^{\times} \cdot f_{1i}^{\ell} \Big) \cdot \xi_{2j}^{\ell} \cdot t_1 \Big) \; + \; \sum_{\ell} \Big(\sum_{j} \Big(\sum_{i} v_{ij}^{\times} \cdot f_{2i}^{\ell} \Big) \cdot \xi_{1j}^{\ell} \cdot t_1 \Big) \\
&\quad + \quad ((\mathbf{v}^{\times})^T \boldsymbol{\Delta}) \cdot s + (0 \text{ or } \mu).
\end{aligned}$$

Recall that in the equations above, the blue terms are fixed, the red terms depend only on the secret key and the black terms depend only on the ciphertext. Hence, one can observe that if the challenge is 0, then the equation above is a sum of products, where in every product one term depends only on the secret key and the other one depends only on the ciphertext. Concatenating all these elements into two vectors, one obtains

$$(5.4) = \langle \mathbf{a}, \mathbf{b} \rangle,$$

where $\mathbf{a}$ depends only on the secret key and $\mathbf{b}$ depends only on the ciphertext (and they are both secret).

The dimension of $\mathbf{a}$ and $\mathbf{b}$ is the number of terms in the sum above. In our case, this dimension is $2rk + 1$. To see this, note that $\ell \in [k]$ and $j \in [r]$, and that we are summing over ℓ and j so we obtain a sum of kr scalars. Hence, this term may be expressed as one big inner product of two vectors of dimension $2rk + 1$.

Assume that we can make $N := 2rk+2$ requests for secret keys and ciphertexts, and let us write $c_{i,j} = \langle \mathbf{a}_i, \mathbf{b}_j \rangle + (0 \text{ or } \mu_{ij})$ the noise term obtained when evaluating the NLinFE scheme with the i-th secret key and the j-th ciphertext. Recall that the values μ_{ij} are chosen by the adversary. Define $\mathbf{C}$ and $\mathbf{M}$ the $N \times N$ matrices $(c_{i,j})_{i,j}$ and $(\mu_{ij})_{i,j}$ respectively.

Then, depending on the challenge, we claim that $\mathbf{C}$ or $\mathbf{C} - \mathbf{M}$ is of rank at most $N - 1$. To see this, note that we have $\mathbf{C} = \mathbf{A} \cdot \mathbf{B} + (0 \text{ or } M)$, where $\mathbf{A}$ has dimension $N \times N - 1$ and $\mathbf{B}$ has dimension $N - 1 \times N$, so that $\mathbf{A} \cdot \mathbf{B}$ has rank at most $N - 1$. On the other hand, the other matrix is of the form $\mathbf{A} \cdot \mathbf{B} \pm M$, which has full rank with good probability (even if M has only rank 1, the sum of a matrix of rank $N - 1$ and a matrix of rank 1 is likely to have rank N if the two matrices are independent, which is the case here).[6] Hence, computing the

[6] Observe that even if the μ_{ij} are somehow chosen by the adversary, they cannot be chosen arbitrarily. Indeed, μ_{ij} is the scalar product between the vector corresponding

determinant of the matrix $\mathbf{C}$ allows to determine what was the challenge, and to break the security of the NLinFE scheme.

The case of degree > 2 In the general case, if the degree of the NLinFE scheme is d instead of 2, then the same reasoning applies. The only difference is that the vectors $\mathbf{a}$ and $\mathbf{b}$ will have dimension $d \cdot k \cdot r + 1$, so one needs to be able to make $N := d \cdot k \cdot r + 2$ key and ciphertext queries for the attack. More precisely, in degree d, the term (5.4) becomes

$$\begin{aligned}
(5.4) &= \sum_{\delta=1}^{d} \sum_{\ell=1}^{k} \sum_{1 \le i_1, \cdots, i_d \le r} v^{\times}_{i_1, \cdots, i_d} \Big(\prod_{j \neq \delta} f^{\ell}_{j, i_j} \cdot t_j \Big) \xi^{\ell}_{\delta, i_\delta} \; + \; (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu) \\
&= \sum_{\delta=1}^{d} \sum_{\ell=1}^{k} \sum_{1 \le i_1, \cdots, i_d \le r} v^{\times}_{i_1, \cdots, i_d} \Big(\prod_{j \neq \delta} f^{\ell}_{j, i_j} \prod_{j \neq \delta} t_j \Big) \xi^{\ell}_{\delta, i_\delta} \; + \; (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu) \\
&= \sum_{\delta=1}^{d} \sum_{\ell=1}^{k} \sum_{1 \le i_1, \cdots, i_d \le r} v^{\times}_{i_1, \cdots, i_d} \Big(\prod_{j \neq \delta} f^{\ell}_{j, i_j} \prod_{j \neq \delta} t_j \Big) \xi^{\ell}_{\delta, i_\delta} \; + \; (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu) \\
&= \sum_{\delta=1}^{d} \sum_{\ell=1}^{k} \sum_{i_\delta=1}^{r} \Big(\sum_{1 \le i_j \le r, j \neq \delta} v^{\times}_{i_1, \cdots, i_d} \cdot \prod_{j \neq \delta} f^{\ell}_{j, i_j} \cdot \xi^{\ell}_{\delta, i_\delta} \cdot \prod_{j \neq \delta} t_j \Big) \; + \; (\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s + (0 \text{ or } \mu).
\end{aligned}$$

For the first term, we are now summing dkr elements, and each one corresponds to the product of two scalars. Hence, the left term can be written as one inner product of two vectors of dimension $d \cdot k \cdot r$, with one vector depending only on the secret key and one depending only on the ciphertext. The analysis of the term $(\mathbf{v}^{\times})^T \boldsymbol{\Delta} \cdot s$ is the same as before. To conclude, taking $N = d \cdot k \cdot r + 2$ and performing the same attack as above enables us to distinguish whether the challenge is 0 or μ.

We thus obtain the bound $N := d \cdot k \cdot r + 2$ on the number of key requests that can be performed by the attacker. Since $k, r = \text{poly}(\kappa)$, the adversary can obtain this number of keys to conduct the above attack.

6 Modifying Construction to Fix Attacks

In this section, we describe an approach to fix the above two attacks (which we will refer to as "the multiple ciphertext attack" and as the "rank attack" respectively).

to the i-th secret key, with the difference of the two messages of the j-th pair of challenge messages. Hence, the matrix M has rank at most w, where w is the size of these vectors. However, as said above, it is sufficient to have M of rank at least 1 for the attack to go through, and this can be ensured by the attacker (it simply needs to take $M \neq 0$).

Intuitively, the reason for the multiple ciphertext attack to work is commutativity: we mix and match the LWE labels and secrets across multiple ciphertexts to compute the large term in two different ways. An over-simplification is that if two ciphertexts CT_1 and CT_2 have LWE secrets s and t respectively, and a and b are labels, then CT_1 contains encodings with large terms as and bs and CT_2 contains encodings with large terms at and bt. But now, $(as)\cdot(bt) = (bs)\cdot(at)$, which implies that we can multiply encodings from different ciphertexts in two different ways to get the same large term, which may then be removed by subtraction. While the attack developed in Section 4 is more elaborate, the intuition remains the same as in the simplification discussed here.

The reason the the rank attack on the other hand is the presence of the moduli p_0 and p_1, which allow to separate the noise terms, and obtain one noise term which is only linear in the freshly chosen error elements.

Fixing the multiple ciphertext attack. As shown by the above discussion, the chief vulnerability exploited by the attack is commutativity of polynomials. However, if we replace scalar product by inner product, we get that the first ciphertext contains the terms $\langle \mathbf{a}, \mathbf{s} \rangle$ and $\langle \mathbf{b}, \mathbf{s} \rangle$ and the second ciphertext contains the terms $\langle \mathbf{a}, \mathbf{t} \rangle$ and $\langle \mathbf{b}, \mathbf{t} \rangle$. Attempting to launch the above attack shows that:

$$\langle \mathbf{a}, \mathbf{s} \rangle \cdot \langle \mathbf{b}, \mathbf{t} \rangle \neq \langle \mathbf{b}, \mathbf{s} \rangle \cdot \langle \mathbf{a}, \mathbf{t} \rangle$$

This prevents the mix and match attacks of the kind discussed in Section 4 since each large term now uniquely corresponds to a single product of encodings and may not be generated in two different ways. As explained in the full version, the multiple ciphertext attack can still be generalized to this setting, but the modulus q will need to be exponential in the dimension of the vectors for the attack to work, and so we can prevent the attack by choosing the dimension to be larger than $\log q$.

Fixing the rank attack. In order to fix the rank attack, we propose to remove the modulus p_0 from the encodings, i.e., consider encodings of the form $d^{\ell}_{1i} = \langle \mathbf{h}^{\ell}_{1i},\ \mathbf{t}_1 \rangle + p_1 \cdot e^{\ell}_{1i} + \tilde{e}^{\ell}_{1i}$. This way, it will be harder to split the noise term at the end (we will only have three "levels" 1, p_1 and p_1^2 instead of 5 before), and we will show that the noise terms obtained this way seem hard to exploit now. One may want to also remove the modulus p_1 from the construction, and only consider one noise term, but as we should see in the construction, the modulus p_1 is needed for correctness (not only for the shape of the output noise), and so cannot be removed easily.

Recall that the modulus p_0 were present because we wanted to flood a noise term of the form $\mathsf{noise}_0 \cdot p_0$ (the modulus p_1 is used because the messages are living in R_{p_1}). In more generality, in the bootstrapping procedure used in [2] to construct iO, we will want to flood a noise term of the form $\mathsf{noise}_0 \cdot p_0 + \cdots + \mathsf{noise}_{D-2} \cdot p_{D-2}$ for some integer D related to the degree of the FE scheme we want to construct. We will also want the message space of the NLinFE scheme to be $R_{p_{D-1}}$ and the ciphertext space to be R_{p_D}, with $p_0 < p_1 < \cdots < p_D$ for prime numbers p_i. We also want for the bootsrapping procedure that the noise output by the NLinFE scheme be of the form $\mathsf{noise}'_0 \cdot p_0 + \cdots + \mathsf{noise}'_{D-2} \cdot p_{D-2}$, so that when we add this noise to the original noise, we still have a linear combination of the p_i's, with $i \leq D-2$.

From arbitrary flooding noise to structured flooding noise. When we remove the moduli from Agrawal's construction as discussed above, we obtain an NLinFE scheme where the flooding noise term is arbitrary in R, and so not of the desired shape $\mathsf{noise}'_0 \cdot p_0 + \cdots + \mathsf{noise}'_{D-2} \cdot p_{D-2}$. We can however use this NLinFE scheme to construct a new NLinFE' scheme, with a flooding noise term of the desired shape. Intuitively, the idea is to use an additive secret sharing of the messages $\mathbf{z} = \mathbf{z}_0 + \cdots + \mathbf{z}_{D-2}$, and then encode $\mathbf{z}_0/p_0, \cdots, \mathbf{z}_{D-2}/p_{D-2}$ using the NLinFE scheme without moduli. To recover the scalar product $\langle \mathbf{v}, \mathbf{z} \rangle$, one then compute $p_0 \cdot \langle \mathbf{v}, \mathbf{z}_0/p_0 \rangle + \cdots + p_{D-2} \cdot \langle \mathbf{v}, \mathbf{z}_{D-2}/p_{D-2} \rangle$, and so the noise term will have the desired shape.

More precisely, the NLinFE' scheme proceeds as follows

$\mathsf{NLinFE}'.\mathsf{Setup}(1^\kappa, 1^w)$: Run $\mathsf{NLinFE}.\mathsf{Setup}(1^\kappa, 1^w)$ $D-1$ times to obtain $D-1$ master secret keys MSK_i and output $(\mathsf{MSK}_0, \cdots, \mathsf{MSK}_{D-2})$.

$\mathsf{NLinFE}'.\mathsf{Enc}((\mathsf{MSK}_0, \cdots, \mathsf{MSK}_{D-2}), \mathbf{z})$: where $\mathbf{z} \in R^w_{p_{D-1}}$.

1. Sample $(\mathbf{z}_0, \cdots, \mathbf{z}_{D-3})$ uniformly at random in $R_{p_{D-1}}$ and define $\mathbf{z}_{D-2}$ such that $\sum_{i=0}^{D-2} \mathbf{z}_i = \mathbf{z}$.
2. For i in $\{0, \cdots, D-2\}$, compute $\mathsf{CT}_i = \mathsf{NLinFE}.\mathsf{Enc}(\mathsf{MSK}_i, \mathbf{z}_i/p_i)$. Here, the division by p_i is performed modulo p_{D-1}, and is possible because p_i is coprime with p_{D-1} for all $i \leq D-2$.

Output $\mathsf{CT}_\mathbf{z} = (\mathsf{CT}_0, \cdots, \mathsf{CT}_{D-2})$.

$\mathsf{NLinFE}'.\mathsf{KeyGen}(\mathsf{MSK}, \mathbf{v}, \mathbf{v}^\times)$: output

$$\mathsf{SK}_\mathbf{v} = (\mathsf{NLinFE}.\mathsf{KeyGen}(\mathsf{MSK}_0, \mathbf{v}, \mathbf{v}^\times), \cdots, \mathsf{NLinFE}.\mathsf{KeyGen}(\mathsf{MSK}_{D-2}, \mathbf{v}, \mathbf{v}^\times)).$$

$\mathsf{NLinFE}'.\mathsf{Dec}(\mathsf{CT}_\mathbf{z}, \mathsf{SK}_\mathbf{v})$: where $\mathbf{z} \in R^w_{p_{D-1}}$.

1. Parse $\mathsf{CT}_\mathbf{z}$ as $\mathsf{CT}_\mathbf{z} = (\mathsf{CT}_0, \cdots, \mathsf{CT}_{D-2})$ and $\mathsf{SK}_\mathbf{v}$ as $\mathsf{SK}_\mathbf{v} = (\mathsf{SK}_1, \cdots, \mathsf{SK}_{D-2})$.
2. Compute $y_i = \mathsf{NLinFE}.\mathsf{Dec}(\mathsf{CT}_i, \mathsf{SK}_i) \in R_{p_{D-1}}$ for $0 \leq i \leq D-2$.

Output $\sum_{i=0}^{D-2} p_i y_i \bmod p_{D-1}$.

For correctness, observe that in the NLinFE' decryption algorithm, we have $y_i = \langle \mathbf{z}_i/p_i, \mathbf{v} \rangle + \mathsf{noise}_i$ by correctness of NLinFE (if the ciphertexts and secret keys are valid). So the output is indeed of the form $\langle \mathbf{z}, \mathbf{v} \rangle + \sum_i \mathsf{noise}_i \cdot p_i$: we have $\langle \mathbf{z}, \mathbf{v} \rangle$ plus a noise term of the desired shape.

We conclude that, for the bootstrapping procedure of [2], it is sufficient to construct an NLinFE scheme, with message space $R_{p_{D-1}}$, ciphertext space R_{p_D} and arbitrary flooding noise. The new NLinFE construction we propose below satisfies these conditions.

6.1 The New NLinFE Construction

Below, we present a modified variant of the NLinFE construction of [2], designed to avoid the multiple ciphertext attack and the rank attack, as discussed above.

$\mathsf{NLinFE.Setup}(1^\kappa, 1^w)$: On input a security parameter κ, a parameter w denoting the length of the function and message vectors, do the following:

1. Sample $\mathbf{W} \leftarrow R_{p_D}^{m\times\kappa^2}$ with a trapdoor $\mathbf{T}$ using the algorithm $\mathsf{TrapGen}$.
2. Sample $\mathbf{E} \in \mathcal{D}^{m\times w}$ and set $\mathbf{A} = \mathbf{E}^\intercal\mathbf{W} \in R_{p_D}^{w\times\kappa^2}$ (recall that $\mathcal{D}$ is a discrete Gaussian distribution over R of parameter σ).
3. For $i \in \{1,\ldots,r\}$, $\ell \in \{1,\ldots,k\}$, sample $\mathbf{f}_{1i}^\ell, \mathbf{f}_{2i}^\ell \leftarrow \mathcal{D}^\kappa$ and $g_1^\ell, g_2^\ell \leftarrow \mathcal{D}$. If g_1^ℓ, g_2^ℓ are not invertible over R_{p_D}, resample.
   Set
$$\mathbf{h}_{1i}^\ell = \frac{\mathbf{f}_{1i}^\ell}{g_1^\ell}, \quad \mathbf{h}_{2i}^\ell = \frac{\mathbf{f}_{2i}^\ell}{g_2^\ell} \in R_{p_D}^\kappa$$
4. Sample a PRF seed, denoted as seed.

Output
$$\mathsf{MSK} = \Big(\mathbf{W}, \mathbf{T}, \mathbf{A}, \mathbf{E}, \big\{\mathbf{f}_{1i}^\ell, \mathbf{f}_{2i}^\ell\big\}_{i\in[r],\ell\in[k]}, \big\{g_1^\ell, g_2^\ell\big\}_{\ell\in[k]}, \mathsf{seed} \Big)$$

$\mathsf{NLinFE.Enc}(\mathsf{MSK}, \mathbf{z})$: On input public key MSK, a message vector $\mathbf{z} \in R_{p_{D-1}}^w$, do:

1. Sample $\mathbf{t}_1, \mathbf{t}_2 \leftarrow \mathcal{D}^\kappa$. Set $\mathbf{s} = \mathbf{t}_1 \otimes \mathbf{t}_2 \in R^{\kappa^2}$.
2. **Construct Message Encodings.** Sample $\boldsymbol{\nu} \leftarrow \mathcal{D}^m$, $\boldsymbol{\eta} \leftarrow \mathcal{D}^w$ and compute:
$$\mathbf{c} = \mathbf{W}\mathbf{s} + p_{D-1}\cdot\boldsymbol{\nu} \in R_{p_D}^m, \quad \mathbf{b} = \mathbf{A}\mathbf{s} + p_{D-1}\cdot\boldsymbol{\eta} + \mathbf{z} \in R_{p_D}^w,$$
   where $\mathbf{z} \in R_{p_{D-1}}^w$ is seen as a vector of R with coefficients in $(-\frac{p_{D-1}}{2}, \frac{p_{D-1}}{2}]$ and then reduced modulo p_D.
3. **Sample Structured Noise.** To compute encodings of noise, do the following:
   (a) Define lattices:
$$\Lambda_1^\ell \overset{\text{def}}{=} g_1^\ell \cdot R, \quad \Lambda_2^\ell \overset{\text{def}}{=} g_2^\ell \cdot R$$
   (b) Sample noise terms from the above lattices as:
$$e_{1i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_2^\ell), \tilde{e}_{1i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_2^\ell), \quad e_{2i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_1^\ell), \tilde{e}_{2i}^\ell \leftarrow \widehat{\mathcal{D}}(\Lambda_1^\ell) \quad \forall i \in [r], \ell \in [k].$$
   Here $\widehat{\mathcal{D}}(\Lambda_1^\ell)$ is a discrete Gaussian distribution on Λ_1^ℓ and $\widehat{\mathcal{D}}(\Lambda_2^\ell)$ is a discrete Gaussian distributions on Λ_2^ℓ, both of parameter σ.
4. **Compute Encodings of Noise.**
   (a) Let
$$d_{1i}^\ell = \langle \mathbf{h}_{1i}^\ell,\ \mathbf{t}_1\rangle + p_{D-1}\cdot e_{1i}^\ell + \tilde{e}_{1i}^\ell \in R_{p_D} \quad \forall i \in [r], \ell \in [k].$$
   Let $\mathbf{d}_1^\ell = (d_{1i}^\ell)$.

(b) Similarly, let

$$d_{2i}^{\ell} = \langle \mathbf{h}_{2i}^{\ell},\ \mathbf{t}_2 \rangle + p_{D-1} \cdot e_{2i}^{\ell} + \tilde{e}_{2i}^{\ell} \in R_{p_2} \quad \forall i \in [r], \ell \in [k].$$

Let $\mathbf{d}_2^{\ell} = (d_{2i}^{\ell})$.

5. **Output Ciphertext.** Output message encodings $(\mathbf{c}, \mathbf{b})$ and noise encodings $(\mathbf{d}_1^{\ell}, \mathbf{d}_2^{\ell})$ for $\ell \in [k]$.

$\mathsf{NLinFE.KeyGen}(\mathsf{MSK}, \mathbf{v}, \mathbf{v}^{\times})$: On input the master secret key MSK, NLinFE function vectors $\mathbf{v} \in R_{p_{D-1}}^{w}$ and $\mathbf{v}^{\times} \in R^{L}$ with coefficients small compared to p_{D-1}, do the following.

1. **Sampling Basis Preimage vectors.**

(a) Sample short $\mathbf{e}_{ij} \in R^m$ using $\mathsf{SamplePre}$ with randomness $\mathsf{PRF}(\mathsf{seed}, ij)$ such that

$$\mathbf{W}^{\intercal}\mathbf{e}_{ij} = \mathbf{h}_{ij}, \text{ where } \mathbf{h}_{ij} \overset{\text{def}}{=} \sum_{\ell \in [k]} \mathbf{h}_{1i}^{\ell} \otimes \mathbf{h}_{2j}^{\ell} + p_{D-1}\mathbf{\Delta}_{ij} + \tilde{\mathbf{\Delta}}_{ij}.$$

Above $\mathbf{\Delta}_{ij}, \tilde{\mathbf{\Delta}}_{ij} \leftarrow \mathcal{D}^{\kappa^2} \in R^{\kappa^2}$ for $1 \leq j \leq i \leq r$.

$$\text{Let } \mathbf{E}^{\times} = (\mathbf{e}_{ij}) \in R^{m \times L}$$

where $L = |1 \leq j \leq i \leq r|$.

2. **Combining Basis Preimages to Functional Preimage.** Define

$$\mathbf{k}_{\mathbf{v}} = \mathbf{E} \cdot \mathbf{v} + \mathbf{E}^{\times} \cdot \mathbf{v}^{\times} \quad \in R^m \tag{6.1}$$

3. Output $(\mathbf{k}_{\mathbf{v}}, \mathbf{v}, \mathbf{v}^{\times})$.

$\mathsf{NLinFE.Dec}(\mathsf{CT}_{\mathbf{z}}, \mathsf{SK}_{\mathbf{v}})$: On input a ciphertext $\mathsf{CT}_{\mathbf{z}} = \big(\ \mathbf{c}, \mathbf{b}, \{\mathbf{d}_1^{\ell}, \mathbf{d}_2^{\ell}\}_{\ell \in [k]}\ \big)$ and a secret key $\mathbf{k}_{\mathbf{v}}$ for function $\mathbf{v}$, do the following

1. Compute encoding of noise term on the fly as:

$$\mathbf{d}^{\times} \overset{\text{def}}{=} (\sum_{\ell \in [k]} \mathbf{d}_1^{\ell} \otimes \mathbf{d}_2^{\ell}) \in R_{p_D}^{L}$$

2. Compute functional ciphertext as:

$$b_{\mathbf{v}} = \mathbf{v}^{\intercal}\mathbf{b} + (\mathbf{v}^{\times})^{\intercal}\mathbf{d}^{\times} \in R_{p_D}$$

3. Compute $(b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\intercal}\mathbf{c} \bmod p_D) \bmod p_{D-1}$ and output it.

Computing Encoding of Correlated Noise Term for the new construction

We compute $d^\ell_{1i} \cdot d^\ell_{2j}$. Recall that

$$d^\ell_{1i} = \langle \mathbf{h}^\ell_{1i},\ \mathbf{t}_1 \rangle + p_{D-1} \cdot e^\ell_{1i} + \tilde{e}^\ell_{1i} \in R_{p_D}$$

$$d^\ell_{2j} = \langle \mathbf{h}^\ell_{2j},\ \mathbf{t}_2 \rangle + p_{D-1} \cdot e^\ell_{2i} + \tilde{e}^\ell_{2i} \in R_{p_D}$$

Recall also that $e^\ell_{1i}, \tilde{e}^\ell_{1i}$ are sampled from lattice Λ^ℓ_2 and $e^\ell_{2i}, \tilde{e}^\ell_{2i}$ are sampled from lattice Λ^ℓ_1.

$$\begin{aligned} \text{Let} \quad & e^\ell_{1i} = g^\ell_2 \cdot \xi^\ell_{1i}, \quad \tilde{e}^\ell_{1i} = g^\ell_2 \cdot \tilde{\xi}^\ell_{1i}, \\ \text{and} \quad & e^\ell_{2i} = g^\ell_1 \cdot \xi^\ell_{2i}, \quad \tilde{e}^\ell_{2i} = g^\ell_1 \cdot \tilde{\xi}^\ell_{2i} \end{aligned}$$

Now, we may compute:

$$\begin{aligned} d^\ell_{1i} \cdot d^\ell_{2j} &= \Big(\langle \mathbf{h}^\ell_{1i},\ \mathbf{t}_1 \rangle + p_{D-1} \cdot e^\ell_{1j} + \tilde{e}^\ell_{1j} \Big) \cdot \Big(\langle \mathbf{h}^\ell_{2j},\ \mathbf{t}_2 \rangle + p_{D-1} \cdot e^\ell_{2j} + \tilde{e}^\ell_{2j} \Big) \\ &= \langle \mathbf{h}^\ell_{1i} \otimes \mathbf{h}^\ell_{2j},\ \underbrace{(\mathbf{t}_1 \otimes \mathbf{t}_2)}_{\mathbf{s}} \rangle + p_{D-1} \cdot \Big(\underbrace{p_{D-1} \cdot (g^\ell_2 \cdot \xi^\ell_{1i} \cdot g^\ell_1 \cdot \xi^\ell_{2j})}_{\mathsf{small}_D} \\ &\quad + \underbrace{(g^\ell_2 \cdot \tilde{\xi}^\ell_{1i} \cdot g^\ell_1 \cdot \xi^\ell_{2j} + g^\ell_2 \cdot \xi^\ell_{1i} \cdot g^\ell_1 \cdot \tilde{\xi}^\ell_{2j})}_{\mathsf{small}_{D-1}} + \underbrace{(\langle \mathbf{f}^\ell_{1i},\ \mathbf{t}_1 \rangle \cdot \xi^\ell_{2j} + \langle \mathbf{f}^\ell_{2j},\ \mathbf{t}_2 \rangle \cdot \xi^\ell_{1i})}_{\mathsf{small}_{D-1}} \Big) \\ &\quad + \underbrace{\Big((g^\ell_2 \cdot \tilde{\xi}^\ell_{1i} \cdot g^\ell_1 \cdot \tilde{\xi}^\ell_{2j}) + (\langle \mathbf{f}^\ell_{1i},\ \mathbf{t}_1 \rangle \cdot \tilde{\xi}^\ell_{2j} + \langle \mathbf{f}^\ell_{2j},\ \mathbf{t}_2 \rangle \cdot \tilde{\xi}^\ell_{1i}) \Big)}_{\mathsf{small}_{D-1}} \end{aligned}$$

(recall that small_i is a term that is small compared to p_i for $i = D-1$ or D).

$$\text{Thus, } \sum_{\ell \in [k]} d^\ell_{1i} \cdot d^\ell_{2j} = \langle (\sum_{\ell \in [k]} \mathbf{h}^\ell_{1i} \otimes \mathbf{h}^\ell_{2j}),\ \mathbf{s} \rangle + p_{D-1} \cdot \mathsf{small}_D + \mathsf{small}_{D-1}. \tag{6.2}$$

Fig. 6.1. Computing encoding of noise term as polynomial of encodings in the new construction of Section 6.

Correctness. In this section, we establish that the above scheme is correct. To simplify the analysis, we let small_{D-1} denote any term which is small compared to p_{D-1} and small_D be any term which is small compared to p_D. We also assume that summing polynomially many small_i terms or multiplying a constant number of them results in an element which is still a small_i (for $i = D-1$ or D). We also assume that the parameters are set so that σ is small compared to p_{D-1} and that p_{D-1} is small compared to p_D.

Let us do the analysis by walking through the steps performed by the decrypt algorithm:

1. We compute an encoding of a correlated noise term on the fly as described in Figure 6.1.
2. The decryption equation is:

$$b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{c} = (\mathbf{v}^{\mathsf{T}}\mathbf{b} + (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{d}^{\times}) - \mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{c}$$

3. Recall that $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + p_{D-1} \cdot \boldsymbol{\eta} + \mathbf{z} \in R_{p_D}^{w}$. Hence,

$$\mathbf{v}^{\mathsf{T}}\mathbf{b} = \mathbf{v}^{\mathsf{T}}\mathbf{A} \cdot \mathbf{s} + p_{D-1} \cdot \mathsf{small}_D + \mathbf{v}^{\mathsf{T}}\mathbf{z}$$

4. Let $\mathbf{H}_{ij}^{\times} = \big(\sum\limits_{\ell \in [k]} \mathbf{h}_{1i}^{\ell} \otimes \mathbf{h}_{2j}^{\ell} \big)$ be the $(i,j)^{th}$ row of $\mathbf{H}^{\times} \in R_{p_D}^{L \times \kappa^2}$. Since

$$\mathbf{d}^{\times} = \mathbf{H}^{\times}\mathbf{s} + p_{D-1} \cdot \mathsf{small}_D + \mathsf{small}_{D-1}$$

   and $\mathbf{v}^{\times} \in R^{L}$ is small compared to p_{D-1}, we have

$$(\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{d}^{\times} = (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{H}^{\times}\mathbf{s} + p_{D-1} \cdot \mathsf{small}_D + \mathsf{small}_{D-1}$$

   Hence we have

$$\mathbf{v}^{\mathsf{T}}\mathbf{b} + (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{d}^{\times} = \big(\mathbf{v}^{\mathsf{T}}\mathbf{A} + (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{H}^{\times}\big)\mathbf{s} + p_{D-1} \cdot \mathsf{small}_D + \mathsf{small}_{D-1} + \mathbf{v}^{\mathsf{T}}\mathbf{z}$$

5. Next, note that

$$\mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{W} = \mathbf{v}^{\mathsf{T}}\mathbf{A} + (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{H}^{\times} + p_{D-1} \cdot \mathsf{small}_{D-1} + \mathsf{small}_{D-1} \overset{\text{def}}{=} \mathbf{a}_{\mathbf{v}} \in R_{p_D}^{1 \times \kappa^2}$$

6. Recall that $\mathbf{c} = \mathbf{W} \cdot \mathbf{s} + p_{D-1} \cdot \boldsymbol{\nu}$ hence,

$$\begin{aligned} \mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{c} &= \mathbf{a}_{\mathbf{v}}^{\mathsf{T}}\mathbf{s} + p_{D-1} \cdot \langle \boldsymbol{\nu}, \mathbf{k}_{\mathbf{v}} \rangle \\ &= \left(\mathbf{v}^{\mathsf{T}}\mathbf{A} + (\mathbf{v}^{\times})^{\mathsf{T}}\mathbf{H}^{\times}\right)\mathbf{s} + \mathsf{small}_{D-1} + p_{D-1} \cdot \mathsf{small}_D \end{aligned}$$

7. Hence, $b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{c} = \mathbf{v}^{\mathsf{T}}\mathbf{z} + \mathsf{small}_{D-1} + p_{D-1} \cdot \mathsf{small}_D$. The right hand side of this equation is smaller than p_D by assumption (if the parameters are carefully chosen), hence, by computing $b_{\mathbf{v}} - \mathbf{k}_{\mathbf{v}}^{\mathsf{T}}\mathbf{c}$ in R_{p_D}, we recover $\mathbf{v}^{\mathsf{T}}\mathbf{z} + \mathsf{small}_{D-1} + p_{D-1} \cdot \mathsf{small}_D$ over R. Now, reducing this term modulo p_{D-1} leads to $\mathbf{v}^{\mathsf{T}}\mathbf{z} + \mathsf{small}_{D-1} \bmod p_1$, where small_{D-1} is small compared to p_{D-1}.

   On the degree of the noise term. As was already observed in Agrawal's original construction [2], the construction above is described with a noise term of degree $d = 2$, but it could easily be generalized to any constant degree d. In the case of a general degree d, we would have d-tuples of encodings $(d_{1i}^{\ell}, \cdots, d_{di}^{\ell})$, where the noise in d_{ai}^{ℓ} is a multiple of $\prod_{b \neq a} g_b^{\ell}$. Then, when computing $\mathbf{d}^{\times}$, one would consider all possible products $d_{1i_1}^{\ell} \cdots d_{d,i_d}^{\ell}$ and obtain a noise term of degree d. Please see the full version for details. For simplicity, we described above the variant with $d = 2$, but we show in the full versionthat for security we need $d \geq 3$.

7 Setting the parameters

We provide in the full version a discussion on the security of the new NLinFE scheme described above. In particular, we generalize the attacks presented in Sections 4 and 5 and argue that our new scheme is not vulnerable to them. Below, we provide an instantiation of the parameters of the scheme which we believe is secure, even against a quantum computer (see the full version for more details). The parameter N is the maximal number of key requests that an attacker is allowed to performed. This parameter should be superpolynomially larger than the ciphertext size for the NLinFE scheme to imply iO.

- κ is the security parameter and $B_1 = \mathrm{poly}(\kappa)$ is given as input
- $d = 3$
- $k = \kappa^3$ and $r = \kappa$
- $\sigma = 2^\kappa \cdot B_1$
- $p_{D-1} = \sigma^{2d}$ and $p_D = \sigma^{6d}$
- $m = \kappa^d \cdot \log p_D$
- w is arbitrary up to κ^{d+1}
- $N = \kappa^{d+2.5}$.

Acknowledgments. This work was supported in part by CyberSecurity Research Flanders with reference number VR20192203 and by the Research Council KU Leuven grant C14/18/067 on Cryptanalysis of post-quantum cryptography.

References

1. M. Abdalla, F. Bourse, A. D. Caro, and D. Pointcheval. Simple functional encryption schemes for inner products. Cryptology ePrint Archive, Report 2015/017, 2015. http://eprint.iacr.org/ To appear in PKC'15.
2. S. Agrawal. Indistinguishability obfuscation without multilinear maps: New techniques for bootstrapping and instantiation. In *Eurocrypt*, 2019.
3. S. Agrawal, B. Libert, and D. Stehle. Fully secure functional encryption for linear functions from standard assumptions, and applications. In *Crypto*, 2016.
4. M. Ajtai. Generating hard instances of the short basis problem. In *ICALP*, volume 1644 of *LNCS*, pages 1–9. Springer, 1999.
5. J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. In *STACS*, pages 75–86, 2009.
6. P. Ananth and A. Jain. Indistinguishability obfuscation from compact functional encryption. In *CRYPTO*, pages 308–326, 2015.
7. P. Ananth, A. Jain, H. Lin, C. Matt, and A. Sahai. Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. In *Crypto*, pages 284–332. Springer, 2019.
8. D. Apon, N. Döttling, S. Garg, and P. Mukherjee. Cryptanalysis of indistinguishability obfuscations of circuits over ggh13. eprint 2016, 2016.
9. B. Applebaum and Z. Brakerski. Obfuscating circuits via composite-order graded encoding. In *Theory of Cryptography Conference*, pages 528–556. Springer, 2015.

10. B. Barak, Z. Brakerski, I. Komargodski, and P. Kothari. Limits on low-degree pseudorandom generators (or: Sum-of-squares meets program obfuscation), 2017.
11. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, 2001.
12. M. Bardet, J.-C. Faugère, and B. Salvy. On the complexity of the f5 gröbner basis algorithm. *Journal of Symbolic Computation*, 70:49–70, 2015.
13. N. Bitansky, S. Garg, H. Lin, R. Pass, and S. Telang. Succinct randomized encodings and their applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 439–448, 2015.
14. N. Bitansky, O. Paneth, and D. Wichs. Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 474–502, 2016.
15. N. Bitansky and V. Vaikuntanathan. Indistinguishability obfuscation from functional encryption. *FOCS*, 2015:163, 2015.
16. R. P. Brent and B. D. McKay. Determinants and ranks of random matrices over zm. *Discrete Mathematics*, 66(1-2):35–49, 1987.
17. R. Canetti, J. Holmgren, A. Jain, and V. Vaikuntanathan. Succinct garbling and indistinguishability obfuscation for RAM programs. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 429–437, 2015.
18. R. Canetti, H. Lin, S. Tessaro, and V. Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC*, 2015.
19. Y. Chen, C. Gentry, and S. Halevi. Cryptanalyses of candidate branching program obfuscators. In *Eurocrypt*, pages 278–307. Springer, 2017.
20. Y. Chen, V. Vaikuntanathan, and H. Wee. Ggh15 beyond permutation branching programs: Proofs, attacks, and candidates. In *Annual International Cryptology Conference*, pages 577–607. Springer, 2018.
21. J. H. Cheon, W. Cho, M. Hhan, J. Kim, and C. Lee. Statistical zeroizing attack: Cryptanalysis of candidates of bp obfuscation over ggh15 multilinear map. In *CRYPTO*, 2019.
22. J.-H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé. Cryptanalysis of the multilinear map over the integers. In *Proc. of EUROCRYPT*, volume 9056 of *LNCS*, pages 3–12. Springer, 2015.
23. J. H. Cheon, M. Hhan, J. Kim, and C. Lee. Cryptanalyses of branching program obfuscations over ggh13 multilinear map from the ntru problem. In *Crypto*, pages 184–210. Springer, 2018.
24. J.-S. Coron, C. Gentry, S. Halevi, T. Lepoint, H. K. Maji, E. Miles, M. Raykova, A. Sahai, and M. Tibouchi. Zeroizing without low-level zeroes: New mmap attacks and their limitations. In *Advances in Cryptology–CRYPTO 2015*, pages 247–266. Springer, 2015.
25. J.-S. Coron, M. S. Lee, T. Lepoint, and M. Tibouchi. Zeroizing attacks on indistinguishability obfuscation over clt13. Eprint 2016, 2016.
26. L. Ducas and A. Pellet-Mary. On the statistical leak of the GGH13 multilinear map and some variants. pages 465–493, 2018.
27. S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, 2013.

28. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013. http://eprint.iacr.org/.
29. S. Garg, E. Miles, P. Mukherjee, A. Sahai, A. Srinivasan, and M. Zhandry. Secure obfuscation in a weak multilinear map model. In M. Hirt and A. Smith, editors, *Theory of Cryptography*, pages 241–268, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
30. C. Gentry, C. S. Jutla, and D. Kane. Obfuscation using tensor products, 2018.
31. Y. Hu and H. Jia. Cryptanalysis of ggh map. In *Eurocrypt*, pages 537–565. Springer, 2016.
32. A. Jain, H. Lin, C. Matt, and A. Sahai. How to leverage hardness of constant-degree expanding polynomials over r to build io. In *Advances in Cryptology-EUROCRYPT*, pages 19–23, 2019.
33. I. Komargodski, T. Moran, M. Naor, R. Pass, A. Rosen, and E. Yogev. One-way functions and (im)perfect obfuscation. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, 2014.
34. V. Koppula, A. B. Lewko, and B. Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 419–428, 2015.
35. H. Lin. Indistinguishability obfuscation from sxdh on 5-linear maps and locality-5 prgs. In *Crypto*, 2017.
36. H. Lin, R. Pass, K. Seth, and S. Telang. Output-compressing randomized encodings and applications. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, pages 96–124, 2016.
37. H. Lin and S. Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local prgs. In *Crypto*, 2017.
38. H. Lin and V. Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In *FOCS*, 2016.
39. A. Lombardi and V. Vaikuntanathan. On the non-existence of blockwise 2-local prgs with applications to indistinguishability obfuscation. IACR Cryptology ePrint Archive, http://eprint.iacr.org/2017/301, 2017.
40. E. W. Mayr. Some complexity results for polynomial ideals. *Journal of complexity*, 13(3):303–325, 1997.
41. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.
42. E. Miles, A. Sahai, and M. Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over ggh13. In *Crypto*, 2016.
43. A. Pellet-Mary. Quantum attacks against indistinguishablility obfuscators proved secure in the weak multilinear map model. In *CRYPTO*, pages 153–183, 2018.
44. A. Sahai and B. Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *STOC*, 2014. http://eprint.iacr.org/2013/454.pdf.
45. D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices. In *ASIACRYPT*, pages 617–635, 2009.
46. J. Zimmerman. How to obfuscate programs directly. In *Eurocrypt*, pages 439–467. Springer, 2015.