# Indistinguishability Obfuscation Without Multilinear Maps: New methods for Bootstrapping and Instantiation

Shweta Agrawal[⋆]

**Abstract.** Constructing indistinguishability obfuscation (iO) [17] is a central open question in cryptography. We provide new methods to make progress towards this goal. Our contributions may be summarized as follows:

1. **Bootstrapping.** In a recent work, Lin and Tessaro [71] (LT) show that iO may be constructed using i) Functional Encryption (FE) for polynomials of degree $L$, ii) Pseudorandom Generators (PRG) with *blockwise locality* $L$ and polynomial expansion, and iii) Learning With Errors (LWE). Since there exist constructions of FE for quadratic polynomials from standard assumptions on bilinear maps [68, 16], the ideal scenario would be to set $L = 2$, yielding iO from widely believed assumptions.

   Unfortunately, it was shown soon after [73, 18] that PRG with block locality 2 and the expansion factor required by the LT construction, concretely $\Omega(n \cdot 2^{b(3+\epsilon)})$, where $n$ is the input length and $b$ is the block length, do not exist. In the worst case, these lower bounds rule out 2-block local PRG with stretch $\Omega(n \cdot 2^{b(2+\epsilon)})$. While [73, 18] provided strong negative evidence for constructing iO based on bilinear maps, they could not rule out the possibility completely; a tantalizing gap has remained. Given the current state of lower bounds, the existence of 2 block local PRG with expansion factor $\Omega(n \cdot 2^{b(1+\epsilon)})$ remains open, although this stretch does not suffice for the LT bootstrapping, and is hence unclear to be relevant for iO.

   In this work, we improve the state of affairs as follows.

   (a) *Weakening requirements on Boolean PRGs:* In this work, we show that the narrow window of expansion factors left open by lower bounds *do* suffice for iO. We show a new method to construct FE for $\mathsf{NC}_1$ from i) FE for degree $L$ polynomials, ii) PRGs of block locality $L$ and expansion factor $\tilde{\Omega}(n \cdot 2^{b(1+\epsilon)})$, and iii) LWE (or RLWE).

   (b) *Broadening class of sufficient randomness generators*: Our bootstrapping theorem may be instantiated with a broader class of pseudorandom generators than hitherto considered for iO, and may circumvent lower bounds known for the arithmetic degree of iO-sufficient PRGs [73, 18]; in particular, these may admit instantiations with arithmetic degree 2, yielding iO with the additional assumptions of SXDH on Bilinear maps and LWE. In more detail, we may use the following two classes of PRG:

      i. *Non-Boolean PRGs*: We may use pseudorandom generators whose inputs and outputs need not be Boolean but may be integers

[⋆] IIT Madras, India. Email: shweta.a@cse.iitm.ac.in.

restricted to a small (polynomial) range. Additionally, the outputs are not required to be pseudorandom but must only satisfy a milder indistinguishability property[1].

ii. *Correlated Noise Generators*: We introduce an even weaker class of pseudorandom generators, which we call correlated noise generators (CNG) which may not only be non-Boolean but are required to satisfy an even milder (seeming) indistinguishability property than $\Delta$ RG.

(c) *Assumptions and Efficiency.* Our bootstrapping theorems can be based on the hardness of the Learning With Errors problem or its ring variant (LWE/RLWE) and can compile FE for degree $L$ polynomials directly to FE for $\mathsf{NC}_1$. Previous work compiles FE for degree $L$ polynomials to FE for $\mathsf{NC}_0$ to FE for $\mathsf{NC}_1$ to iO [72, 68, 12, 45].

Our method for bootstrapping to $\mathsf{NC}_1$ does not go via randomized encodings as in previous works, which makes it simpler and more efficient than in previous works.

2. **Instantiating Primitives.** In this work, we provide the first direct candidate of FE for constant degree polynomials from new assumptions on lattices. Our construction is new and does not go via multilinear maps or graded encoding schemes as all previous constructions. Together with the bootstrapping step above, this yields a **completely new candidate for** iO (as well as FE for $\mathsf{NC}_1$), which makes no use of multilinear or even bilinear maps. Our construction is based on the ring learning with errors assumption (RLWE) as well as new untested assumptions on NTRU rings.

   We provide a detailed security analysis and discuss why previously known attacks in the context of multilinear maps, especially zeroizing and annihilation attacks, do not appear to apply to our setting. We caution that our construction must yet be subject to rigorous cryptanalysis by the community before confidence can be gained in its security. However, we believe that the significant departure from known multilinear map based constructions opens up a new and potentially fruitful direction to explore in the quest for iO.

   Our construction is based entirely on lattices, due to which one may hope for post quantum security . Note that this feature is not enjoyed by instantiations that make any use of bilinear maps even if secure instances of weak PRGs, as identified by the present work, the follow-up by Lin and Matt [69] and the independent work by Ananth, Jain and Sahai [9] are found.

[1] We note that our notion of non Boolean PRGs is qualitatively similar to the notion of $\Delta$ RGs defined in the concurrent work of Ananth, Jain and Sahai [9]. We emphasize that the methods of [9] and the present work are very different, but both works independently discover the same notion of weak PRG as sufficient for building iO.

## 1 Introduction

*Indistinguishability Obfuscation.* Program obfuscation aims to make a program "unintelligible" while preserving its functionality. Indistinguishability obfuscation [17] (iO) is a flavour of obfuscation, which converts a circuit $C$ to an obfuscated circuit $\mathcal{O}(C)$ such that any two circuits that have the same size and compute the same function are indistinguishable to a computationally bounded adversary.

While it is non-obvious at first glance what this notion is useful for, recent work has demonstrated the tremendous power of iO. iO can be used to construct almost any cryptographic object that one may desire – ranging (non-exhaustively) from classical primitives such as one way functions [63], trapdoor permutations [22], public key encryption [82] to deinable encryption [82], fully homomorphic encryption [31], functional encryption [45], succinct garbling schemes [30, 20, 64, 70] and many more.

The breakthrough work of Garg et al. [45] presented the first candidate construction of iO from the beautiful machinery of graded encoding schemes [43]. This work heralded substantial research effort towards understanding iO: from cryptanalysis to new constructions to understanding and weakening underlying assumptions to applications. On the cryptanalysis front, unfortunately, all known candidate graded encoding schemes [43, 39, 51] as well as several candidates of iO have been broken [33, 37, 60, 35, 34, 75, 38, 13]. Given the power of iO, a central question in cryptography is to construct iO from better understood hardness assumptions.

*Functional Encryption.* Functional encryption (FE) [81, 80] is a generalization of public key encryption in which secret keys correspond to programs rather than users. In more detail, a secret key embeds inside it a circuit, say $f$, so that given a secret key $\mathsf{SK}_f$ and ciphertext $\mathsf{CT}_{\mathbf{x}}$ encrypting a message $\mathbf{x}$, the user may run the decryption procedure to learn the value $f(\mathbf{x})$. Security of the system guarantees that nothing beyond $f(\mathbf{x})$ can be learned from $\mathsf{CT}_{\mathbf{x}}$ and $\mathsf{SK}_f$. Recent years have witnessed significant progress towards constructing functional encryption for advanced functionalities, even from standard assumptions [24, 36, 27, 26, 52, 32, 4, 59, 19, 62, 66, 5, 83, 57, 46, 45, 58]. However, most constructions supporting general functionalities severely restrict the attacker in the security game: she must request only a bounded number of keys [56, 8, 55], or may request unbounded number of keys but from a restricted space[2] [58, 2]. Schemes that may be proven secure against a general adversary are restricted to compute linear or quadratic functions [1, 6, 68, 16].

*Constructing* iO *from* FE. Recent work [10, 23, 11] provided an approach for constructing iO via FE. While we do not have any candidate constructions for FE that satisfy the security and efficiency requirements for constructing iO (except constructions that themselves rely on graded encoding schemes or iO [45, 47]), FE is a primitive that is closer to what cryptographers know to construct and brings iO nearer the realm of reachable cryptography.

An elegant sequence of works [67, 72, 68, 12, 71] has attempted to shrink the functionality of FE that suffices for iO, and construct FE for this minimal functionality

[2] Referred to in the literature as "predicate encryption"

from graded encoding schemes or multilinear maps. Concretely, the question is: what is the smallest $L$ such that FE supporting polynomials of degree $L$ suffices for constructing iO? At a high level, these works follow a two step approach described below:

1. **Bootstrapping FE to iO.** The so called "bootstrapping" theorems have shown that general purpose iO can be built from one of the following: i) sub-exponentially secure FE for $\mathsf{NC}_1$ [10, 23, 11, 21], or ii) sub-exponentially secure FE for $\mathsf{NC}_0$ and PRG in $\mathsf{NC}_0$ [72] iii) PRGs with locality $L$ and FE for computing degree $L$ polynomials [68] or iv) PRGs with *blockwise* locality $L$ and FE for computing degree $L$ polynomials [71].
   At a high level, all bootstrapping theorems make use of *randomized encodings* [61, 14] to reduce computation of a polynomial sized circuit to computation of low degree polynomials.
2. **Instantiating Primitives.** Construct FE supporting degree $L$ polynomials based on graded encodings or multilinear maps [72, 68].

## 1.1 Bootstrapping, the Ideal.

Since we have candidates of FE for quadratic polynomials from standard assumptions on bilinear maps [68, 16], a dream along this line of work would be to reduce the degree required to be supported by FE all the way down to 2, yielding iO, from bilinear maps and other widely believed assumptions (like LWE and PRG with constant locality). The recent work of Lin and Tessaro [71] (LT) came closest to achieving this, by leveraging a new notion of PRG they termed *blockwise local* PRG. A PRG has blockwise locality $L$ and block-size $b$, if when viewing the input seed as a matrix of $b$ rows and $n$ columns, every output bit depends on input bits in at most $L$ columns. As mentioned above, they showed that PRGs with blockwise locality $L$ and certain polynomial stretch, along with FE for computing degree $L$ polynomials and LWE suffice for iO.

Unfortunately, it was shown soon after [73, 18] that PRG with block locality 2 and the stretch required by the LT construction, concretely $\Omega(n \cdot 2^{b(3+\epsilon)})$, do not exist. In the worst case, these lower bounds rule out 2-block local PRG with stretch $\Omega(n \cdot 2^{b(2+\epsilon)})$. On the other hand, these works suggest that 3 block local PRG are likely to exist, thus shrinking the iO-sufficient degree requirement on FE to 3.

While [73, 18] provided strong negative evidence for constructing iO based on 2 block local PRG and hence bilinear maps, they could not rule out the possibility completely; a tantalizing gap has remained. Roughly speaking, the construction of candidate PRG (first suggested by Goldreich [53]) must choose a hyper-graph with variables on vertices, then choose predicates that are placed on each hyper-edge of the graph and output the values of the edge-predicates on the vertex-variables. The lower bounds provided by [73, 18] vary depending on how the graph and predicates are chosen in the above construction: in particular whether the graph is chosen randomly or could be constructed in some arbitrary "worst case" way, whether the predicate is chosen randomly or arbitrarily, and whether the same predicate is used for each hyper-edge or different predicates may be used per hyper-edge or output bit. The following table by [73] summarises our current understanding on the existence of 2 block local PRG:

| Stretch | Worst case versus Random Predicate | Worst case versus Random Graph | Different versus Same Predicate per output bit | Reference |
|---|---|---|---|---|
| $\tilde{\Omega}(n \cdot 2^{b(1+\epsilon)})$ | Random | Random | Different | [18] |
| $\tilde{\Omega}(n \cdot 2^{b(2+\epsilon)})$ | Worst Case | Worst Case | Different | [18] |
| $\tilde{\Omega}(n \cdot 2^{b(1+\epsilon)})$ | Worst Case | Worst Case | Same | [73] |
| $\tilde{\Omega}(n \cdot 2^{b(1+\epsilon)})$ | Worst Case | Worst Case | Different | Open |

As we see in the above table, the existence of 2 block local PRG with carefully chosen graph and predicates with stretch $\tilde{\Omega}(n \cdot 2^{b(1+\epsilon)})$ is open. However, even in the case that these exist, it is not clear whether its even useful, since the Lin-Tessaro compiler requires larger stretch $\Omega(n \cdot 2^{b(3+\epsilon)})$, which is ruled out by row 2 above. In the current version of their paper, [71] remark that "Strictly speaking, our results leave a narrow window of expansion factors open where block-wise PRGs could exist, but we are not aware whether our approach could be modified to use such low-stretch PRGs."

*Bootstrapping: Our Results.* In this work, we show that the narrow window of expansion factors left open by lower bounds *do* suffice for iO. Moreover, we define a larger class of pseudorandomness generators than those considered so far, which may admit lower degree instantiations. We then show that these generators with the same expansion suffice for iO. We discuss each of these contributions below.

*Weakening requirements on PRGs:* We show a new method to construct FE for $\mathsf{NC}_1$ from FE for degree $L$ polynomials, sub-exponentially secure PRGs of block locality $L$ and LWE (or RLWE). Since FE for $\mathsf{NC}_1$ implies iO for P/Poly [10, 23, 21], this suffices for bootstrapping all the way to iO for P/Poly. Our transformation requires the PRG to only have expansion $n \cdot 2^{b(1+\epsilon)}$ which is not ruled out as discussed above. This re-opens the possibility of realizing 2 block local PRG with our desired expansion factor (see below for a detailed discussion), which would imply iO from 2 block local PRG, SXDH on Bilinear maps and LWE. A summary of the state of art in PRG based bootstrapping is provided in Figure 1.1.

*Broadening class of sufficient PRGs*: Our bootstrapping theorem may be instantiated with a broader class of pseudorandom generators than hitherto considered for iO, and may circumvent lower bounds known for the arithmetic degree of iO-sufficient PRGs [73, 18]; in particular, these may admit instantiations with arithmetic degree 2, yielding iO along with the additional assumptions of SXDH on Bilinear maps and LWE. In more detail, we may use the following two classes of PRG:

1. *Non-Boolean PRGs*: We may use pseudorandom generators whose inputs and outputs need not be Boolean but may be integers restricted to a small (polynomial) range. Additionally, the outputs are not required to be pseudorandom but must only

satisfy a milder indistinguishability property[3]. We tentatively propose initializing these PRGs using the multivariate quadratic assumption MQ which has been widely studied in the literature [74, 84, 41] and against the general case of which, no efficient attacks are known.
2. *Correlated Noise Generators*: We introduce an even weaker class of pseudorandom generators, which we call correlated noise generators (CNG) which may not only be non-Boolean but are required to satisfy an even milder (seeming) indistinguishability property.

*Assumptions and Efficiency.* Our bootstrapping theorems can be based on the hardness of LWE or its ring variant RLWE and compiles FE for degree $L$ polynomials directly to FE for $\mathsf{NC}_1$. Our method for bootstrapping to $\mathsf{NC}_1$ does not go via randomized encodings as in previous works. Saving the transformation to randomized encodings makes bootstrapping to $\mathsf{NC}_1$ more efficient than in previous works. For instance, [71] require the encryptor to choose $Q$ PRG seeds, where $Q$ is the (polynomial) length of random tapes needed by the randomized encodings. On the other hand we only need 2 PRG seeds, since we avoid using randomized encodings, yielding a ciphertext that is shorter by a factor of $Q$, as well as (significantly) simpler pre-processing.

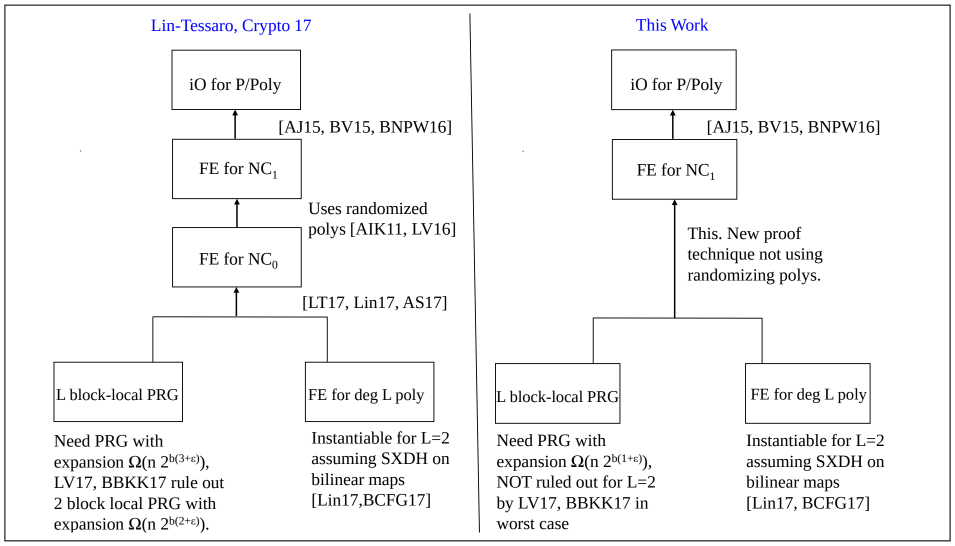
Lin-Tessaro, Crypto 17
iO for P/Poly
[AJ15, BV15, BNPW16]
FE for NC$_1$
Uses randomized polys [AIK11, LV16]
FE for NC$_0$
[LT17, Lin17, AS17]
L block-local PRG
FE for deg L poly
Need PRG with expansion Ω(n 2$^{b(3+ε)}$), LV17, BBKK17 rule out 2 block local PRG with expansion Ω(n 2$^{b(2+ε)}$).
Instantiable for L=2 assuming SXDH on bilinear maps [Lin17,BCFG17]
This Work
iO for P/Poly
[AJ15, BV15, BNPW16]
FE for NC$_1$
This. New proof technique not using randomizing polys.
L block-local PRG
FE for deg L poly
Need PRG with expansion Ω(n 2$^{b(1+ε)}$), NOT ruled out for L=2 by LV17, BBKK17 in worst case
Instantiable for L=2 assuming SXDH on bilinear maps [Lin17, BCFG17]

**Fig. 1.1.** State of the Art in Bootstrapping FE to iO. In the present work, we may bootstrap directly to FE for $\mathsf{NC}_1$ without going through $\mathsf{NC}_0$.

[3] For the knowledgeable reader, we do not require the polynomials computing our PRGs to be sparse and hence the general attack of [18] does not rule out existence of degree 2 instantiations to the best of our knowledge.

## 1.2 Instantiation: the Ideal.

To instantiate $\mathsf{iO}$ via $\mathsf{FE}$ for constant degree polynomials, [71] rely on the $\mathsf{FE}$ for degree $L$ polynomials constructed by Lin [68], which relies on $\mathsf{SXDH}$ on noiseless algebraic multilinear maps of degree $L$, for which no candidates of degree greater than 2 are known to exist. As discussed by [68], instantiating her construction with noisy multilinear maps causes the proof to fail, in addition to the SXDH assumption itself being false on existing noisy multilinear map candidates. We refer the reader to [71, 68] for a detailed discussion.

Evidently, one ideal instantiation for $\mathsf{iO}$ would be to construct noiseless multilinear maps of degree at least 3[4], on which the $\mathsf{SXDH}$ assumption is believed to hold. At the moment, we have no evidence that such objects exist. Another ideal instantiation would be to provide a direct construction of $\mathsf{FE}$ for constant degree polynomials from well understood hardness assumptions, satisfying the requisite compactness properties for implication to $\mathsf{iO}$. Constructing $\mathsf{FE}$ from well-understood hardness assumptions has received significant attention in recent years, and for the moment we do not have any constructions that suffice for $\mathsf{iO}$ excepting those that themselves rely on multilinear maps or $\mathsf{iO}$.

Thus, at present, all concrete instantiations of the $\mathsf{FE}$ to $\mathsf{iO}$ compiler must go via noisy multilinear maps on which $\mathsf{SXDH}$ fails.

*Instantiation: Our Results.* In our work, we take a different approach to the question of instantiation. We propose to construct $\mathsf{FE}$ *directly*, without going through multilinear maps or graded encoding schemes, and use this $\mathsf{FE}$ to instantiate the transformation to $\mathsf{iO}$. We believe this new approach has the following advantages:

1. **May be Simpler**: Construction of $\mathsf{iO}$-sufficient $\mathsf{FE}$ might not need the full power of asymmetric multilinear maps, since $\mathsf{FE}$ is not known to imply asymmetric multilinear maps equipped with $\mathsf{SXDH}$ to the best of our knowledge [5]. Hence, constructing $\mathsf{FE}$ directly may be simpler.
2. **Yield new and possibly more robust assumptions:** Attempts to construct $\mathsf{FE}$ directly for low degree polynomials yield new hardness assumptions which are likely different from current assumptions on noisy multilinear maps. This direction may yield more resilient candidates than those that go via multilinear maps.

In this work, we provide the first direct candidate of symmetric key $\mathsf{FE}$ for constant degree polynomials from new assumptions on lattices. Let $\mathcal{F}$ be the class of circuits with depth $d$ and output length $\ell$. Then, for any $f \in \mathcal{F}$, our scheme achieves $\mathsf{Time}(\mathsf{KeyGen}) = O\big(\mathrm{poly}(\kappa, |f|)\big)$, and $\mathsf{Time}(\mathsf{Enc}) = O(|\mathbf{x}| \cdot 2^d \cdot \mathrm{poly}(\kappa))$ where $\kappa$ is the security parameter. This suffices to instantiate the bootstrapping step above. Our construction is based on the ring learning with errors assumption ($\mathsf{RLWE}$) as well as new untested assumptions on NTRU rings. We provide a detailed security analysis and discuss why currently known attacks in the multilinear map setting do not appear

[4] Ideally degree 5, so as to remove the reliance on even blockwise local $\mathsf{PRG}$ and rely directly on 5 local $\mathsf{PRG}$ which are better understood.

[5] A line of work can traverse the route of $\mathsf{FE}$ to $\mathsf{iO}$ to $\mathsf{PiO}$ (probabilistic $\mathsf{iO}$) to *symmetric* multilinear maps (see [42] and references therein) using multiple complex subexponential reductions, still not yielding *asymmetric* multilinear maps with $\mathsf{SXDH}$.

to apply. We also provide a proof in a restricted security game where the adversary is allowed to request only one ciphertext, based on a new assumption. While such a security game is too limited to be reasonable, we view this as a first step to provable security. We caution that the assumptions underlying our construction must yet be subject to rigorous cryptanalysis by the community. However, our approach is fundamentally different and we hope it inspires other candidates.

## 1.3 Our Techniques: Bootstrapping

Let us start by restating the goal: we wish to construct $\mathsf{FE}$ for the function class $\mathsf{NC}_1$ such that the size of the ciphertext depends only sublinearly on the size of the function. Previous work [10, 23] shows that such an $\mathsf{FE}$ suffices to construct $\mathsf{iO}$. At a high level, to compute $f(\mathbf{x})$, our $\mathsf{FE}$ scheme will make use of a fully homomorphic encryption scheme ($\mathsf{FHE}$) to evaluate the function $f$ on the $\mathsf{FHE}$ ciphertext $\mathsf{CT}_{\mathbf{x}}$ of $\mathbf{x}$ to obtain a "functional" ciphertext $\mathsf{CT}_{f(\mathbf{x})}$ and then perform $\mathsf{FHE}$ decryption on $\mathsf{CT}_{f(\mathbf{x})}$ to obtain $f(\mathbf{x})$.

Agrawal and Rosen [8] show how to instantiate the above blueprint from the $\mathsf{LWE}$ assumption, but incur large ciphertext size that does not suffice for bootstrapping to $\mathsf{iO}$. Their construction is the starting point of our work. Below, we assume familiarity of the reader with $\mathsf{RLWE}$ and Regev's public key encryption scheme [79, 52]. Although our bootstrapping can also be based on standard $\mathsf{LWE}$, we describe it using $\mathsf{RLWE}$ here since it is simpler.

*"$\mathsf{FE}$-compatible" Homomorphic Encryption by [8].* The main technical contribution of [8] may be seen as developing a special "$\mathsf{FE}$-compatible" $\mathsf{FHE}$ scheme that lends itself to the *constrained* decryption required by $\mathsf{FE}$. Note that $\mathsf{FHE}$ enables an evaluator to compute arbitrary functions on the ciphertext. In contrast, $\mathsf{FE}$ requires that given a a ciphertext $\mathsf{CT}_{\mathbf{x}}$, decryption is constrained to some function $f$ for which the decyptor possess a secret key $\mathsf{SK}_f$. Thus, decryption must reveal $f(\mathbf{x})$ alone, and leak no other function of $\mathbf{x}$.

To address this issue, [8] design new algorithms for $\mathsf{FHE}$ encryption and ciphertext evaluation, inspired by an $\mathsf{FHE}$ by Brakerski and Vaikuntanathan [29]. The evaluator/decryptor, given the encoding of some input $\mathbf{x}$ and some (arithmetic) circuit $f \in \mathsf{NC}_1$ can execute the ciphertext evaluation algorithm, which we denote by $\mathsf{Eval}_{\mathsf{CT}}$, to compute a "functional" ciphertext $\mathsf{CT}_{f(\mathbf{x})}$ that encodes $f(\mathbf{x})$. The functional ciphertext can then by decrypted by $\mathsf{SK}_f$ alone, to reveal $f(\mathbf{x})$ and nothing else.

The encryption algorithm of [8] is "levelled" in that given input $\mathbf{x}$, it outputs a set of encodings $\mathcal{C}^i$ for $i \in [d]$ where $d$ is the depth of the circuit being computed. The functional ciphertext $\mathsf{CT}_{f(\mathbf{x})} = \mathsf{Eval}_{\mathsf{CT}}(\underset{i\in[d]}{\cup}\mathcal{C}^i, f)$ of [8] has the following useful structure:

$$\mathsf{CT}_{f(\mathbf{x})} = \langle \mathsf{Lin}_f,\ \mathcal{C}^d \rangle + \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1})$$

for some $f$-dependent linear function $\mathsf{Lin}_f$ and polynomial $\mathsf{Poly}_f$. Moreover, upon decrypting $\mathsf{CT}_{f(\mathbf{x})}$, we get

$$f(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})} = \langle \mathsf{Lin}_f,\ \mathcal{M}^d \rangle + \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) \tag{1.1}$$

where $\mathcal{M}^d$ is the message vector encoded in level $d$ encodings $\mathcal{C}^d$. Here, $f(\mathbf{x}) \in R_{p_0}$ for some ring $R_{p_0}$ and $\mathsf{noise}_{f(\mathbf{x})}$ is the noise term that results from FHE evaluation which may be removed using standard techniques to recover $f(\mathbf{x})$ as desired.

*Using Linear* FE *and Noise Flooding.* Given the above structure, an approach to compute $f(\mathbf{x})$ is to leverage functional encryption for linear functions [1, 6], denoted by LinFE to compute the term $\langle \mathsf{Lin}_f,\ \mathcal{M}^d \rangle$. Recall the functionality of LinFE: the encryptor provides a ciphertext $\mathsf{CT}_{\mathbf{z}}$ for some vector $\mathbf{z} \in R^n$, the key generator provides a key $\mathsf{SK}_{\mathbf{v}}$ for some vector $\mathbf{v} \in R^n$ and the decryptor learns $\langle \mathbf{z}, \mathbf{v} \rangle \ \in R$. Thus, we may use LinFE to enable the decryptor to compute $\langle \mathsf{Lin}_f,\ \mathcal{M}^d \rangle$, let the decryptor compute $\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1})$ herself to recover $f(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})}$. Surprisingly, constrained decryption of a *linear* function on secret values suffices, along with additional public computation, to perform constrained decryption of a function in $\mathsf{NC}_1$.

Unfortunately, this approach is insecure as is, as discussed in [8]. For bounded collusion FE, the authors achieve security by having the encryptor encode a fresh, large noise term $\mathsf{noise}_{\mathsf{fld}}$ for each requested key $f$ which "floods" $\mathsf{noise}_{f(\mathbf{x})}$. This noise is forcibly added to the decryption equation so that the decryptor recovers $f(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})} + \mathsf{noise}_{\mathsf{fld}}$, which by design is statistically indistinguishable from $f(\mathbf{x}) + \mathsf{noise}_{\mathsf{fld}}$. [8] show that with this modification the scheme can be shown to achieve strong simulation style security, by relying just on security of LinFE. However, encoding a fresh noise term per key causes the ciphertext size to grow at least linearly with the number of function keys requested, or in the case of single key FE, with the output length of the function. As noted above, this renders their FE insufficient for iO.

*Noisy Linear Functional Encryption.* In this work, we show that the approach of [8] can be extended to construct a single key FE for $\mathsf{NC}_1$ with ciphertext size sublinear in the output length, by replacing linear functional encryption LinFE with *noisy linear functional encryption*, denoted by NLinFE. Noisy linear functional encryption is like like regular linear functional encryption [1, 6], except that the function value is recovered only up to some bounded additive error/noise, 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 $\mathsf{CT}_{\mathbf{z}}$ which encodes vector $\mathbf{z} \in R^n$ and a secret key $\mathsf{SK}_{\mathbf{v}}$ which encodes vector $\mathbf{v} \in R^n$, 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.

Let $f \in \mathsf{NC}_1$ and let the output of $f$ be of size $\ell$. Let $f_1, \ldots, f_\ell$ be the functions that output the $i^{th}$ bit of $f$ for $i \in [\ell]$. At a high level, our FE for $\mathsf{NC}_1$ will enable the decryptor to compute $\langle \mathsf{Lin}_{f_i},\ \mathcal{M}^d \rangle + \mathsf{noise}_{\mathsf{fld}\,i}$ as in [8] but instead of having the encryptor encode $\ell$ noise terms during encryption, we use NLinFE to compute and add noise terms $\mathsf{noise}_{\mathsf{fld}\,i}$ into the decryption equation. Given NLinFE with sublinear ciphertext size, we can then construct FE for $\mathsf{NC}_1$ with sublinear ciphertext size, which suffices for bootstrapping to iO. In more detail, we show:

**Theorem 1.1.** *(Informal) There exists an* FE *scheme for the circuit class* $\mathsf{NC}_1$ *with sublinear ciphertext and satisfying indistinguishability based security, assuming:*

– *A noisy linear FE scheme* NLinFE *with sublinear ciphertext satisfying indistinguishability based security.*
– *The Learning with Errors (*LWE*) Assumption.*
– *A pseudorandom generator (*PRG*).* [6]

The formal theorem is provided in Section 4. Note that while [8] argue simulation based security of FE for $\mathsf{NC}_1$ using simulation security of LinFE in the bounded key setting, we must argue *indistinguishability* based security of FE for $\mathsf{NC}_1$ assuming indistinguishability based security NLinFE. This is significantly more complex and requires new proof techniques, which we develop in this work. Please see Section 4 for details.

The key question that remains is how does NLinFE construct the noise term to be added to the decryption equation? As discussed next, NLinFE is a primitive flexible enough to admit multiple instantiations, which in turn yield FE for $\mathsf{NC}_1$ from diverse assumptions, improving the state of art.

*Constructing* NLinFE. Next, we discuss multiple methods to construct NLinFE, which imply FE for $\mathsf{NC}_1$ from various assumptions. Together with the bootstrapping of NLinFE to FE for $\mathsf{NC}_1$ described above, this suffices for applying the FE to iO compiler of [10, 23]. Before we describe our constructions, we provide a summary via the following theorem:

**Theorem 1.2.** *(Informal) Noisy linear functional encryption (*NLinFE*) with sublinear ciphertext and satisfying indistinguishability based security may be constructed using:*

1. *i) An* FE *scheme supporting evaluation of degree $L$ polynomials and satisfying indistinguishability based security, ii) sub-exponentially secure* PRG *with block locality $L$ and expansion $n \cdot 2^{b(1+\epsilon)}$, where $n$ is the input length and $b$ is the block length, and iii)* LWE *(or* RLWE*).*
2. *i) An* FE *scheme supporting evaluation of degree $L$ polynomials and satisfying indistinguishability based security, ii) sub-exponentially secure weak randomness generators called "Correlated Noise Generators" (*CNG*) iii)* LWE *(or* RLWE*).*
3. *i) An* FE *scheme supporting evaluation of degree $L$ polynomials and satisfying indistinguishability based security, ii) sub-exponentially secure weak randomness generators called "non-Boolean* PRG*" iii)* LWE *(or* RLWE*).*
4. *New lattice assumptions on NTRU rings.*

Note that the above instantiations of NLinFE have several desirable features as discussed below:

1. The first instantiation uses a block local PRG with smaller expansion factor than that required by the Lin-Tessaro compiler [71]. More importantly, 2-local PRG with the above expansion factor is not ruled out in the worst case by [18, 73]. Thus, if PRG with block locality 2 and the above expansion factor exist, we may instantiate FE for $L = 2$ using SXDH on bilinear maps [68].

[6] We actually only need a randomness generator that is weaker than a standard PRG but do not discuss this here. for the formal statement.

2. The notion of correlated noise generators CNG is new to our work, and appears significantly weaker than a standard Boolean PRG, as discussed below. Prior to our work, the only notions of PRG that were used to construct iO are standard Boolean PRG and block local PRG [71].
3. Our notion of non-Boolean PRG interpolates CNG and standard Boolean PRG. This notion is qualitatively the same as the notion of $\Delta$-RG discovered concurrently and independently by [9].
4. Our direct construction of NLinFE makes no use of bi/multi-linear maps, and provides a candidate for post quantum iO. We note that most (if not all) candidates of iO are vulnerable in the post quantum setting [77].

*The "Right" Abstraction.* Noisy linear functional encryption provides the right abstraction (in our opinion) for the smallest functionality that may be bootstrapped to FE for $\mathsf{NC}_1$ using our methods. NLinFE captures the precise requirements on noise that is required for the security of FE for $\mathsf{NC}_1$ and integrates seamlessly with our new proof technique. Moreover, as discussed above, NLinFE may be constructed in different ways from different assumptions, and properties such as ciphertext size and collusion resistance achieved by NLinFE are inherited by FE for $\mathsf{NC}_1$. We remark that the follow up work by Lin and Matt [69] also uses our notion of NLinFE in essentially the same manner for their overall construction.

Put together, an overview of our transformation is provided in Figure 1.2.

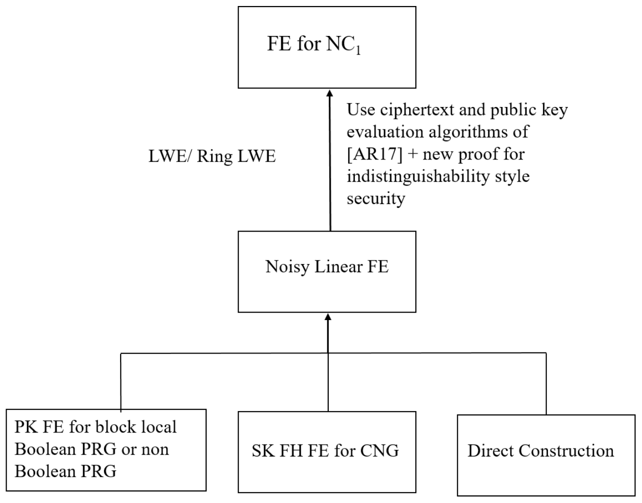
FE for NC$_1$
LWE/ Ring LWE
Use ciphertext and public key evaluation algorithms of [AR17] + new proof for indistinguishability style security
Noisy Linear FE
PK FE for block local Boolean PRG or non Boolean PRG
SK FH FE for CNG
Direct Construction

**Fig. 1.2.** Overview of our transformation. Above, FH refers to function hiding.

**Weaker Requirements on PRG** Next, we discuss the requirements on the PRG used by the first three instantiations of NLinFE discussed above.

$\mathsf{PRG}$ *with smaller expansion factor.* Our first method makes use of a compact $\mathsf{FE}$ scheme which is powerful enough to compute $\mathsf{PRG}$/blockwise local $\mathsf{PRG}$ [71]. Let $\mathsf{PrgFE}$ be a functional encryption scheme that supports evaluation of a $\mathsf{PRG}$ with polynomial stretch. Then, we may construct $\mathsf{NLinFE}$ and hence $\mathsf{FE}$ for $\mathsf{NC}_1$ by leveraging $\mathsf{PrgFE}$ to compute the the noise to be added by $\mathsf{NLinFE}$ as the output of a $\mathsf{PRG}$.

In more detail, by the discussion above, we would like the decryptor to compute:

$$f(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})} + \mathsf{noise}_{\mathsf{fld}} = \langle \mathsf{Lin}_f,\ \mathcal{M}^d \rangle + \mathsf{noise}_{\mathsf{fld}} + \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1})$$

where $\mathsf{noise}_{f(\mathbf{x})} + \mathsf{noise}_{\mathsf{fld}}$ is indistinguishable from $\mathsf{noise}_{\mathsf{fld}}$. Say that the norm of $\mathsf{noise}_{f(\mathbf{x})}$ may be bounded above by value $\mathsf{B}_{\mathsf{nse}}$. Then, it suffices to sample a uniformly distributed noise term $\mathsf{noise}_{\mathsf{fld}}$ of norm bounded by $\mathsf{B}_{\mathsf{fld}}$, where $\mathsf{B}_{\mathsf{fld}}$ is superpolynomially larger than $\mathsf{B}_{\mathsf{nse}}$ for the above indistinguishability to hold. This follows from security of $\mathsf{PRG}$ and a standard statistical flooding argument. We will use $\mathsf{PrgFE}$ to compute $\mathsf{noise}_{\mathsf{fld}}$.

In more detail, let $G$ be a $\mathsf{PRG}$ with polynomial stretch which outputs $\ell$ uniform ring elements of norm bounded by $\mathsf{B}_{\mathsf{fld}}$, and let $G_i$ be the function that selects the $i^{th}$ output symbol of $G$, namely $G_i(\mathsf{seed}) = G(\mathsf{seed})[i]$ where seed is the seed of the $\mathsf{PRG}$. Then,

1. The encryptor may provide $\mathsf{PrgFE}$ encryptions of $(\mathcal{M}^d, \mathsf{seed})$ along with encodings $\bigcup_{i\in[d-1]} \mathcal{C}^i$,
2. The key generator may provide a $\mathsf{PrgFE}$ secret key for polynomial $P_i(\mathbf{z}_1, \mathbf{z}_2) = \langle \mathsf{Lin}_{f_i},\ \mathbf{z}_1 \rangle + G_i(\mathbf{z}_2)$
3. The decryptor may compute $\langle \mathsf{Lin}_{f_i},\ \mathcal{M}^d \rangle + G_i(\mathsf{seed})$ as well as $\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1})$, to recover $f(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})} + G_i(\mathsf{seed})$ by Equation 1.1 as desired.

It is crucial to note that the degree of the polynomial $P_i$ is equal to the degree required to compute $G_i$, because $\mathsf{Lin}_{f_i}$ is a linear function. Moreover, the degree is unchanged even if we make use of a standard $\mathsf{PRG}$ with binary range. To see this, take a binary $\mathsf{PRG}$ that requires degree $L$ to compute, and apply the standard (linear) powers of two transformation to convert binary output to larger alphabet.

Thus, an $\mathsf{FE}$ scheme that supports polynomials of degree $L$, where $L$ is the degree required to compute a $\mathsf{PRG}$, suffices to construct $\mathsf{NLinFE}$ and hence $\mathsf{FE}$ for $\mathsf{NC}_1$. Moreover, we may pre-process the seed of the $\mathsf{PRG}$ as in [71] to leverage "blockwise locality"; our construction allows the $\mathsf{PRG}$ to have smaller expansion factor than that required by the Lin-Tessaro construction, as discussed next.

*Analyzing the Expansion Factor of the* $\mathsf{PRG}$. Above, the polynomial $P_i$ we are required to construct must compute a function that is degree 1 in the $\mathsf{PRG}$ output plus an additional linear function of the encoded messages. By comparison, the underlying degree $L$ $\mathsf{FE}$ in [72, 68, 71] must natively compute a polynomial which has degree 3 in output of a $\mathsf{PRG}$. In more detail, the $\mathsf{FE}$ of [68, 71] must compute a randomizing polynomial [15], which contains terms of the form $r_i r_j r_k$ where $r_i, r_j, r_k$ are random elements, each computed using a $\mathsf{PRG}$. Thus, if $L$ is the locality of the $\mathsf{PRG}$, the total degree of the polynomial is $3L$, which is reduced to $L$ using a clever precomputing trick developed by Lin [68]. In contrast, our $\mathsf{FE}$ must compute a polynomial of the form $\langle \mathsf{Lin}_f,\ \mathcal{M}^d \rangle + \mathsf{PRG}(\mathsf{seed})$ as discussed above, thus natively yielding a polynomial of degree $L$.

Further, Lin and Tessaro [71] construct a method to leverage the blockwise locality $L$ of the $\mathsf{PRG}$, which is believed to be smaller than locality as discussed above. Recall that the $\mathsf{PRG}$ seed is now a matrix of $b$ rows and $n$ columns, and each output bit depends on input bits in at most $L$ columns. Then, to begin, [71] suggest computing all possible monomials in any block, for all blocks, resulting in $O(n \cdot 2^b)$ total monomials. At this point, the output of a $\mathsf{PRG}$ can be computed using a degree $L$ polynomial. However, since the final polynomial has degree 3 in the $\mathsf{PRG}$ outputs, LT further suggest precomputing all degree 3 products of the monomials constructed so far, leading to a total of $O(n \cdot 2^{3b})$ terms. To maintain ciphertext compactness then, the expansion of the $\mathsf{PRG}$ must be at least $\Omega(n \cdot 2^{b(3+\epsilon)})$. We refer the reader to [18, Appendix B] for an excellent overview of the LT construction, and to [71] for complete details.

Since the polynomial in our $\mathsf{FE}$ must compute has degree only $1$ rather than 3 in $\mathsf{PRG}$ output, we need not compute degree 3 products in $O(n \cdot 2^b)$ monomials as required by [71], thus requiring to encode a total number of $O(n \cdot 2^b)$ monomials. Hence, to maintain ciphertext compactness (which is necessary for implication to $\mathsf{iO}$ [10, 23]), the expansion of the $\mathsf{PRG}$ in our case must be $\Omega(n \cdot 2^{b(1+\epsilon)})$. Thus, our transformation requires a lower expansion factor than that of [71]. Moreover, by sidestepping the need to compute randomized encodings, our bootstrapping becomes simpler and more efficient than that of [72, 68, 71].

*Correlated Noise Generators.* As discussed above, $\mathsf{FE}$ for implementing $\mathsf{PRG}$ suffices to construct $\mathsf{NLinFE}$ and hence $\mathsf{FE}$ for $\mathsf{NC}_1$. However, examining carefully the requirements on the noise that must be added to decryption by $\mathsf{NLinFE}$, reveals that using a $\mathsf{PRG}$ to compute the noise is wasteful; a weaker object appears to suffice. Specifically, we observe that the noise terms $\mathsf{noise}_{f_i(\mathbf{x})}$ for $i \in [\ell]$, which must be flooded are low entropy, correlated random variables, constructed as polynomials in $O(L)$ noise terms where $L = |\bigcup_{k\in[d]} \mathcal{C}_k|$. A $\mathsf{PRG}$ mimics $\ell$ i.i.d noise terms where $\ell > L$, i.e. $O(\ell)$ bits of entropy, whereas the random variables that must be flooded have only $O(L)$ bits of entropy.

Indeed, even to flood $\ell$ functions on $L$ noise terms *statistically*, only $L$ fresh noise terms are needed. For instance, let us say that we are required to flood $(f_i(\boldsymbol{\mu}))_{i\in[\ell]}$ for $\boldsymbol{\mu} \in R^L$. Then, it suffices to choose $\boldsymbol{\beta} \in R^L$ such that $\boldsymbol{\beta}$ is superpolynomially larger than $\boldsymbol{\mu}$ to conclude that

$$\mathsf{SD}\Big(\boldsymbol{\beta},\ \boldsymbol{\beta}+\boldsymbol{\mu}\Big) = \mathrm{negl}(\kappa)$$

This implies that

$$\mathsf{SD}\Big(\big(f_1(\boldsymbol{\beta}),\ldots,f_\ell(\boldsymbol{\beta})\big),\ \big(f_1(\boldsymbol{\beta}+\boldsymbol{\mu}),\ldots,f_\ell(\boldsymbol{\beta}+\boldsymbol{\mu})\big)\ \Big) = \mathrm{negl}(\kappa)$$

Considering that we must only generate $O(L)$ bits of pseudoentropy, can we make do with something weaker than a $\mathsf{PRG}$?

To make this question precise, we define the notion of a correlated noise generator, denoted by $\mathsf{CNG}$. A $\mathsf{CNG}$ captures computational flooding of correlated noise, to mimic statistical flooding described above. In more detail, we will use a $\mathsf{CNG}$ to generate flooding terms $g_1(\boldsymbol{\beta}),\ldots,g_\ell(\boldsymbol{\beta})$ such that to any computationally bounded adversary

Adv, it holds that

$$\big(g_1(\boldsymbol{\beta}), \ldots, g_\ell(\boldsymbol{\beta})\big) \overset{c}{\approx} \big((g_1(\boldsymbol{\beta}) + f_1(\boldsymbol{\mu}), \ldots g_\ell(\boldsymbol{\beta}) + f_\ell(\boldsymbol{\mu})\big)$$

Note that if we denote by $g_i$ the function for computing the $i^{th}$ element of the PRG output, then by choosing the range of PRG superpolynomially larger than $|f_i(\boldsymbol{\mu})|$, the above condition is satisfied. Thus, a PRG implies a CNG. However, implication in the other direction does not hold, since CNG only generates strictly fewer bits of pseudoentropy than a PRG.

Moreover, matters are even nicer in the case of CNG, because $g_i$ can be chosen after seeing $f_i$, and the distribution of $\boldsymbol{\mu}$ is known at the time of choosing $g_i$. So each $g_i$ can be different depending on what it needs to "swallow". Additionally, we may leverage the fact that a CNG posits that a distribution must be indistinguishable from *itself* plus a fixed function, not indistinguishable from uniform.

Our hope is that since a CNG appears weaker than a PRG, it may sidestep the lower bounds known for the blockwise-locality of polynomial stretch PRGs, thereby providing a new route to iO from bilinear maps. Suggesting candidates for CNG that have lower degree than PRG is outside the scope of this work but we believe it is useful to identify the weakest object that suffices for bootstrapping to iO. For the precise definition of CNG, please see Section 3.

*Non Boolean* PRG. A notion of randomness generators that interpolates CNG and Boolean PRG is that of non-Boolean PRG, which allows the inputs and outputs to lie in a bounded (polynomial) sized interval over the integers and must only satisfy the computational flooding property described above. Taking a step back, we note that in prior work [68, 71, 12], Boolean PRGs were required in order to compute the binary randomness needed for constructing randomizing polynomials. In our case, the PRG output need not be binary since we do not require these as input to randomized encodings. Additionally, they must satisfy a much weaker property than indistinguishability to uniform i.i.d random variables as discussed above. In more detail, say we can bound $|f_i(\boldsymbol{\mu})| \leq \epsilon$ for $i \in [\ell]$. Then we require the PRG output $G_i(\boldsymbol{\beta})$ to computationally flood $f_i(\boldsymbol{\mu})$ for $i \in [\ell]$, i.e. $G_i(\boldsymbol{\beta}) + f_i(\boldsymbol{\mu})$ must be computationally indistinguishable from $G_i(\boldsymbol{\beta})$.

We note that the above notion of non Boolean PRGs is qualitatively the same as the notion of $\Delta$RGs defined in the concurrent work of Ananth et al. [9] except that $\Delta$RG are weaker, in that they allow the adversary to win the game with $1/\operatorname{poly}$ probability whereas we require that the adversary only wins with standard negligible probability. By relying on the security amplification theorem of [9], our notion can be weakened and made equivalent to $\Delta$RG.

## 1.4 Related Work: Bootstrapping

In this section, we provide a detailed comparison with works that are most closely related to ours.

*Predicate encryption [58, 2, 28] and reusable garbled circuits [54, 2].* A successful approach to constructing functional encryption schemes from standard assumptions is the predicate encryption scheme by Gorbunov et al. [58] and its extensions [2, 28]. Roughly speaking, these schemes make use of an attribute based encryption (ABE) scheme for circuits [57, 25] in conjunction with a fully homomorphic encryption scheme to achieve a system where the input $\mathbf{x}$ is hidden only as long as the adversary's key requests obey a certain "one sided" restriction w.r.t the challenge messages. In more detail, security holds as long as the adversary does not obtain keys $\mathsf{SK}_f$ for any circuit $f$ such that $f(\mathbf{x}) = 0$. Given an adversary who obeys this "one sided" restriction, functionality is general, i.e. the adversary may request for a key corresponding to any polynomial sized circuit. However, in the general "two sided" security game, the schemes are shown to be insecure [58, 2]. The reusable garbled schemes of [54, 2] satisfy general two sided security but do not achieve compact ciphertext required for bootstrapping to $\mathsf{iO}$.

The techniques of the aforementioned line of work and the present work are fundamentally different. While [58, 55] also use FHE in order to hide the attributes in an FE scheme, the building block of ABE necessitates the restriction of one sided security due to the basic structure of ciphertexts and secret keys. As discussed in [2], if the predicate encryption scheme [58] is subject to a general two sided adversary, the adversary may requests keys for related functions which can lead to the recovery of a secret lattice basis, leading to a complete break in security. We emphasize that this attack exploits the structure of the secret keys and ciphertext in the underlying ABE scheme and is in distinct from the attack implied by the leakage of the FHE noise learnt by the attacker upon decryption – indeed, the follow-up work of [28] shows how to construct predicate encryption that does not contain the FHE noise leakage. Thus, despite supporting powerful functionality, current techniques for generalizing ABE to FE get stuck in the quicksand of one sided security.

To overcome this challenge, we insist on a two sided adversary at any cost to functionality. We follow the approach of [8] which starts with the modest functionality of linear functional encryption [1, 6] satisfying two sided security, and makes use of special properties of the FHE scheme of Brakerski and Vaikuntanathan [29] to decompose function computation into a "deep" public computation performed using FHE and a "shallow" (linear) private computation, performed using linear FE [1, 6]. The public FHE computation is performed by the decryptor outside any FE scheme, namely, without any guarantee of constrained decryption. This is in contrast to [58, 2, 28] where the entire function evaluation is performed within the confines of the ABE evaluation, which constrains decryption of the final FHE ciphertext and renders futile any attempts to tamper with the functionality. However, in [8] the only constrained decryption is via the modest functionality of linear FE but the authors argue that constraining a linear function suffices to constrain computation in $\mathsf{NC}_1$, at the cost of non-compact ciphertext.

In the present work, to achieve security as well as succinct ciphertext, we look at the mildest possible strengthening of this functionality, namely one that supports computation of linear functions plus a noise term which satisfies a relatively mild statistical property, as formalized via the notion of noisy linear functional encryption ($\mathsf{NLinFE}$). We then show that this notion of $\mathsf{NLinFE}$ may be bootstrapped all the way to $\mathsf{iO}$. Our bootstrapping uses $\mathsf{NLinFE}$ in a black box way, and when $\mathsf{NLinFE}$ is instantiated using bilinear maps, the results in an interesting "hybrid" scheme which uses FHE to perform deep computation in the clear and then performs a careful FHE decryption in the exponent.

*Comparison with [8].* Even though the present work uses the ciphertext and public key evaluation algorithms developed by Agrawal and Rosen [8], our construction of FE for $\mathsf{NC}_1$ and particularly our proof technique are quite different. Firstly, [8] is in the bounded collusion setting with non-compact ciphertext, and achieves a simulation based security which is known to be impossible in our setting where compact ciphertext is crucial. Hence, we must give an indistinguishability style proof which is significantly harder, and requires using a new proof technique developed in this work. Moreover, [8] adds statistical large flooding noise which is oblivious of the distribution of noise it needs to drown, whereas we will analyze and leverage the distribution carefully. Most importantly, [8] can make do with linear FE whereas we crucially need noisy linear FE[7]. Finally, we give many instantiations of NLinFE using bilinear maps and weak randomness generators as well as directly using new assumptions.

*Independent and concurrent work.* In an independent and concurrent work, Ananth, Jain and Sahai [9] also provide an approach to construct iO without multilinear maps. They rely on (subexponentially-secure) bilinear maps, LWE, block-locality 3 PRGs, and a new type of randomness generator, which they call perturbation resilient generator, denoted as $\Delta$RG. Their techniques and overall construction are extremely different from ours. However, we find it very interesting that both works intersect in identifying a very similar new type of PRG as sufficing to fill the gap between assumptions we believe and iO. Their notion of $\Delta$RG is almost exactly the same as the non-Boolean PRG that (is one of the types of PRGs) we identify – both notions require the generation of some noise $N$ such that $N$ is indistinguishable from $N + e$ for some bounded $e$. However, they only require a weak form of indistinguishability, namely the adversary is allowed to distinguish between $N$ and $N + e$ with $1/\operatorname{poly}$ probability in their case, whereas we require standard negligible distinguishing probability. They also provide a generic security amplification theorem, which transforms FE for $\mathsf{NC}_1$ which satisfies this weak indistinguishability to FE with standard indistinguishability. Their security amplification theorem can be used black box in our construction to also rely on $\Delta$RG (or the weaker notion of CNG) with similar weak indistinguishability.

We may also use their security amplification theorem to weaken the requirement on the underlying quadratic FE scheme so that it can be instantiated using existing constructions [68, 16]. In more detail, we use quadratic FE to compute a noise term which must be natively superpolynomial in size to argue security. However, existing constructions of quadratic functional encryption schemes [68, 16] perform decryption "brute force", by computing a discrete logarithm in the end, restricting the space of decryptable values to be polynomial in size. To align with known constructions of

[7] We remark that a weak version of NLinFE in the bounded collusion setting was developed in an earlier version of [8] (see [7]) but was found to be redundant and was subsequently removed. The current, published version of [8] relies on LinFE alone. Our definition of NLinFE is significantly more general.

quadratic FE, we choose our flooding noise to be polynomial in size – this overcomes the above issue but results in $1/\operatorname{poly}$ advantage to the adversary. This advantage can be made negligible by leveraging the security amplification theorem of [9] in a black box manner.

Aside from significantly different techniques, the final results obtained by the two works are also different. First, we define the abstraction of noisy linear FE, and bootstrap this to $\mathsf{iO}$. The instantiation of noisy linear FE using bilinear maps and $\Delta$RG is only one of the ways of achieving $\mathsf{iO}$; we also define an even weaker type of PRG, namely correlated noise generators (denoted by $\mathsf{CNG}$, please see Section 3) which suffices for $\mathsf{iO}$. On the other hand, their security requirement from their randomness generators is significantly weaker – they only require $1/\operatorname{poly}$ security as discussed above. Moreover, they provide a general security amplification theorem which we do not. The details of the techniques in the two works are vastly different: [9] define and instantiate the notion of tempered cubic encodings which do not have any analogue in our work. Also, we provide a direct construction of $\mathsf{NLinFE}$ from new lattice assumptions, which they do not. In our instantiation that uses bilinear maps, we rely on the $\mathsf{SXDH}$ assumption in the standard model, whereas they argue security in the generic bilinear map model.

We remark that in [9], the special PRG, namely $\Delta$RG needs to be computable by a cubic polynomial that degree $1$ in a public seed component and degree $2$ in the secret seed components. In the present work, as well as [69], the special PRG output must be computed using quadratic polynomials.

*Follow-up work.* In a follow-up work[8], Lin and Matt [69] leverage our techniques to provide a different construction of $\mathsf{iO}$ from bilinear maps, $\mathsf{LWE}$ and weak pseudorandom objects, which they term *Pseudo Flawed-smudging Generators* (PFG). The high level structure of their construction is very similar to ours: they also use special properties of the FHE scheme of Brakerski and Vaikuntanathan [29] to split the functional computation into a deep public computation and a shallow private computation, the former being done by the decryptor in the clear, and the latter being performed in the exponent of a bilinear group using quadratic operations. To argue security, they must, similarly to us, perform noise flooding in the exponent. The main difference from our work is that the choice of noise in our setting is natively super-polynomial as discussed above, and we must use security amplification to make do with polynomial noise. On the other hand, [69] can make do with polynomial noise via their notion of Pseudo Flawed-smudging Generators (PFG). We remark that in contrast to the present work, [69] construct FE for $\mathsf{NC}_0$ and then rely on randomized encodings to bootstrap this to FE for $\mathsf{NC}_1$, as in prior work [72, 68, 71]. On the other hand, we use techniques from [29] in a non blackbox way to bootstrap all the way to $\mathsf{NC}_1$ directly.

## 1.5 Our Techniques: Direct Construction of NLinFE

Next, we provide a direct construction of $\mathsf{NLinFE}$ based on new (untested) assumptions that strengthen ring learning with errors ($\mathsf{RLWE}$) and $\mathsf{NTRU}$. Our construction is quite

[8] We had shared an earlier version with the authors several months ago.

different from known constructions and does not rely on multilinear maps or graded encoding schemes.

As discussed above, flooding correlated noise terms appears qualitatively easier than generating uniform pseudorandom variables. Recall that $\ell$ is the output length of the function and $L$ is sublinear in $\ell$. In this section, we discuss a method to provide $L$ encodings of a seed vector $\boldsymbol{\beta}$ in a way that the decryptor can compute $\ell$ encodings of $\{g_i(\boldsymbol{\beta})\}_{i\in[\ell]}$ on the fly. The careful reader may suspect that we are going in circles: if we could compute encodings of $g_i(\boldsymbol{\beta})$ on the fly, could we not just compute encodings of $f_i(\mathbf{x})$ on the fly?

We resolve this circularity by arguing that the demands placed on noise in lattice based constructions are significantly weaker than the demands placed on messages. In particular, while computation on messages must maintain integrity, noise need only create some perturbation, the exact value of this perturbation is not important. Therefore if, in our attempt to compute an encoding $g_i(\boldsymbol{\beta})$, we instead compute an encoding of $g_i'(\boldsymbol{\beta}')$, this still suffices for functionality. Intuitively $g_i'(\boldsymbol{\beta}')$ will be a polynomial equation of $\boldsymbol{\beta}$ designed to flood $f_i(\boldsymbol{\mu})$.

In order to construct FE that supports the computation of noisy linear equations, we begin with an FE that supports computation of linear equations, denoted by LinFE, provided by [1, 6]. All our constructions use the blueprint provided in [6] to support linear equations, and develop new techniques to add noise. In order to interface with the LinFE construction of [6], we are required to provide encodings of noise terms $\boldsymbol{\beta}$ such that:

1. Given encodings of $\boldsymbol{\beta}$ and $g_i$ the decryptor may herself compute on these to construct an encoding of $g_i(\boldsymbol{\beta})$.
2. The functional encoding of $g_i(\boldsymbol{\beta})$ must have the form $h_{g,i}\cdot s + \mathsf{noise}_{g,i} + g_i(\boldsymbol{\beta})$ where $h_{g,i}$ *is computable by the key generator given only the public/secret key and the function value. In particular,* $h_{g,i}$ *should not depend on the ciphertext.*

In order to compute efficiently on encodings of noise, we introduce a strengthening of the RLWE and NTRU assumptions. 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, the following assumptions are necessary (but not sufficient) for security of our scheme:

1. *We assume that the* NTRU *assumption holds even if multiple samples have the same denominator*. This assumption has been discussed by Peikert [76, 4.4.4], denoted as the *NTRU learning problem* and is considered a reasonable assumption. Moreover, this assumption is also used in the multilinear map constructions [44] and has never been subject to attack despite extensive cryptanalysis.
   In more detail, for $i \in \{1,\ldots,w\}$, sample $f_{1i}, f_{2i}$ and $g_1, g_2$ from a discrete Gaussian over ring $R$. If $g_1, g_2$ are not invertible over $R_{p_2}$, resample. Set
   $$h_{1i} = \frac{f_{1i}}{g_1}, \quad h_{2i} = \frac{f_{2i}}{g_2} \in R_{p_2}$$
   We assume that the samples $\{h_{1i}, h_{2j}\}$ for $i,j \in [w]$ are indistinguishable from random. Note that NTRU requires the denominator to be chosen afresh for each

sample, i.e. $h_{1i}$ (resp. $h_{2i}$) should be constructed using denominator $g_{1i}$ (resp. $g_{2i}$), for $i \in [w]$.

2. *We assume that* RLWE *with small secrets remains secure if the noise terms in* RLWE *samples live in some secret ideal.* In more detail, for $i \in [w]$, let $\widehat{\mathcal{D}}(\Lambda_2), \widehat{\mathcal{D}}(\Lambda_1)$ be discrete Gaussian distributions over lattices $\Lambda_2$ and $\Lambda_1$ respectively. Then, sample

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

Above, small is a place-holder term that implies the norm of the relevant element can be bounded well below the modulus size, $p_2/5$, say. We use it for intuition when the precise bound on the norm is not important. Hence, 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

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

We assume that the elements $d_{1i}, d_{2j}$ for $i, j \in [w]$ are pseudorandom. The powerful property that this assumption provides is that the product of the samples $d_{1i} \cdot d_{2j}$ do not suffer from large cross terms for any $i, j \in [w]$ – since the error of one sample is chosen to annihilate with the large element of the other sample, the product yields a well behaved RLWE sample whose label is a product of the original labels. In more detail,

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

If we treat each $d_{1i}, d_{2j}$ as an RLWE sample, then we may use these samples to encode noise terms so that direct multiplication of samples is well behaved. Note that the noise terms we wish to compute on, are the messages encoded by the "RLWE sample" $d_{1i}$, hence $d_{1i}$ must contain two kinds of noise: the noise required for RLWE security and the noise that behaves as the encoded message. This requires some care, but can be achieved by nesting these noise terms in different ideals as:

$$d_{1i} = h_{1i} \cdot t_1 + p_1 \cdot \tilde{e}_{1i} + p_0 \cdot e_{1i} \in R_{p_2}$$
$$d_{2i} = h_{2i} \cdot t_2 + p_1 \cdot \tilde{e}_{2i} + p_0 \cdot e_{2i} \in R_{p_2}$$

Here, $(p_1 \cdot \tilde{e}_{1i},\ p_1 \cdot \tilde{e}_{2i})$ behave as RLWE noise and $(p_0 \cdot e_{1i},\ p_0 \cdot e_{2i})$ behave as the encoded messages. Both $\tilde{e}_{1i}, e_{1i}$ as well as $\tilde{e}_{2i}, e_{2i}$ are chosen from special ideals as before. Now, we may compute quadratic polynomials on the encodings "on-the-fly" as $\sum_{i,j} d_{1i} d_{2j}$ to obtain a structured-noise RLWE sample whose label is computable by the key generator. If we treat this dynamically generated encoding as an RLWE encoding of correlated noise, then we can use this to add noise to the NLinFE decryption equation by generalizing techniques from [6]. The decryptor can, using all the machinery developed so far, recover $f_i(\mathbf{x}) + \mathsf{noise}_{f(\mathbf{x})} + \mathsf{noise}_{\mathsf{fld}\,i}$ where $\mathsf{noise}_{\mathsf{fld}\,i}$ is constructed as a quadratic polynomial of noise terms that live in special ideals.

*Mixing Ideals.* While it suffices for functionality to choose the correlated noise term as a polynomial evaluated on noise living in special ideals, the question of security is more worrisome. By using the new "on-the-fly" encodings of noise, the decryptor recovers noise which lives in special, secret ideals, and learning these ideals would compromise security. In more detail, the noise term we constructed above is a random linear combination of terms $(g_1 \cdot g_2)$, $\{f_{1i}\}_i$, $\{f_{2j}\}_j$, which must be kept secret for semantic security of $d_{1i}, d_{2j}$ to hold. Indeed, if we over-simplify and assume that the attacker can recover noise terms that live in the ideal generated by $g_1 \cdot g_2$, then recovering $g_1 \cdot g_2$ from these terms becomes an instance of the *principal ideal problem* [50, 40].

While the principal ideal problem has itself resisted efficient classical algorithms so far, things in our setting can be made significantly better by breaking the ideal structure using additional tricks. We describe these next.

1. *Mixing ideals.* Instead of computing a single set of pairs $\Big( \big(h_{1i}, d_{1i}\big), \big(h_{2i}, d_{2i}\big) \Big)$, we now compute $k$ of them, for some polynomial $k$ fixed in advance. Thus, we sample $f^j_{1i}, f^j_{2i}$ and $g^j_1, g^j_2$ for $i \in \{1, \ldots, w\}$, $j \in \{1, \ldots, k\}$, where $w, k$ are fixed polynomials independent of function output length $\ell$, and set
$$h^j_{1i} = \frac{f^j_{1i}}{g^j_1}, \quad h^j_{2i} = \frac{f^j_{2i}}{g^j_2} \in R_{p_2}$$
The encoding of a noise term constructed corresponding to the $(i,j)^{th}$ monomial is $d_{ii'} = \sum\limits_{j \in [k]} d_{1i} d_{2i'}$. Thus, the resultant noise term that gets added to the decryption equation looks like:
$$p_0 \cdot \left[ \sum_{j \in [k]} \left( g^j_2 \cdot g^j_1 \cdot \Big( p_0 \cdot (\xi^j_{1i} \cdot \xi^j_{2i'}) \Big) + \Big( f^j_{1i} \cdot \xi^j_{2i'} \cdot t_1 + f^j_{2i'} \cdot \xi^\ell_{1i} \cdot t_2 \Big) \right) \right] \quad (1.2)$$
Thus, by adding together noise terms from multiple ideals, we "spread" it out over the entire ring rather than restricting it to a single secret ideal. Also, we note that it is only the higher degree noise terms that must live in special ideals; if the polynomial contains linear terms, these may be chosen from the whole ring without any restrictions. In more detail, above, we computed a noise term corresponding to a quadratic monomial which required multiplying and summing encodings. If we modify the above quadratic polynomial to include a linear term, we will need to add a degree 1 encoding into the above equation. The degree 1 encoding which does not participate in products, may encode noise that is chosen without any restrictions, further randomizing the resultant noise.
2. *Adding noise generated collectively by ciphertext and key.* Aside from computing polynomials over structured noise terms encoded in the ciphertext, we suggest an additional trick which forces noise terms into the decryption equation. These noise terms are quadratic polynomials where each monomial is constructed jointly by the encryptor and the key generator. This trick relies on the structure of the key

and ciphertext in our construction. We describe the relevant aspects of the key and ciphertext here. The key for function $f_i$ is a short vector $\mathbf{k}$ such that:

$$\langle \mathbf{w},\ \mathbf{k}\rangle = u_{f_i}$$

where $\mathbf{w}$ is part of the master secret, and $u_{f_i}$ is computed by the key generator. The encryptor provides an encoding

$$\mathbf{c} = \mathbf{w} \cdot s + p_1 \cdot \mathsf{noise}_0$$

As part of decryption, the decryptor computes $\langle \mathbf{k}, \mathbf{c}\rangle$ to obtain $u_{f_i} \cdot s + p_1 \cdot \langle \mathbf{k},\ \mathsf{noise}_0\rangle$. Moreover by running $\mathsf{Eval}_{\mathsf{CT}}(\mathcal{C}^1, \ldots, \mathcal{C}^d)$, she also obtains $u_{f_i} \cdot s + f(\mathbf{x}) + p_0 \cdot \mathsf{noise} + p_1 \cdot \mathsf{noise}'$. Subtracting these and reducing modulo $p_1$ and then modulo $p_0$ yields $f(\mathbf{x})$ as desired.
Intuitively, the structured noise computed above is part of the noise in the sample computed by $\mathsf{Eval}_{\mathsf{CT}}$, i.e. part of $\big(\ p_0 \cdot \mathsf{noise} + p_1 \cdot \mathsf{noise}'\ \big)$ in the notation above. Our next trick shows how to add noise to $\langle \mathbf{k}, \mathbf{c}\rangle$.
We modify $\mathsf{KeyGen}$ so that instead of choosing a single $\mathbf{k}$, it now chooses a pair $(\mathbf{k}^1, \mathbf{k}^2)$ such that:

$$\begin{aligned}\langle \mathbf{w},\ \mathbf{k}^1\rangle &= u_{f_i} + p_0 \cdot \Delta^1 + p_1 \cdot \tilde{\Delta}^1\\ \langle \mathbf{w},\ \mathbf{k}^2\rangle &= u_{f_i} + p_0 \cdot \Delta^2 + p_1 \cdot \tilde{\Delta}^2\end{aligned}$$

Here, $\Delta^1$, $\Delta^2$, $\tilde{\Delta}^1$, $\tilde{\Delta}^2$ are discrete Gaussians sampled by the key generator *unique to the key for $f_i$*. Additionally, the encryptor splits $\mathbf{c}$ as:

$$\begin{aligned}\mathbf{c}_{01} &= \mathbf{w} \cdot s_1 + p_1 \cdot \boldsymbol{\nu}_1\\ \mathbf{c}_{02} &= \mathbf{w} \cdot s_2 + p_1 \cdot \boldsymbol{\nu}_2\end{aligned}$$

where $s_1 + s_2 = s$ and $s_1, s_2$ are small, then,

$$\begin{aligned}\langle \mathbf{k}^1,\ \mathbf{c}_{01}\rangle + \langle \mathbf{k}^2,\ \mathbf{c}_{02}\rangle = u_{f_i} \cdot s + p_0 \cdot \big(\Delta^1 \cdot s_1 + \Delta^2 \cdot s_2\big)\\ + p_1 \cdot \big(\tilde{\Delta}^1 \cdot s_1 + \tilde{\Delta}^2 \cdot s_2\big) + p_1 \cdot \mathsf{noise}\end{aligned}$$

Thus, we forced the quadratic polynomial $p_0 \cdot \big(\Delta^1 \cdot s_1 + \Delta^2 \cdot s_2\big) + p_1 \cdot \big(\tilde{\Delta}^1 \cdot s_1 + \tilde{\Delta}^2 \cdot s_2\big)$ into the noise, where $\Delta^1, \Delta^2$ and $\tilde{\Delta}^1, \tilde{\Delta}^2$ are chosen by the key generator for the particular key request and the terms $s_1$ and $s_2$ are chosen by the encryptor unique to that ciphertext. Note that $\mathbf{w}$ can be hidden from the view of the adversary since it is not required for decryption, hence the adversary may not compute $\langle \mathbf{w},\ \mathbf{k}^1\rangle$, $\langle \mathbf{w},\ \mathbf{k}^2\rangle$ in the clear. For more details, please see the full version [3].

## 1.6 Related Work: Instantiation

To the best of our knowledge, all prior work constructing FE for degree $L \geq 3$ polynomials rely on either $\mathsf{iO}$ itself [45] or multilinear maps [48] or bilinear maps and weak pseudorandomness [12, 68, 71] as discussed above. Since our direct construction also makes use of NTRU lattice assumptions, we discuss here some high level differences between multilinear map based approaches and our approach.

Let us describe the main ideas behind the multilinear map construction of [44]. Our description follows the summary of [65]. Similarly to us, the authors consider the polynomial rings $R = \mathbb{Z}[x]/\langle x^n + 1\rangle$ and $R_q = R/qR$. They generate a small secret $g \in R$ and set $I = \langle g \rangle$ to be the principal ideal over $R$ generated by $g$. Next, they sample a uniform $z \in R_q$ which stays secret. The "plaintext" is an element of $R/I$, and is encoded via a division by $z$ in $R_q$: to encode a coset of $R/I$, give element $[c/z]_q$ where $c$ is an arbitrary small coset representative. Since $g$ is hidden, the authors provide another public parameter $y$, which is an encoding of 1 and the encoding of the coset is chosen as $[e \cdot y]_q$ where $e$ is a small coset representative. At level $i \neq 1$, the encoding has the form $[c/z^i]_q$. The encodings are additively and multiplicatively homomorphic, and for testing whether an element in the last level $D$ (say) encodes 0, the authors provide a "zero test parameter" $p_{zt} = hz^D g^{-1} \mod q$ where $h$ is an element of norm approximately $\sqrt{q}$. The parameters are set so that if an element encodes 0, its product with this parameter is "small" otherwise it is "large".

Known attacks against multilinear maps and obfuscators operate on the following broad principle: i) perform algebraic manipulations on some initial encodings, ii) apply the zero test to each top level encoding, iii) perform an algebraic computation on the results of the zero testing so as to obtain an element in the ideal $\langle g \rangle$, iv) use this somehow break the scheme. once an element in $\langle g \rangle$ is obtained, different attacks work in different ways, but in the "weak multilinear map model" [49], being able to recover an element in $\langle g \rangle$ is considered a successful attack. Thus, the unique element $g$ must crucially be kept secret.

In our work, decryption of the FE scheme also results in a high degree polynomial containing secret elements $f_{1i}g_1, f_{2i}g_2$ for $i \in [\text{poly}]$ along with fresh random elements per ciphertext. However, unlike the multilinear map template where there is a single secret $g$, there are a polynomial number of secret elements that play (what appears to us) qualitatively the same role as $g$ in our construction. Moreover, these are "spread out" in the recovered polynomial which makes obtaining any term isolating any one secret element via algebraic manipulations seem improbable. Additionally, annihilation attacks [75] crucially make use of the fact that the unstructured elements that are unique to every encoding are *linear*, which assists in the computation of the annihilation polynomial. In contrast, unstructured elements in our recovered polynomial that are unique to the encoding are high degree and seem much harder to annihilate.

Our construction of NLinFE appears much simpler in design than the construction of multilinear map based obfuscators, we refer the reader to [75, 78] for a clean description of an abstract obfuscator. Unlike current candidate obfuscators, we do not need to use straddling sets for handling mixed input attacks, eliminating a vulnerability recently exploited by [78]. This is because mixed input attacks seem very hard to launch in our construction, since we do not use branching programs and all parts of a given input are tied together using an LWE secret (albeit with a non-standard LWE assumption). Moreover, the function keys in our FE construction have a different structure than the ciphertext and do not seem amenable to mix and match attacks.

## 2 Noisy Linear Functional Encryption

We refer the reader to the full version [3] for definitions and preliminaries. In this section, we define the notion of noisy linear functional encryption. At a high level, noisy linear functional encryption is like regular linear functional encryption [1, 6], except that the function value is recovered only up to some bounded additive error (which we informally call noise), and indistinguishability holds even if the challenge messages evaluated on all the function keys are only "approximately" equal, i.e. they differ by an additive term of low norm.

*On the noise added by* $\mathsf{NLinFE}$. Recall from Section 1.3, that $\mathsf{NLinFE}$ must add a noise term $\mathsf{noise}_{\mathsf{fld}}$ which "floods" the noise term $\mathsf{noise}_{f(\mathbf{x})}$ for security. Also recall that in our setting, $\mathsf{noise}_{f(\mathbf{x})}$ is the noise term that results from evaluating the circuit $f$ on the $\mathsf{FHE}$ encodings of $\mathbf{x}$. We denote by $\mathcal{D}$ the distribution from which the noise terms in $\mathsf{FHE}$ are sampled and by $\mathcal{F}$ the class of circuits that are used to compute on the $\mathsf{FHE}$ noise terms, resulting in $\mathsf{noise}_{f(\mathbf{x})}$. Thus, $\mathsf{NLinFE}$ must add a noise term that wipes out the leakage resulting from the adversary learning $\mathsf{noise}_{f(\mathbf{x})}$. In general, $\mathcal{F}$ represents the class of functions that $\mathsf{NLinFE}$ can be used to bootstrapped to. In more detail, if $\mathcal{F} = \mathsf{NC}_1$, $\mathsf{NLinFE}$ enables bootstrapping to $\mathsf{FE}$ for $\mathsf{NC}_1$, whereas when $\mathcal{F} = \mathsf{NC}_0$, it enables bootstrapping to $\mathsf{FE}$ for $\mathsf{NC}_0$.

*Definition of* $\mathsf{NLinFE}$. In our constructions, $R$ is 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 $R_q = R/qR$ for some prime $q$. Let $\mathcal{D}$ be a distribution over $R$, $\mathcal{F}$ be a class of functions $\mathcal{F} : R^\ell \to R$ and $B \in \mathbb{R}^+$ a bounding value on the norm of the decryption error. In general, we require $B << q$. We are ready for the formal definition.

**Definition 2.1.** *A* $(\mathcal{D}, \mathcal{F}, B)$*-noisy linear functional encryption scheme* $\mathsf{FE}$ *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:*

- $\mathsf{FE.Setup}(1^\kappa, R_q^\ell)$ *takes as input the security parameter* $\kappa$ *and the space of message and function vectors* $R_q^\ell$ *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* $\mathsf{MSK}$ *and a vector* $\mathbf{v} \in R_q^\ell$ *and outputs the secret key* $\mathsf{SK}_\mathbf{v}$.
- $\mathsf{FE.Enc}(\mathsf{PK}, \mathbf{z})$ *takes as input the public key* $\mathsf{PK}$ *and a message* $\mathbf{z} \in R_q^\ell$ *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* $y \in R_q \cup \{\perp\}$.

**Definition 2.2 ( Approximate Correctness).** *A noisy linear functional encryption scheme* $\mathsf{FE}$ *is correct if for all* $\mathbf{v}, \mathbf{z} \in R_q^\ell$,

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

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

*Security.* Next, we define the notion of Noisy-IND security.

**Definition 2.3** **(**Noisy-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* $\mathbf{v}_i \in R_q^\ell$ *for* $i \in [k]$ *for some polynomial* $k$*. Along with* $\mathbf{v}_i$*,* Adv *also submits a function* $f_i \in \mathcal{F}$ *which must satisfy some constraints discussed later. In response,* Adv *is given the corresponding keys* $\mathsf{SK}(\mathbf{v}_i)$*.*
3. **Challenge Ciphertexts:** Adv *outputs the challenge message pairs* $(\mathbf{z}_0^i, \mathbf{z}_1^i) \in R_q^\ell \times R_q^\ell$ *for* $i \in [Q]$*, where* $Q$ *is some polynomial, to the challenger. Along with the challenger pair* $(\mathbf{z}_0^i, \mathbf{z}_1^i)$*, the adversary also outputs* $\boldsymbol{\mu}^i \leftarrow \mathcal{D}^\ell$*, which must satisfy some constraints discussed later. 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* $1/2$*.*

*In the* selective *game, the adversary must announce the challenge in the first step, before receiving the public key. In the* semi-adaptive *game, the adversary must announce the challenge after seeing the public key but before making any key requests.*

We next define the notion of admissible adversary.

**Definition 2.4** **(Admissible Adversary).** *We say an adversary is* $\mathcal{F}$*-admissible if for any pair of challenge messages* $\mathbf{z}_0, \mathbf{z}_1 \in R_q^\ell$ *and its corresponding vector* $\boldsymbol{\mu}^i \leftarrow \mathcal{D}^\ell$*, any queried key* $\mathbf{v}_i \in R_q^\ell$ *and corresponding function* $f_i \in \mathcal{F}$*, it holds that* $\langle \mathbf{v}_i, \mathbf{z}_0 - \mathbf{z}_1 \rangle = f_i(\boldsymbol{\mu})$ *.*

**Definition 2.5** **(**Noisy-IND **security).**

*A* $(\mathcal{D}, \mathcal{F}, B)$ *noisy linear FE scheme* NLinFE *is* Noisy-IND *secure if for all* $\mathcal{F}$*-admissible probabilistic polynomial-time adversaries* Adv*, the advantage of* Adv *in the* Noisy-IND *security game is negligible in the security parameter* $\kappa$*.*

*Remark 2.6.* In most of our constructions of NLinFE, the precise distribution of $f_i(\boldsymbol{\mu})$ will not be important, and it will suffice that $\|f_i(\boldsymbol{\mu})\| < B_{\mathsf{leak}}$ for some bound $B_{\mathsf{leak}}$, to perform the noise flooding. While it may appear strange to restrict the adversary to choosing messages and functions that satisfy a strong constraint such as the above, such a restricted adversary suffices for our main application in Section 4.

## 3 Broader Classes of Randomness Generators

In this section we define broader classes of randomness generators that suffice for our bootstrapping.

### 3.1 Correlated Noise Generators

In this section we define the notion of a correlated noise generator, which we denote by $\mathsf{CNG}$. We denote by $R$ the ring of integers $\mathbb{Z}$ or the ring of polynomials $\mathbb{Z}[x]/f(x)$ where $f(x) = x^d+1$. Let $\mathcal{D}_1$ be a distribution over $R$ and $\mathcal{F} : R^w \to R$ be a set of deterministic functions. Let $\mathsf{Dom}_{\mathsf{Cng}}, \mathsf{Rg}_{\mathsf{Cng}}$ be finite subsets of $R$, let $\mathcal{G} : \mathsf{Dom}^n_{\mathsf{Cng}} \to \mathsf{Rg}^m_{\mathsf{Cng}}$ be a family of deterministic functions and $\mathcal{D}_2$ be a distribution over $\mathsf{Dom}_{\mathsf{Cng}}$. We require that $n$ be linear in $w$, i.e. $n = O(w, \mathrm{poly}(\kappa))$.

**Definition 3.1** **($(\mathcal{D}_1, \mathcal{F})$- Correlated Noise Generator).** *We say that $(\mathcal{D}_2, \mathcal{G})$ is a $(\mathcal{D}_1, \mathcal{F})$- Correlated Noise Generator (*$\mathsf{CNG}$*) if the advantage of any P.P.T adversary $\mathcal{A}$ is negligible in the following game:*

1. *Challenger chooses $n$ i.i.d samples $\boldsymbol{\beta} \leftarrow \mathcal{D}_2^n$.*
2. *The adversary $\mathcal{A}$ does the following:*
   *(a) It chooses $m$ functions $f_1, \ldots, f_m \in \mathcal{F}$.*
   *(b) It samples $\boldsymbol{\mu} \leftarrow \mathcal{D}_1^w$.*
   *(c) It returns $(\{f_i\}_{i\in[m]}, \boldsymbol{\mu})$ to the challenger.*
3. *The challenger chooses $m$ functions $G_1, \ldots, G_m \in \mathcal{G}$. It tosses a coin $b$. If $b = 0$, it returns $\{f_i(\boldsymbol{\mu}) + G_i(\boldsymbol{\beta})\}_{i\in[m]}$, else it returns $\{G_i(\boldsymbol{\beta})\}_{i\in[m]}$.*
4. *The adversary outputs a guess for the bit $b$ and wins if correct.*

*We will refer to $\boldsymbol{\beta}$ as the seed of the* $\mathsf{CNG}$*. We say that an* $\mathsf{CNG}$ *has polynomial stretch if $m = n^{1+c}$ for some constant $c > 0$.*

### 3.2 Non Boolean Pseudorandom Generators

As discussed in Section 1, in prior work [68, 71], Boolean PRGs were required in order to compute the binary randomness needed for constructing randomizing polynomials. In our case, the PRG output must satisfy a much weaker property. Say we can bound $\|f_i(\boldsymbol{\mu})\| \leq \epsilon$ for $i \in [m]$. Then we require the PRG output $G_i(\boldsymbol{\beta})$ to computationally flood $f_i(\boldsymbol{\mu})$ for $i \in [m]$, i.e., $G_i(\boldsymbol{\beta}) + f_i(\boldsymbol{\mu}) \overset{c}{\approx} G_i(\boldsymbol{\beta}), \;\; \forall i \in [m]$.

## 4 Functional Encryption for $\mathsf{NC}_1$

In this section, we construct a functional encryption scheme for $\mathsf{NC}_1$, denoted by $\mathsf{FeNC}_1$, from a correlated noise generator $\mathsf{CNG}$, the $\mathsf{RLWE}$ assumption and a noisy linear functional encryption scheme $\mathsf{NLinFE}$.

*Background.* Let $R = \mathbb{Z}[x]/(\phi)$ where $\phi = x^d + 1$ and $d$ is a power of 2. Let $R_p \triangleq R/pR$ for any large prime $p$ satisfying $p = 1 \mod 2n$.

We consider arithmetic circuits $\mathcal{F} : R_{p_0}^w \to R_{p_0}$ of depth $d$, consisting of alternate addition and multiplication layers. For circuits with long output, say $\ell$, we consider $\ell$ functions, one computing each output bit. For $k \in [d]$, layer $k$ of the circuit is associated with a modulus $p_k$. For an addition layer at level $k$, the modulus $p_k$ will be the same as the previous modulus $p_{k-1}$; for a multiplication layer at level $k$, we require $p_k > p_{k-1}$. Thus, we get a tower of moduli $p_1 < p_2 = p_3 < p_4 = \ldots < p_d$. We define encoding functions $\mathcal{E}^k$ for $k \in [d]$ such that $\mathcal{E}^k : R_{p_{k-1}} \to R_{p_k}$. The message space of the scheme $\mathsf{FeNC}_1$ is $R_{p_0}$.

At level $k$, the encryptor will provide $L^k$ encodings, denoted by $\mathcal{C}^k$, for some $L^k = O(2^k)$. For $i \in [L^k]$ we define

$$\mathcal{E}^k(y_i) = u_i^k \cdot s + p_{k-1} \cdot \eta_i^k + y_i.$$

Here $u_i^k \in R_{p_k}$ is called the "label" or "public key" of the encoding, $\eta_i^k$ is noise chosen from some distribution $\chi_k$, $s \leftarrow R_{p_1}$ is the RLWE secret, and $y_i \in R_{p_{k-1}}$ is the message being encoded. We will refer to $\mathcal{E}^k(y_i)$ as the *Regev encoding* of $y_i$. We denote:

$$\mathsf{PK}\big(\mathcal{E}^k(y_i)\big) \triangleq u_i^k, \quad \mathsf{Nse}(\mathcal{C}^k) \triangleq p_{k-1} \cdot \eta_i^k$$

The messages encoded in level $k$ encodings $\mathcal{C}^k$ are denoted by $\mathcal{M}^k$.

Agrawal and Rosen [8] show that at level $k$, the decryptor is able to compute a Regev encoding of functional message $f^k(\mathbf{x})$ where $f^k$ is the circuit $f$ restricted to level $k$. Formally:

**Theorem 4.1.** *[8] There exists a set of encodings $\mathcal{C}^i$ for $i \in [d]$, such that:*

1. **Encodings have size sublinear in circuit.** $\forall i \in [d]$ $|\mathcal{C}^i| = O(2^i)$.
2. **Efficient public key and ciphertext evaluation algorithms.** *There exist efficient algorithms $\mathsf{Eval}_{\mathsf{PK}}$ and $\mathsf{Eval}_{\mathsf{CT}}$ so that for any circuit $f$ of depth $d$, if $\mathsf{PK}_f \leftarrow \mathsf{Eval}_{\mathsf{PK}}(\mathsf{PK}, f)$ and $\mathsf{CT}_{(f(\mathbf{x}))} \leftarrow \mathsf{Eval}_{\mathsf{CT}}(\underset{i\in[d]}{\cup} \mathcal{C}^i, f)$, then $\mathsf{CT}_{(f(\mathbf{x}))}$ is a "Regev encoding" of $f(\mathbf{x})$ under public key $\mathsf{PK}_f$. Specifically, for some LWE secret $s$, we have:*
$$\mathsf{CT}_{(f(\mathbf{x}))} = \mathsf{PK}_f \cdot s + p_{d-1} \cdot \eta_f^{d-1} + \mu_{f(\mathbf{x})} + f(\mathbf{x}) \tag{4.1}$$
*where $p_{d-1} \cdot \eta_f^{d-1}$ is $\mathsf{RLWE}$ noise and $\mu_{f(\mathbf{x})} + f(\mathbf{x})$ is the desired message $f(\mathbf{x})$ plus some noise $\mu_{f(\mathbf{x})}$*[9].
3. **Ciphertext and public key structure.** *The structure of the functional ciphertext is as:*
$$\mathsf{CT}_{f(\mathbf{x})} = \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \langle \mathsf{Lin}_f, \mathcal{C}^d \rangle \tag{4.2}$$
*where $\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) \in R_{p_{d-1}}$ is a degree $d$ polynomial and $\mathsf{Lin}_f \in R_{p_d}^{L_d}$ computed by $\mathsf{Eval}_{\mathsf{PK}}(\mathsf{PK}, f)$ is a linear function. We also have*
$$f(\mathbf{x}) + \mu_{f(\mathbf{x})} = \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \langle \mathsf{Lin}_f, \mathcal{M}^d \rangle \tag{4.3}$$

[9] Here $\mu_{f(\mathbf{x})}$ is clubbed with the message $f(\mathbf{x})$ rather than the RLWE noise $p_{d-1} \cdot \eta_f^{d-1}$ since $\mu_{f(\mathbf{x})} + f(\mathbf{x})$ is what will be recovered after decryption of $\mathsf{CT}_{f(\mathbf{x})}$.

*where $\mathcal{M}^d$ are the messages encoded in $\mathcal{C}^d$. The public key for the functional ciphertext is structured as:*

$$\mathsf{PK}\big(\mathsf{CT}_{f(\mathbf{x})}\big) = \Big\langle \mathsf{Lin}_f,\ \big(\mathsf{PK}(\mathcal{C}^d_1), \ldots, \mathsf{PK}(\mathcal{C}^d_{L_d})\big)\Big\rangle \tag{4.4}$$

4. **Noise Structure.** *The term $\mu_{f(\mathbf{x})}$ is the noise resulting from FHE evaluation of function $f$ on the encodings of $\mathbf{x}$. Moreover, $\mu_{f(\mathbf{x})}$ can be expressed as a linear combination of noise terms, each noise term being a multiple of $p_k$ for $k \in \{0, \ldots, d-1\}$.*

*The Encodings.* The encodings $\mathcal{C}^k$ for $k \in [d]$ are defined recursively as:

1. $\mathcal{C}^1 \triangleq \{\mathcal{E}^1(x_i), \mathcal{E}^1(s)\}$
2. If $k$ is a multiplication layer, $\mathcal{C}^k = \{\mathcal{E}^k(\mathcal{C}^{k-1}), \mathcal{E}^k(\mathcal{C}^{k-1} \cdot s), \mathcal{E}^k(s^2)\}$[10]. If $k$ is an addition layer, let $\mathcal{C}^k = \mathcal{C}^{k-1}$.

We will use $\mathsf{NLinFE}$ to enable the decryptor to compute $\langle \mathsf{Lin}_f, \mathcal{M}^d\rangle + G_f(\boldsymbol{\beta})$ where $G_f(\boldsymbol{\beta})$ is a large noise term that floods functional noise $\mu_{f(\mathbf{x})}$. She may then compute $\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1})$ herself and by Equation 4.3 recover $f(\mathbf{x}) + \mu_{f(\mathbf{x})} + G_f(\boldsymbol{\beta})$.

## 4.1 Construction

Next, we proceed to describe the construction. The construction below supports a single function of output length $\ell$ or equivalently $\ell$ functions with constant size output (however, in this case $\ell$ must be fixed in advance and input to all algorithms).

$\mathsf{FeNC_1.Setup}(1^\kappa, 1^w, 1^d)$: Upon input the security parameter $\kappa$, the message dimension $w$, and the circuit depth $d$, do:

1. For $k \in [d]$, let $L_k = |\mathcal{C}^k|$ where $\mathcal{C}^k$ is as defined in Theorem 4.1. For $k \in [d-1]$, $i \in [L_k]$, choose uniformly random $u^k_i \in R_{p_k}$. Denote $\mathbf{u}^k = (u^k_i) \in R^{L_k}_{p_k}$.
2. Invoke $\mathsf{NLinFE.Setup}(1^\kappa, 1^{L_d}, p_d)$ to obtain $\mathsf{PK} = \mathsf{NLinFE.PK}$ and $\mathsf{MSK} = \mathsf{NLinFE.MSK}$.
3. Sample a CNG seed $\boldsymbol{\beta} \leftarrow \mathcal{D}^n_{\mathsf{seed}}$. Sample $t_0, \ldots, t_{L_d} \leftarrow R_{p_{d-1}}$ and let $\mathbf{t} = (t_0, \ldots, t_{L_d})$.
4. Output $\mathsf{PK} = (\mathbf{u}^1, \ldots, \mathbf{u}^{d-1}, \mathsf{NLinFE.PK})$ and $\mathsf{MSK} = (\mathsf{NLinFE.MSK}, \boldsymbol{\beta}, \mathbf{t})$.

$\mathsf{FeNC_1.KeyGen}(\mathsf{MSK}, f)$: Upon input the master secret key $\mathsf{NLinFE.MSK}$, CNG seed $\boldsymbol{\beta}$ and a circuit $f : R^w_{p_0} \to R_{p_0}$[11] of depth $d$, do:

1. Let $\mathsf{Lin}_f \leftarrow \mathsf{Eval_{PK}}(\mathsf{PK}, f) \in R^{L_d}_{p_d}$ as described in Equation 4.4.
2. Let $G_f$ denote the CNG chosen corresponding to function $f$ as described in the full version [3].
3. Compute $\mathsf{key}_f = \langle \mathsf{Lin}_f, \mathbf{t}\rangle - G_f(\boldsymbol{\beta})$.

[10] Here, we use the same secret $s$ for all RLWE samples, but this is for ease of exposition – it is possible to have a different secret at each level so that circular security need not be assumed. We do not describe this extension here.

[11] We will let the adversary request $\ell$ functions

4. Let $\mathsf{SK}_f = \mathsf{NLinFE.KeyGen}(\mathsf{MSK}, \mathsf{Lin}_f \| \mathsf{key}_f)$.

$\mathsf{FeNC}_1.\mathsf{Enc}(\mathbf{x}, \mathsf{PK})$: Upon input the public key and the input $\mathbf{x}$, do:

1. Compute the encodings $\mathcal{C}^k$ for $k \in [d-1]$ as defined in Theorem 4.1. Denote by $s$ the RLWE secret used for these encodings.
2. Define $\mathcal{M}^d = \left( \mathcal{C}^{d-1}, \ \mathcal{C}^{d-1} \cdot s, \ s^2 \right) \in R_{p_d}^{L_d}$. Compute $\mathcal{C}^d = \mathsf{NLinFE.Enc}(\mathsf{NLinFE.PK}, \mathcal{M}^d)$.
3. Output $\mathsf{CT}_{\mathbf{x}} = (\{\mathcal{C}^k\}_{k \in [d]})$.

$\mathsf{FeNC}_1.\mathsf{Dec}(\mathsf{PK}, \mathsf{CT}_{\mathbf{x}}, \mathsf{SK}_f)$: Upon input a ciphertext $\mathsf{CT}_{\mathbf{x}}$ for vector $\mathbf{x}$, and a secret key $\mathsf{SK}_f$ for circuit $f$, do:

1. Compute $\mathsf{CT}_{f(\mathbf{x})} = \mathsf{Eval}_{\mathsf{CT}}(\{\mathcal{C}^k\}_{k \in [d-1]}, f)$. Express $\mathsf{CT}_{f(\mathbf{x})} = \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \langle \mathsf{Lin}_f, \mathcal{C}^d \rangle$ as described in Equation 4.2.
2. Compute $\mathsf{NLinFE.Dec}(\mathsf{SK}_f, \mathcal{C}^d)$ to obtain $\langle \mathsf{Lin}_f, \mathcal{M}^d \rangle + \eta_f$ for some noise $\eta_f$ added by $\mathsf{NLinFE}$.
3. Compute $\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \langle \mathsf{Lin}_f, \mathcal{M}^d \rangle + \eta_f \mod p_d \mod p_{d-1}, \ldots, \mod p_0$ and output it.

## 4.2 Correctness

Correctness follows from correctness of $\mathsf{Eval}_{\mathsf{PK}}$, $\mathsf{Eval}_{\mathsf{CT}}$ and $\mathsf{NLinFE}$. We have by correctness of $\mathsf{Eval}_{\mathsf{PK}}$, $\mathsf{Eval}_{\mathsf{CT}}$ that:

$$\begin{aligned}
\mathsf{CT}_{f(\mathbf{x})} &= \langle \mathsf{Lin}_f, \mathcal{C}^d \rangle + \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) \\
\mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \mathsf{NLinFE.Dec}(\mathsf{SK}_f, \mathcal{C}^d) &= \mathsf{Poly}_f(\mathcal{C}^1, \ldots, \mathcal{C}^{d-1}) + \langle \mathsf{Lin}_f, \mathcal{M}^d \rangle + \eta_f \\
&= f(\mathbf{x}) + \mu_{f(\mathbf{x})} + \eta_f \ \text{ by theorem 4.1} \\
&= f(\mathbf{x}) \mod p_d \mod p_{d-1}, \ldots, \mod p_0
\end{aligned}$$

where the last step follows since:

1. $\mu_{f(\mathbf{x})}$ and $\eta_f$ are linear combinations of noise terms, each noise term being a multiple of $p_k$ for $k \in \{0, \ldots, d-1\}$. For details regarding the structure of the noise terms $\mu_{f(\mathbf{x})}$. The noise term $\eta_f$ is chosen by $\mathsf{NLinFE}$ to flood $\mu_{f(\mathbf{x})}$ as discussed in Section 2, and hence is also a linear combination of noise terms, each noise term being a multiple of $p_k$ for $k \in \{0, \ldots, d-1\}$.
2. We set the parameters so that $p_i$ is sufficiently larger than $p_{i-1}$ for $i \in [d]$, so that over $R_{p_i}$, any error term which is a multiple of $p_{i-1}$ may be removed by reducing modulo $p_{i-1}$. Thus the successive computation of $\mod p_i$, for $i = d, \ldots, 0$, results in $f(\mathbf{x}) \mod p_0$ in the end.

## 4.3 Efficiency and Security

The size of the ciphertext is $|\bigcup_{k \in [d-1]} \mathcal{C}^k| + |\mathsf{NLinFE.CT}(\mathcal{M}^d)|$. Note that $|\bigcup_{k \in [d-1]} \mathcal{C}^k| = O(2^d)$ and $|\mathcal{M}^d| = O(2^d)$ by Theorem 4.1. All our constructions of $\mathsf{NLinFE}$ will have compact ciphertext, hence the ciphertext of the above scheme is also sublinear in circuit size. We refer the reader to the full version [3] for our constructions of $\mathsf{NLinFE}$.

In the full version [3], we prove the following security theorem:

**Theorem 4.2.** *Assume the noisy linear FE scheme* NLinFE *satisfies semi-adaptive indistinguishability based security as in Definition 2.5 and that $G$ is a secure* CNG *as defined in Definition 3.1. Then, the construction* $\mathsf{FeNC}_1$ *achieves semi-adaptive indistinguishability based security.*

## References

1. Abdalla, M., Bourse, F., Caro, A.D., Pointcheval, D.: Simple functional encryption schemes for inner products. Cryptology ePrint Archive, Report 2015/017 (2015), `http://eprint.iacr.org/` To appear in PKC'15.
2. Agrawal, S.: Stronger security for reusable garbled circuits, new definitions and attacks. In: Crypto (2017)
3. Agrawal, S.: Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. Cryptology ePrint Archive, Report 2018 (2018)
4. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: EUROCRYPT. pp. 553–572 (2010)
5. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Asiacrypt (2011)
6. Agrawal, S., Libert, B., Stehle, D.: Fully secure functional encryption for linear functions from standard assumptions, and applications. In: Crypto (2016)
7. Agrawal, S., Rosen, A.: Online offline functional encryption for bounded collusions. Eprint/2016 (2016)
8. Agrawal, S., Rosen, A.: Functional encryption for bounded collusions, revisited. In: TCC (2017)
9. Ananth, P., Jain, A., Sahai, A.: Indistinguishability obfuscation without multilinear maps: io from lwe, bilinear maps, and weak pseudorandomness. Cryptology ePrint Archive, Report 2018/615 (2018)
10. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. pp. 308–326 (2015)
11. Ananth, P., Jain, A., Sahai, A.: Achieving compactness generically: Indistinguishability obfuscation from non-compact functional encryption. IACR Cryptology ePrint Archive, 2015:730 (2015)
12. Ananth, P., Sahai, A.: Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In: EUROCRYPT (2017)
13. Apon, D., Döttling, N., Garg, S., Mukherjee, P.: Cryptanalysis of indistinguishability obfuscations of circuits over ggh13. eprint 2016 (2016)
14. Applebaum, B., Ishai, Y., Kushilevitz, E.: Computationally private randomizing polynomials and their applications. Computational Complexity 15(2), 115–162 (2006)
15. Applebaum, B., Ishai, Y., Kushilevitz, E.: How to garble arithmetic circuits. In: IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011. pp. 120–129 (2011)
16. Baltico, C.E.Z., Catalano, D., Fiore, D., Gay, R.: Practical functional encryption for quadratic functions with applications to predicate encryption. In: Crypto (2017)
17. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (im)possibility of obfuscating programs. In: CRYPTO (2001)
18. Barak, B., Brakerski, Z., Komargodski, I., Kothari, P.: Limits on low-degree pseudorandom generators (or: Sum-of-squares meets program obfuscation). In: Eurocrypt (2018)

19. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy. pp. 321–334 (2007)
20. Bitansky, N., Garg, S., Lin, H., Pass, R., Telang, S.: 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. pp. 439–448 (2015)
21. Bitansky, N., Nishimaki, R., Passelègue, A., Wichs, D.: From cryptomania to obfustopia through secret-key functional encryption. In: TCC (B2). Lecture Notes in Computer Science, vol. 9986, pp. 391–418 (2016)
22. Bitansky, N., Paneth, O., Wichs, D.: 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. pp. 474–502 (2016)
23. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. FOCS 2015, 163 (2015), http://eprint.iacr.org/2015/163
24. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: CRYPTO. pp. 213–229 (2001)
25. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: EUROCRYPT. pp. 533–556 (2014)
26. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: TCC. pp. 535–554 (2007)
27. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: CRYPTO. pp. 290–307 (2006)
28. Brakerski, Z., Tsabary, R., Vaikuntanathan, V., Wee, H.: Private constrained prfs (and more) from lwe. In: Theory of Cryptography Conference (TCC) (2017)
29. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-lwe and security for key dependent messages. In: CRYPTO (2011)
30. Canetti, R., Holmgren, J., Jain, A., Vaikuntanathan, V.: 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. pp. 429–437 (2015)
31. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: TCC (2015)
32. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: EUROCRYPT. pp. 523–552 (2010)
33. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: Proc. of EUROCRYPT. LNCS, vol. 9056, pp. 3–12. Springer (2015)
34. Cheon, J.H., Fouque, P.A., Lee, C., Minaud, B., Ryu, H.: Cryptanalysis of the new clt multilinear map over the integers. Eprint 2016/135
35. Cheon, J.H., Jeong, J., Lee, C.: An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low level encoding of zero. Eprint 2016/139
36. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: IMA Int. Conf. pp. 360–363 (2001)
37. Coron, J.S., Gentry, C., Halevi, S., Lepoint, T., Maji, H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing without low-level zeroes: New mmap attacks and their limitations. In: Advances in Cryptology–CRYPTO 2015, pp. 247–266. Springer (2015)
38. Coron, J.S., Lee, M.S., Lepoint, T., Tibouchi, M.: Zeroizing attacks on indistinguishability obfuscation over clt13. Eprint 2016 (2016)
39. Coron, J., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 476–493 (2013)

40. Cramer, R., Ducas, L., Peikert, C., Regev, O.: Recovering short generators of principal ideals in cyclotomic rings. In: EUROCRYPT (2016)
41. Ding, J., Yang, B.Y.: Multivariate Public Key Cryptography, pp. 193–241. Springer Berlin Heidelberg (2009)
42. Farshim, P., Hesse, J., Hofheinz, D., Larraia, E.: Graded encoding schemes from obfuscation. In: PKC (2018)
43. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT (2013)
44. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. pp. 1–17 (2013)
45. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS (2013), http://eprint.iacr.org/
46. Garg, S., Gentry, C., Halevi, S., Sahai, A., Waters, B.: Attribute-based encryption for circuits from multilinear maps. In: CRYPTO (2013)
47. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Fully secure functional encryption without obfuscation. In: IACR Cryptology ePrint Archive. vol. 2014, p. 666 (2014), http://eprint.iacr.org/2014/666
48. Garg, S., Gentry, C., Halevi, S., Zhandry, M.: Functional encryption without obfuscation. In: Kushilevitz, E., Malkin, T. (eds.) Theory of Cryptography (2016)
49. Garg, S., Miles, E., Mukherjee, P., Sahai, A., Srinivasan, A., Zhandry, M.: Secure obfuscation in a weak multilinear map model. In: Hirt, M., Smith, A. (eds.) Theory of Cryptography. pp. 241–268. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
50. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: STOC. pp. 169–178 (2009)
51. Gentry, C., Gorbunov, S., Halevi, S.: Graph-induced multilinear maps from lattices. In: TCC (2015)
52. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC. pp. 197–206 (2008)
53. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press, New York, NY, USA (2000)
54. Goldwasser, S., Tauman Kalai, Y., Popa, R., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Proc. of STOC. pp. 555–564. ACM Press (2013)
55. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: STOC. pp. 555–564 (2013)
56. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions from multiparty computation. In: CRYPTO (2012)
57. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute based encryption for circuits. In: STOC (2013)
58. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Predicate encryption for circuits from lwe. In: Crypto (2015)
59. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security. pp. 89–98 (2006)
60. Hu, Y., Jia, H.: Cryptanalysis of GGH map. Cryptology ePrint Archive: Report 2015/301 (2015)
61. Ishai, Y., Kushilevitz, E.: Randomizing polynomials: A new representation with applications to round-efficient secure computation. In: FOCS (2000)

62. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: EUROCRYPT. pp. 146–162 (2008)
63. Komargodski, I., Moran, T., Naor, M., Pass, R., Rosen, A., Yogev, E.: One-way functions and (im)perfect obfuscation. In: 55th IEEE Annual Symposium on Foundations of Computer Science, FOCS (2014)
64. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. In: STOC (2015)
65. Langlois, A., Stehlé, D., Steinfeld, R.: Gghlite: More efficient multilinear maps from ideal lattices. In: EUROCRYPT (2014)
66. Lewko, A.B., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: EUROCRYPT. pp. 62–91 (2010)
67. Lin, H.: Indistinguishability obfuscation from constant-degree graded encoding schemes. In: EUROCRYPT. pp. 28–57 (2016)
68. Lin, H.: Indistinguishability obfuscation from sxdh on 5-linear maps and locality-5 prgs. In: Crypto (2017)
69. Lin, H., Matt, C.: Pseudo flawed-smudging generators and their application to indistinguishability obfuscation. Cryptology ePrint Archive, Report 2018/646 (2018)
70. Lin, H., Pass, R., Seth, K., Telang, S.: Output-compressing randomized encodings and applications. In: TCC-A (2016)
71. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local prgs. In: Crypto (2017)
72. Lin, H., Vaikuntanathan, V.: Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In: FOCS (2016)
73. Lombardi, A., Vaikuntanathan, V.: On the non-existence of blockwise 2-local prgs with applications to indistinguishability obfuscation. In: TCC (2018)
74. Matsumoto, T., Imai, H.: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In: Advances in Cryptology — EUROCRYPT '88. pp. 419–453. Springer Berlin Heidelberg, Berlin, Heidelberg (1988)
75. Miles, E., Sahai, A., Zhandry, M.: Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over ggh13. In: Crypto (2016)
76. Peikert, C.: A Decade of Lattice Cryptography, vol. 10, pp. 283–424 (03 2016)
77. Pellet-Mary, A.: Quantum attacks against indistinguishablility obfuscators proved secure in the weak multilinear map model. In: Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III. pp. 153–183 (2018)
78. Pellet-Mary, A.: Quantum attacks against indistinguishablility obfuscators proved secure in the weak multilinear map model. In: Advances in Cryptology - CRYPTO (2018)
79. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J.ACM 56(6) (2009), extended abstract in STOC'05
80. Sahai, A., Waters, B.: Functional encryption:beyond public key cryptography. Power Point Presentation, 2008. http://userweb.cs.utexas.edu/˜bwaters/presentations/ files/functional.ppt.
81. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: EUROCRYPT. pp. 457–473 (2005)
82. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: Deniable encryption, and more. In: STOC (2014), http://eprint.iacr.org/2013/454.pdf
83. Waters, B.: Functional encryption for regular languages. In: Crypto (2012)
84. Wolf, C.: Multivariate Quadratic Polynomials In Public Key Cryptography. Ph.D. thesis, KATHOLIEKE UNIVERSITEIT LEUVEN (2005)