Evaluating the effectiveness of heuristic worst-case noise analysis in FHE

No Author Given

No Institute Given

Abstract. The purpose of this paper is to test the accuracy of worst-case heuristic bounds on the noise growth in ring-based homomorphic encryption schemes. We use the methodology of Iliashenko (PhD thesis, 2019) to provide a new heuristic noise analysis for the BGV scheme. We demonstrate that for both the BGV and FV schemes, this approach gives tighter bounds than previous heuristic approaches, by as much as 10 bits of noise budget. Then, we provide experimental data on the noise growth of HElib and SEAL ciphertexts, in order to evaluate how well the heuristic bounds model the noise growth in practice. We find that, in spite of our improvements, there is still a gap between the heuristic estimate of the noise and the observed noise in practice. We extensively justify that the heuristic worst-case approach inherently leads to this gap, and hence leads to selecting significantly larger parameters than needed. We propose tightening this gap as an open problem, and suggest a possible solution. As an additional contribution, we update the comparison between the two schemes presented by Costache and Smart (CT-RSA, 2016). Using the new analysis, we show that FV slightly outperforms BGV, even for large plaintext moduli, well beyond the crossover point reported by Costache and Smart.

1 Introduction

Fully homomorphic encryption enables the evaluation of arbitrary polynomials on encrypted data, without requiring access to the secret key. In contrast, somewhat homomorphic encryption enables the evaluation of limited functions on encrypted data; this is usually characterised by a bound of the depth of the circuits that can be evaluated. The first fully homomorphic encryption scheme was presented by Gentry [24], whose construction augmented a somewhat homomorphic encryption scheme with a technique known as bootstrapping.

In all homomorphic encryption schemes ciphertexts contain noise that grows during homomorphic evaluation operations. Once the noise exceeds a certain threshold, decryption will fail. In practice, managing the noise to ensure it is always below the threshold can be done in two ways. The first approach uses the bootstrapping procedure, which takes as input a ciphertext with large noise, and outputs a new ciphertext which has less noise and can be further computed on. Hence by bootstrapping at appropriate points, the entire evaluation can be performed. The second approach is to pre-determine the function to be evaluated and set the parameters so as to allow for the noise growth that this specific function will incur. Using this method, we are sure that the output ciphertext at the end of the evaluation will have noise below the threshold, thus no bootstrapping will be necessary and correct decryption is ensured. In either case, good understanding of the noise growth behaviour is essential to achieve correctness and optimal performance. In fact, a good understanding of the noise growth in any scheme is crucial to parameter setting, large parameters remaining one of the main hurdles in homomorphic encryption development.

1.1 Contributions

This paper presents two main contributions. Firstly, we evaluate the effectiveness of the heuristic worst-case method. We do so by reworking the noise growth estimates produced by this method for the somewhat homomorphic encryption (SHE) schemes BGV [11] and FV[1] [23]. We use the Iliashenko method [29] for obtaining the heuristic bounds. The bounds for FV were presented in [29], with the exception of modulus switching, while the BGV bounds we present using this method are new. We compare these new bounds against the previous heuristic analyses [14, 20, 26, 25], and show that Iliashenko's approach improves on the previous approach by as much as 10 bits of noise budget in certain settings, particularly so for the FV scheme. To demonstrate this, we provide the noise estimated by the old bounds and the new approach in Tables 1, 2, 3, 4 and 8.

Next, we evaluate the practical noise growth incurred when evaluating homomorphic operations in BGV and FV by looking at their implementations in the HElib and SEAL libraries, respectively. The first HElib noise results concern the growth of the *critical quantity* [20] and can be found in Table 1. In order to facilitate comparison, we define and implement in HElib a noise budget for the critical quantity for BGV, analogous to the *invariant noise budget* [14] for FV that is implemented in SEAL. The results in terms of the noise budget are presented in Table 2. Our SEAL noise results are presented in Tables 3 and Table 4, for the binary encoding and batch settings, respectively. We find that, despite the improvements mentioned above, the predictions are not tight, and that a significant gap between the predicted noise and the actual noise remains. We will refer to this gap as the *heuristic-to-practical gap*.

We conclude that a worst-case heuristic estimate of homomorphic noise growth is inadequate. That is to say, we conjecture that the theoretical bounds we present in this work cannot be made tighter. We give an extensive justification for this conjecture, and comment on other methods we attempted for improvement, in Section 6. Therefore, we propose further tightening the heuristic-to-practical gap as an open problem. We believe that a better model of the noise growth behaviour can only be achieved by fine-tuning the analysis of a specific scheme to its specific implementation.

Our second main contribution, which can be of independent interest, is to use our improved analysis to update the Costache-Smart [20] comparison of the BGV

[1] FV is based on a scheme of Brakerski [10] and hence is sometimes referred to as BFV.

and FV schemes. We improve upon the previous work of Costache-Smart in several ways. As well as applying the updated noise analysis following [29], we use a different notion of noise for FV than that used in [20], namely the invariant noise. In addition, our comparison relies on an up-to-date security analysis conforming to HE standards [1]. Indeed, it has since been shown [19] that parameters used in [20] that were estimated to have 80 bits of security are now estimated to have as little as 50. In contrast, the HE standards security recommendations start at the level of 128 bits [1]. We further observe that our analysis also allows for a more flexible modulus switching for FV. This is an important functionality in practice, for example to be able to compress communication after homomorphic computations are done.

The BGV and FV schemes remain two of the most popular SHE schemes, as they continue to see many performance improvements and optimisations and are implemented in several actively maintained homomorphic encryption libraries, including PALISADE[2] as well as SEAL and HElib. It is therefore an important question to accurately assess how they perform and compare them against one another. Finally, we no longer consider the NTRU-based schemes YASHE [8] and LTV [34]. It has since been shown that such schemes may be vulnerable to attacks in "overstretched" parameter settings of interest [4, 31] and as a consequence most implementations of NTRU-based homomorphic encryption schemes are not currently maintained.

We conduct our comparison for various plaintext moduli and present our results in Tables 5, 6 and 7. We find that in some situations, FV has a slight advantage over BGV. It retains this advantage for plaintext moduli much larger than the crossover point reported in Costache-Smart [20]. In most cases, we can conclude from our results that the two schemes present only minor performance differences in terms of supporting a specific homomorphic evaluation. This is not surprising: we can see from the BGV and FV encryption algorithms that the part of a fresh ciphertext that is not the message part (that is, m in BGV and Δm in FV) is essentially the same: terms of the form $-eu + e_1 + e_2 s$ (scaled by t in the case of BGV). Therefore, purely from the perspective of computational capabilities, the question 'Should I prefer the BGV scheme to the FV scheme?' should not be an important one for the implementor deciding which scheme to use.

1.2 Standardisation

Partly due to their widespread implementation, the BGV and FV schemes are among the primary schemes being considered in the ongoing effort to standardise homomorphic encryption[3]. Indeed, the Homomorphic Encryption Security Standard [1] from the standardisation consortium explicitly mentions the comparison of BGV and FV as an open problem, and motivates the present work.

[2] `https://git.njit.edu/palisade/PALISADE`

[3] HomomorphicEncryption.org

After completing the Ring-LWE security and scheme descriptions, the standardisation initiative has started moving fast in the direction on making homomorphic encryption easier to use, in particular through creating a standard library API, and introducing ideas for automation such as a domain specific programming language and a compiler/optimiser toolchain [12]. The analysis presented in our work should be expected to feed into these discussions, as an accurate noise growth estimator is likely to be a central component of any homomorphic computation optimiser or parameter selector tool.

1.3 Related work

Several variants of the FV scheme that improve performance have been proposed in the literature e.g. [6, 27]. Al Badawi *et al.* [5] conclude from experiments that BEHZ-FV [6] has worse noise growth in practice than HPS-FV [27], and call for further study on BEHZ-FV noise growth, which further motivates the present work. In addition, a comparison of BGV as implemented in HElib and FV as implemented in SEAL is left as an open problem in [15].

Apart from that of Costache-Smart [20], other previous comparisons of homomorphic encryption schemes include [30, 32, 36]. We do not consider newer schemes such as CKKS [17] or TFHE [18], which come with entirely different trade-offs. Chimera [9] describes a framework for the FV, CKKS and TFHE schemes, with the goal of providing a common API, rather directly comparing the schemes.

2 Preliminaries

For reasons of space, we introduce the BGV scheme in Appendix A and the FV scheme in Appendix B. We note that in this work we deviate from the original description of FV by also defining a modulus switching operation, as was done in [20]. In particular, we describe switching from a modulus q to a modulus p.

2.1 Parameters

A Ring-LWE-based (levelled) FHE scheme is parameterised by L, n, Q, t, χ, S, w, ℓ and λ. There are L primes $p_0, \ldots, p_{L-1}$ which are used to form the chain of moduli $q_0, \ldots, q_{L-1}$. Elements in the chain of moduli are formed as $q_k = \prod_{j=0}^{k} p_j$. The dimension n is a typically chosen as a power of two, and we will only use such n in this work. The dimension n, plaintext modulus t and the chain of moduli parameterise the underlying plaintext and ciphertext rings. In particular, the ciphertext modulus $Q = q_{L-1} = \prod_{j=0}^{L-1} p_j$ is the product all the primes. Each intermediate prime q_j corresponds to a level and all ciphertexts are with respect to a specific level. We denote by q some fixed level when describing the schemes, so that the ciphertext space at any given moment is $R_q = \mathbb{Z}_q[x]/(x^n + 1)$. Note that for key generation and for fresh ciphertexts, we always have $q = Q$. The plaintext space is always $R_t = \mathbb{Z}_t[x]/(x^n + 1)$. Let w be a base, then $\ell + 1 =$

$\lfloor \log_w q \rfloor + 1$ is the number of terms in the decomposition into base w of an integer in base q. The security parameter is λ.

The Ring-LWE error distribution is denoted χ and is typically a discrete gaussian with standard deviation $\sigma = 3.2$. The underlying Ring-LWE problem, parameterised by n, Q and σ, is a variant with small secret. The parameter S denotes the secret key distribution. In the FV scheme [23] the distribution S is the uniform distribution on the subspace of R_q consisting of polynomials whose coefficients are in the set $\{0, 1\}$. In the SEAL implementation [41] the distribution S is the uniform distribution on the subspace of R_q consisting of polynomials whose coefficients are in the set $\{-1, 0, 1\}$. In the BGV scheme [11], the distribution S is the same as the error distribution χ. In the HElib implementation [28], S is the distribution on the subspace of R_q consisting of polynomials whose coefficients are in the set $\{-1, 0, 1\}$ where each coefficient is sampled as follows: the element 0 is sampled with probability 0.5 and the elements ± 1 are each sampled with probability 0.25. To ensure our comparison is fair, for both BGV and FV, we take S to be the uniform distribution on the subspace of R_q consisting of polynomials whose coefficients are in the set $\{-1, 0, 1\}$.

2.2 Canonical embedding norm

Following previous work [20, 25, 26, 29], we will present heuristic bounds for the noise growth behaviour of FV and BGV with respect to the canonical embedding norm $\|\cdot\|^{\mathrm{can}}$. Throughout this work, the notation $\|a\|$ refers to the infinity norm of a, while $\|a\|^{\mathrm{can}}$ refers to the canonical embedding norm. The canonical embedding norm of an element a is defined to be the infinity norm of the canonical embedding[4] $\sigma(a)$ of a, so $\|a\|^{\mathrm{can}} = \|\sigma(a)\|$.

We will use the following properties of the canonical embedding norm. For any polynomial $a \in R$ we have $\|a\| \leq c_m \|a\|^{\mathrm{can}} \leq \|a\|_1$ where c_m is a constant known as the ring expansion factor (see [22]). We have $c_m = 1$ when the dimension n is a power of two [22]. In this case, it suffices for correctness to ensure that $\|v\|^{\mathrm{can}}$ is less than the maximal value of $\|v\|$ such that decryption succeeds. For any polynomials a, b we have $\|ab\|^{\mathrm{can}} \leq \|a\|^{\mathrm{can}} \|b\|^{\mathrm{can}}$.

For our bounds, we use the method presented in [29]. This allows us to improve our noise bounds compared to previous ones [20, 25, 26] by as much as 11 bits of noise budget in certain settings. Therefore, the noise bounds we present in this work are much tighter than ones presented in previous works.

Let $R = \mathbb{Z}[x]/(x^n + 1)$ and let ζ be a primitive $2n^{th}$ root of unity (it does not matter which one, by the definition of the canonical embedding norm). Let $a \in R$ be a polynomial for which the variance of each coefficient is V_a. Then, the variance of the random variable $a(\zeta)$ is nV_a [20, 26, 29]. We use the fact that $\mathrm{erfc}(6) \approx 2^{-55}$ to obtain the following bound $\|a\|^{\mathrm{can}} \leq 6\sqrt{n}\sqrt{V_a}$.

We also use the following facts. Let V_a and V_b the variances of the coefficients of two polynomials $a \in R$ and $b \in R$, and let γ be a constant. The variance of the coefficients of the polynomial $a + b$ is $V_{a+b} = V_a + V_b$. The variance of the

[4] For a definition of the canonical embedding and other algebraic background, see [35].

coefficients of the polynomial γa is $V_{\gamma a} = \gamma^2 V_a$. The variance of the coefficients of the polynomial ab is $V_{ab} = nV_aV_b$ (see [29] for a proof).

The variances in situations of interest for this paper are as follows. The coefficients of a polynomial f that are distributed uniformly in $[-\frac{k}{2}, \frac{k}{2}]$ have variance $V_f = \frac{k^2}{12}$. The coefficients of a polynomial e that are drawn from an error distribution χ, which has standard deviation σ, have variance $V_e = \sigma^2$. The coefficients of a polynomial s that are drawn from the FV secret key distribution as implemented in SEAL [41] have variance $V_s = \frac{2}{3}$.

3 BGV noise growth in practice

3.1 Noise growth behaviour

In this section we present our heuristic bounds on the noise growth behaviour of BGV, developed using the methodology of [29]. In Section 3.2 we compare our bounds with those that would be obtained following the methodology of [20, 25, 26], and show that our analysis provides a better estimate of the noise growth.

Our bounds use the *critical quantity* [20] definition of noise, which is the notion of noise used in the HElib [28] implementation of BGV. We assume that the plaintext is chosen uniformly at random from the plaintext space. We further assume that the secret key distribution S is the uniform ternary distribution. Note this is a different assumption than was used for developing heuristic bounds for BGV in previous work [20, 25, 26]. In particluar, in Tables 1 and 2, the [20] column refers to the [20] heuristic method with uniform ternary distribution for the secret key. In particular, evaluating the [20] bounds as they are presented would not lead to the numbers we present. We make this choice in order to preserve fairness; using a small hamming weight distribution would not only lead to much smaller bounds, but would also not be a fair comparison.

Definition 1 (BGV critical quantity [20]). *Let* $\mathtt{ct} = (c_0, c_1)$ *be a BGV ciphertext encrypting the message* $m \in R_t$*. Its* critical quantity v *is the polynomial*

$$v = [\mathtt{ct}(s)]_q = (c_0 + c_1 s) \pmod{q}.$$

During decryption, we first compute the critical quantity and then take the result modulo t. If there is no wraparound modulo q then for some integer polynomial k, the critical quantity satisfies $[\mathtt{ct}(s)]_q = m + tk$. The reduction modulo t hence returns m. Therefore for correctness, we require that $\|v\| \leq q/2$.

Lemma 1 (Maximal noise [20]). *A BGV ciphertext* $\mathtt{ct}$ *encrypting a message* m *can be correctly decrypted if the critical quantity* v *satisfies* $\|v\| < q/2$*.*

Encrypt: Let $\mathtt{ct}$ be a fresh BGV encryption of a message $m \in R_t$. With high probability, the critical quantity v in $\mathtt{ct}$ satisfies

$$\|v\|^{\mathrm{can}} \leq 6t\sqrt{\frac{n}{12} + n\sigma^2\left(\frac{4}{3}n + 1\right)}.$$

To see this, we use that for a polynomial a with coefficients with variance V_a, and a scalar t, the polynomial ta has coefficients with variance $V_{at} = t^2 V_a$. The noise polynomial is $v = m + t(e_1 + e_2 s - eu)$. Its coefficients have variance

$$V_v = V_{m+t(e_1+e_2 s-eu)} = V_m + t^2 V_{e_1+e_2 s-eu} = t^2 \left(\frac{1}{12} + \sigma^2 \left(\frac{4}{3} n + 1 \right) \right) .$$

Hence $\|v\|^{\mathrm{can}} \le 6\sqrt{nV_v} = 6\sqrt{nt^2 \left(\frac{1}{12} + \sigma^2 \left(\frac{4}{3} n + 1 \right) \right)}$.

Add [20]: Let $\mathtt{ct}_1$ and $\mathtt{ct}_2$ be two BGV ciphertexts encrypting $m_1, m_2 \in R_t$, and having critical quantities v_1, v_2, respectively. Then the critical quantity v_{add} in their sum $\mathtt{ct}_{\mathrm{add}}$ satisfies $\|v_{\mathrm{add}}\|^{\mathrm{can}} \le \|v_1\|^{\mathrm{can}} + \|v_2\|^{\mathrm{can}}$.

Mult [20]: Let $\mathtt{ct}_1$ and $\mathtt{ct}_2$ be two BGV ciphertexts encrypting $m_1, m_2 \in R_t$, and having critical quantities v_1, v_2, respectively. Then the critical quantity v_{mult} in their product $\mathtt{ct}_{\mathrm{mult}}$ satisfies $\|v_{\mathrm{mult}}\|^{\mathrm{can}} \le \|v_1\|^{\mathrm{can}} \cdot \|v_2\|^{\mathrm{can}}$.

Relinearize: Let $\mathtt{ct}$ be a BGV ciphertext encrypting m and having noise v. Let $\mathtt{ct}_{\mathrm{relin}}$ be the ciphertext obtained by the relinearization of $\mathtt{ct}$. Then with high probability, the critical quantity v_{relin} in $\mathtt{ct}_{\mathrm{relin}}$ satisfies

$$\|v_{\mathrm{relin}}\|^{\mathrm{can}} \le \|v\|^{\mathrm{can}} + t\sqrt{(\ell+1)} n w \sigma \sqrt{3} \, .$$

For reasons of space, this bound is justified in Appendix C.

ModSwitch: Let $\mathtt{ct}$ be a BGV ciphertext encrypting m with critical quantity v with respect to a modulus q. Let $\mathtt{ct}_{\mathrm{mod}}$ be the ciphertext encrypting m obtained by modulus switching to the modulus p. Then with high probability, the critical quantity v_{mod} in $\mathtt{ct}_{\mathrm{mod}}$ satisfies

$$\|v_{\mathrm{mod}}\|^{\mathrm{can}} \le \frac{p}{q} \|v\|^{\mathrm{can}} + t\sqrt{3n + 2n^2} \, .$$

Let $\mathtt{ct}_{\mathrm{mod}} = (c_0', c_1')$, the result of the modulus switching operation applied to $\mathtt{ct} = (c_0, c_1)$. As in [20], we let τ_i be the rounding error of $\frac{p}{q} \cdot \delta_i$. Then:

$$\begin{aligned} \|c_0' - c_1' s\|^{\mathrm{can}} &\le \frac{p}{q} \big(\|c_0 - c_1 s\|^{\mathrm{can}} + \|\delta_0 - \delta_1 s\|^{\mathrm{can}} \big) \le \frac{p}{q} \|v\|^{\mathrm{can}} + \|\tau_0 + \tau_1 s\|^{\mathrm{can}} \\ &\le \frac{p}{q} \|v\|^{\mathrm{can}} + 6t\sqrt{\frac{n}{12} \left(1 + \frac{2n}{3} \right)} \, . \end{aligned}$$

3.2 Practical experiments

In this section we compare the observed critical quantity in HElib ciphertexts formed as a result of certain homomorphic evaluation operations with expected estimates on the noise growth from the heuristic upper bounds. We run the following experiment for a certain number of trials: we step through a specific homomorphic evaluation, and for each operation, we record the observed noise growth. We then output the mean of the observed noise. Separately, we compute an estimate of the noise growth using the heuristic bounds presented in Section 3.1.

HElib offers a debugging function[5] that implements an augmented decryption, which also returns the critical quantity v. We modify this to create a function that returns $\|v\|$.

The evaluation is as follows in the i-th trial. We first generate fresh ciphertexts $\mathtt{ct}_1$ and $\mathtt{ct}_2$ encrypting $i+1$ and i. Next, generate $\mathtt{ct}_3$ as the homomorphic addition of $\mathtt{ct}_1$ and $\mathtt{ct}_2$. Next, generate $\mathtt{ct}_4$ as the homomorphic multiplication of $\mathtt{ct}_3$ and $\mathtt{ct}_2$. Finally, generate $\mathtt{ct}_5$ by modulus switching $\mathtt{ct}_4$ down to the next prime in the chain.

Relinearization for BGV as defined in Appendix A above is not implemented in HElib. Instead, a different variant is implemented (see [26]). Indeed, relinearization can be (and, in practice, is) implemented in a number of ways, all with easy-to-understand additive noise growth. Therefore, we do not investigate the noise growth behaviour during relinearization in our practical experiments.

Table 1 gives the results of this experiment for 10000 trials. We used the follow default parameter settings in HElib: we set the standard deviation of the error distribution as $\sigma = 3.2$ and the security parameter[6] $\lambda = 80$. The HElib parameter c, which relates to relinearization, was set as a default value $c = 2$. We set the number of plaintext slots as $s = 1$ as we did not require batching functionality. We used the default HElib secret distribution, which slightly differs from a uniform ternary secret distribution, as discussed in Section 2.1.

We set the dimension[7] $n \in \{2048, 4096, 8192, 16384\}$. The HElib parameter `nBits` is passed to the function `buildModChain` which sets an appropriate chain of moduli for which the product of all the primes, Q, satisfies $Q \approx 2^{\mathtt{nBits}}$. We set $\mathtt{nBits} \in \{54, 109, 218, 438\}$, which are the same values as for the default Q in SEAL [41]. The parameters for $n = 2048$ were not large enough to perform modulus switching. We set the plaintext modulus[8] as $t = 3$. Such a small plaintext modulus means that the values encrypted in our trials 'cover' the whole plaintext space and hence the assumption used in the noise bounds that m is a random plaintext is reasonable.

Table 1 shows that the heuristic bounds hold on average: the actual observed mean noise is less than the estimated noise. However, it will be difficult to directly compare these results with those for experiments in SEAL, which are given in terms of a *noise budget*, rather than the noise itself [14]. In order to facilitate an

[5] `decryptAndPrint`

[6] In HElib, the security parameter is typically denoted as k. This may not be an accurate security estimate [3].

[7] In HElib, the dimension is selected as m where $n = \varphi(m)$ and $\varphi(\cdot)$ is the Euler totient function. Hence, we set $m \in \{4096, 8192, 16384, 32768\}$. We verified that our other choices allowed for these m using the function `FindM`.

[8] In HElib, the plaintext modulus is parameterised as p^r hence we set $p = 3$ and $r = 1$.

n	Enc			Add			Mult			ModSwitch		
	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$
2048	19.0	17.1	5.12	20.0	18.1	5.62	39.0	35.1	14.7	-	-	-
4096	20.0	18.1	5.19	21.0	19.1	5.69	40.9	37.1	15.3	15.5	14.1	3.62
8192	21.0	19.1	5.25	22.0	20.1	5.76	42.9	39.1	15.8	16.5	15.1	3.65
16384	22.0	20.1	5.31	23.0	21.1	5.81	44.9	41.1	16.4	17.5	16.1	3.70

Table 1. The column $\overline{x}$ gives the logarithm to base 2 of the observed mean of the noise in HElib ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$. The column E gives an estimate of the noise growth using heuristic bounds obtained following our analysis. The remaining column gives an estimate of the noise growth using heuristic bounds obtained following an analysis as in [20].

n	Enc			Add			Mult			ModSwitch		
	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$	[20]	E	$\overline{x}$
2048	34.0	35.0	41.1	33.0	34.0	40.2	14.0	17.0	26.0	-	-	-
4096	88.0	89.0	97.9	87.0	88.0	97.0	67.0	70.0	82.4	38.0	39.0	38.1
8192	196	197	209	195	196	209	174	177	194	146	147	150
16384	415	416	433	414	415	432	392	395	416	365	366	373

Table 2. The column $\overline{x}$ gives the observed mean of the noise budget in HElib ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$. The column E gives an estimate of the noise growth using heuristic bounds obtained following our analysis. The remaining column gives an estimate of the noise growth using heuristic bounds obtained following an analysis as in [20].

easier comparison, we define a noise budget for BGV that is analogous to the invariant noise budget in FV.

Definition 2 (BGV noise budget). *Let* `ct` *be a BGV ciphertext with respect to modulus q having critical quantity v. The* noise budget *for this ciphertext is defined as*

$$\log_2(q) - \log_2(\|v\|) - 1\,.$$

To see that this is an analogous definition, note that for FV the invariant noise budget is defined in [14] as $-\log_2(2 \cdot \|v\|) = \log_2(q) - \log_2(q \cdot \|v\|) - 1$. This captures that for correctness in FV, we require that $q \cdot \|v\| < \frac{q}{2}$. Similarly, Definition 2 captures that for correctness in BGV, we require $\|v\| \leq q/2$.

We implemented a function in HElib to measure the noise budget, and a function to estimate the noise budget using the heuristic bounds. We then ran the same experiment as detailed above to compare the growth of the observed noise budget in HElib ciphertexts with that predicted from the heuristic bounds. Table 2 gives the results of this experiment for 10000 trials.

We see from Tables 1 and 2 that the heuristic bounds hold: the observed mean noise is less than the estimated noise, so the observed mean noise budget is more than the estimated noise budget. Moreover, we see that using our analysis (following [29]) to obtain the heuristic bounds gives an estimate closer to the observed noise than an analysis as in the line of prior work [20, 25, 26].

Despite this improvement, the heuristic bounds are still not tight[9]. For example, for fresh ciphertexts, our heuristic bound predicts 6 to 17 fewer bits of remaining noise budget than the mean observed. We see that the gap compounds as we move through the computation: after multiplication, the gap is 9 to 21 bits. The gap narrows after modulus switching, to below 7 bits. Although the HElib implementation uses a secret key distribution that is slightly different from the uniform ternary distribution assumed in the heuristic bounds, we do not expect this to significantly contribute to the gap.

We also found that the observed noise budgets follow narrow distributions, which gives us confidence that the heuristic bounds will hold very often, and so could be relied upon to set parameters for correctness. However, since the heuristic bounds are not tight, they may lead us to choose larger parameters than is necessary. It is not clear that choosing BGV parameters using the heuristic bounds will be optimal for performance.

4 FV noise growth in practice

4.1 Heuristic upper bounds

In this section, we present a heuristic upper bound for modulus switching in FV following Iliashenko [29]. The heuristic upper bounds for the other FV operations are the same as in [29] and thus we defer them to Appendix D. In Section 4.2 we compare these bounds with those that would be obtained following the methodology of previous work [14, 20, 25, 26], and show that the analysis of Iliashenko provides a better estimate of the noise growth.

The bounds use the *invariant noise* definition for noise [14], as used in the SEAL [41] implementation of FV. We assume that the secret key distribution S is the uniform ternary distribution, as in SEAL [41], and that plaintexts are chosen uniformly at random in the plaintext space.

Definition 3 (FV invariant noise [14]). *Let $\mathtt{ct} = (c_0, c_1)$ be an FV ciphertext encrypting the message $m \in R_t$. Its* invariant noise *v is the polynomial with the smallest infinity norm such that, for some integer coefficient polynomial a,*

$$\frac{t}{q}\mathtt{ct}(s) = \frac{t}{q}(c_0 + c_1 s) = m + v + at\,.$$

The intuition for this definition of noise is that v is exactly the term which will be removed by the rounding in a successful decryption. Therefore for correctness, we require that $\|v\| < \frac{1}{2}$ [14].

[9] An exception is modulus switching for $n = 4096$, which seems to be well-modelled by both approaches for obtaining heuristic bounds.

ModSwitch: Let $\mathtt{ct}$ be an FV ciphertext encrypting m with invariant noise v with respect to a modulus q. Let $\mathtt{ct}_{\mathrm{mod}}$ be the ciphertext encrypting m obtained by modulus switching to the modulus p. Then with high probability, the invariant noise v_{mod} in $\mathtt{ct}_{\mathrm{mod}}$ satisfies

$$\|v_{\mathrm{mod}}\|^{\mathrm{can}} \le \|v\|^{\mathrm{can}} + \frac{t}{p} \cdot \sqrt{3n + 2n^2}\,.$$

Let $\mathtt{ct} = (c_0, c_1)$. Then $\mathtt{ModSwitch}(\mathtt{ct}, p) = (c_0', c_1')$ where $c_0' = \left[\left\lfloor \frac{p}{q} c_0 \right\rceil\right]_p$ and $c_1' = \left[\left\lfloor \frac{p}{q} c_1 \right\rceil\right]_p$. Let ϵ_i for $i \in \{0, 1\}$ be terms introduced from the rounding. By definition of invariant noise (with respect to q) in $\mathtt{ct}$,

$$\begin{aligned}
\frac{t}{p}\left(c_0' + c_1' s\right) &= \frac{t}{p}\left(\left[\left\lfloor \frac{p}{q} c_0 \right\rceil\right]_p + s \cdot \left[\left\lfloor \frac{p}{q} c_1 \right\rceil\right]_p\right) \\
&= \frac{t}{p}\left(\frac{p}{q} c_0 + \epsilon_0 + k_0 p + s \cdot \left(\frac{p}{q} c_1 + \epsilon_1 + k_1 p\right)\right) \\
&= (m + v) + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right) + (k_0 + k_1 s + a)t\,.
\end{aligned}$$

By definition, the invariant noise in the original ciphertext is v such that

$$\frac{t}{q}(c_0 + c_1 s) = m + v + at\,.$$

By the above line,

$$\begin{aligned}
\frac{t}{p}\left(c_0' + c_1' s\right) &= \frac{t}{q}\left(c_0 + c_1 s\right) + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right) + (k_0 + k_1 s)t \\
&= m + v + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right) + (k_0 + k_1 s + a)t\,.
\end{aligned}$$

Hence by definition of invariant noise (with respect to p), $v_{\mathrm{mod}} = v + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right)$. Let $V_\epsilon = \frac{1}{12}$ denote the variance of a coefficient of ϵ_i and $V_s = \frac{2}{3}$ denote the variance of a coefficient of s. Let B be the polynomial $B = \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right)$. Then the variance of the coefficients of the polynomial B is

$$V_B = \frac{t^2}{p^2}\left(V_\epsilon + n V_\epsilon V_s\right) = \frac{t^2}{p^2}\left(\frac{1}{12} + n \cdot \frac{1}{12} \cdot \frac{2}{3}\right) = \frac{t^2}{12p^2}\left(1 + \frac{2n}{3}\right)$$

Hence we can bound v_{mod} as follows:

$$\|v_{\mathrm{mod}}\|^{\mathrm{can}} = \|v\|^{\mathrm{can}} + \left\|\frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right)\right\|^{\mathrm{can}} \le \|v\|^{\mathrm{can}} + 6 \cdot \sqrt{n} \cdot \frac{t}{p} \cdot \sqrt{\frac{1}{12}} \cdot \sqrt{1 + \frac{2n}{3}}\,,$$

which simplifies to the stated bound.

4.2 Practical experiments

In this section we compare the observed noise in SEAL ciphertexts formed as a result of certain homomorphic evaluation operations with expected estimates on the noise growth from the heuristic upper bounds. We run the following experiment for a certain number of trials: we step through a specific homomorphic evaluation and for each operation we record the observed noise growth. We then output the mean of the observed noise. Separately, we compute an estimate of the noise growth using the heuristic bounds.

Recall that since $\|v\| \leq \|v\|^{\text{can}}$, we can use the bounds presented in Section 4.1 as upper bounds for the infinity norm $\|v\|$ of the invariant noise v. Rather than working with the invariant noise v directly, since it can be an extremely small quantity, SEAL instead uses the current *invariant noise budget* [14], which is defined as $-\log_2(2 \cdot \|v\|)$.

We conduct the same evaluation in SEAL as we did in Section 3.2 for HElib. In particular, this means we do not measure the noise growth in relinearization. Apart from the reasons discussed in Section 3.2, this is also necessary for two reasons. Firstly, the choice of the parameter w is no longer part of the API in SEAL, so it is difficult to compare to the relinearization heuristic bound. Secondly, SEAL reserves one of the chain of moduli as 'special prime' used both in relinearization and in a modulus switching implemented as part of the encryption operation. This reduces noise in a fresh SEAL ciphertext, but deviates from a plain FV encryption, and hence would not be accurately captured by the fresh noise bound presented in Equation 1. We modify SEAL to disable this special prime functionality. This enables us to obtain data on the noise growth in an implementation of plain FV encryption, at the cost of being unable to investigate relinearization.

The evaluation is as follows in the i-th trial. First, generate fresh ciphertexts $\texttt{ct}_1$ and $\texttt{ct}_2$ encrypting $i+1$ and i. Next, generate $\texttt{ct}_3$ as the homomorphic addition of $\texttt{ct}_1$ and $\texttt{ct}_2$. Next, generate $\texttt{ct}_4$ as the homomorphic multiplication of $\texttt{ct}_3$ and $\texttt{ct}_2$. Finally, generate $\texttt{ct}_5$ by modulus switching $\texttt{ct}_4$ down to the next prime in the chain. We ran this evaluation over 10000 trials, using the SEAL default parameters n, Q, σ for the 128-bit security level for dimensions $n \in \{2048, 4096, 8192, 16384\}$. The SEAL default parameters for $n = 2048$ correspond to a chain of only one modulus, and hence we cannot perform modulus switching in this case. We used a plaintext modulus $t = 256$. Such a small plaintext modulus means that the values encrypted in our trials 'cover' the whole plaintext space and hence the assumption used in the noise bounds that m is a random plaintext is reasonable. To generate the plaintexts encoding $i+1$ and i, we used the default binary encoder. Table 3 reports on the results of this experiment.

In a second experiment, we repeated the above evaluation using a batch encoder. In each trial we generate two plaintexts, encoding the values j and $j+1$ for $j \in \{0, 1, \ldots, n\}$ respectively in each of the n slots. To enable batching, we changed the plaintext modulus to be $t = 65537$, a prime congruent to 1 modulo $2n$. All other parameters were kept the same. Table 4 reports on the

n	Enc			Add			Mult			ModSwitch		
	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$
2048	27.0	29.0	35.4	26.0	28.0	35.0	0.000	8.00	16.9	-	-	-
4096	81.0	83.0	90.0	80.0	82.0	89.1	51.0	61.0	69.8	31.0	33.0	50.2
8192	189	191	198	188	190	198	157	168	178	139	141	151
16384	408	410	418	407	409	417	375	386	396	358	360	365

Table 3. Binary encoder setting. The column $\overline{x}$ gives the observed mean of the invariant noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$. The column E gives an estimate of the noise growth using heuristic bounds obtained following our analysis. The remaining column gives an estimate of the noise growth using heuristic bounds obtained following an analysis as in [14].

n	Enc			Add			Mult			ModSwitch		
	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$
2048	19.0	21.0	27.4	18.0	20.0	27.0	0.000	0.00	1.00	-	-	-
4096	71.0	71.0	82.0	70.0	70.0	81.1	32.0	41.0	54.0	23.0	25.0	42.3
8192	179	179	190	178	178	190	139	148	161	131	133	143
16384	398	398	410	397	397	409	356	366	380	350	352	357

Table 4. Batching setting. The column $\overline{x}$ gives the observed mean of the invariant noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$. The column E gives an estimate of the noise growth using heuristic bounds obtained following our analysis. The remaining column gives an estimate of the noise growth using heuristic bounds obtained following an analysis as in [14].

results of this experiment for 10000 trials. We report on further experiments in the batching setting in Table 8 in Appendix E.

Tables 3 and 4 show that the heuristic bounds indeed hold: the observed mean noise is less than the estimated noise, so the observed mean noise budget is more than the estimate obtained using the heuristic bounds. This gives us confidence that the heuristic bounds will hold very often, and so can be used reliably to set parameters to ensure correctness. However, the bounds do not appear to be tight. Indeed, for encryption, the heuristic bound predicts 6 to 8 (respectively 6 to 12) fewer bits of remaining noise budget than the mean observed in Table 3 (respectively Table 4). This gap is compounded as the number of operations increases, reaching 8 to 17 (respectively 7 to 14) bits after multiplication in Table 3 (respectively Table 4, for $n = 4096$ and above). It appears that the gap reduces after modulus switching, with 8 or 9 fewer bits of remaining noise budget than the mean observed in both Table 3 and Table 4. Comparing to Table 2 we see that these trends are all similar to the HElib case. Finally, notice that while the new method tightens the bounds by up to 3 bits for BGV as seen in Tables 1 and 2, for FV the improvement is more dramatic. Indeed, the new analysis tightens the bounds by as much as 10 bits, as seen in Tables 4 and 3. This difference happens with the multiplication operation, and indeed is explained by looking at the multiplication bounds. We see that while the BGV one is very simple, the complexity of the FV one implies that the scheme has a much larger benefit from a tighter analysis.

5 Updated comparison between BGV and FV

In this section we compare the BGV and FV schemes following the methodology of a prior work by Costache and Smart [20]. We improve on the prior work in several aspects. Our first and most important improvement is to select parameters that achieve a security level $\lambda = 128$ according to the Homomorphic Encryption Standard [1]. In contrast, the prior work [20] relied on a security analysis by Lindner and Peikert [33], which has been shown to be incorrect [2, 3]. In fact, as shown in [19], FHE parameters which were estimated by [33] to have 80 bits of security had as little as 51 bits of security according to [3, 2]. Our second main improvement is to use a heuristic noise analysis following the methodology of Iliashenko [29]. Our experimental results in Sections 3 and 4 show that this analysis more closely represents the noise growth in implementations than the heuristic analysis that was used in [20]. Thirdly, our analysis allows for a more flexible modulus switching for FV compared to that in the prior work [20]. We discuss this in Appendix F.

5.1 Methodology and parameter selection

Our comparison uses the same homomorphic evaluation function as in [20]. We begin by guessing the dimension n. We go through a pre-determined circuit as follows: we take a fresh ciphertext, perform ζ additions, followed by a multiplication, and a relinearization. We then modulus switch down to the next prime in the chain, perform ζ additions, followed by a multiplication and relinearization, and so on. After modulus switching to the smallest prime, we check if we get a decryption error. If that is the case, we increase the guess, and repeat the procedure until decryption succeeds. Each of the circuits we consider in this work is parameterised by a number of additions ζ and a multiplicative depth L. Any circuit that is to be homomorphically evaluated consists of additions and/ or multiplications, thus this approach is as comprehensive as can be. We refer to the reader to [21] for real-life applications of such circuits. For the given circuit, and for a fixed level L, plaintext modulus t, and security level λ, our goal is find the smallest parameter set, in terms of ciphertext size in kilobytes, such that decryption succeeds.

The decision to compare BGV and FV based on ciphertext size is consistent with choices made in [20]. While we could have considered other criteria such as key size, it is ciphertexts which are sent over networks and computed on, thus a very large ciphertext could present the biggest overhead in an implementation. Therefore, we believe ciphertext size is the most relevant criterion.

We largely follow the parameter choices in [20]: we perform $\zeta = 8$ additions before each multiplication and we set the standard deviation $\sigma = 3.2$. We set the ring constant $c_m = 1$, as n (and hence m) is always a power of two. We consider a range of levels L of circuits, choosing $L \in \{2, 4, 6, \ldots, 30\}$. We set the parameters n and (top modulus) Q to achieve a security level $\lambda = 128$ according to the Homomorphic Encryption Standard [1], when $\sigma = 3.2$ and the secret follows a uniform ternary distribution. The possible pairs of n and Q are reproduced in Table 9 in Appendix G. To keep the comparison fair, we assume a uniform ternary distribution for the secret keys, as well for the ephemeral keys sampled in encryption, in both BGV and FV.

We first perform the comparison using plaintext modulus $t = 3$, which was shown to be optimal among integral bases for encoding by Costache *et al.* [21]. For such a small plaintext modulus, the analysis of [20] reports that FV outperforms BGV, although for larger plaintext moduli BGV performs better than FV. For our second comparison, we use the plaintext modulus $t = 256$, a choice slightly beyond the crossover point according to [20]. For our third comparison, we use the plaintext modulus $t = 32768$, which is well beyond the reported crossover point. In Appendix H we investigate even larger plaintext moduli.

5.2 Results, analysis and limitations

Table 5 presents the results of the comparison for plaintext modulus $t = 3$. We see that for most values of L, both BGV and FV required the same minimal values of n and Q to support the computation and hence the ciphertext sizes were the same. That is, as the level increases, the point at which we need to switch to the next parameter set is roughly the same for both schemes. However, for $L \in \{10, 18\}$ we see from Table 5 that BGV required a larger parameter set than FV. This would suggest that for small plaintext modulus, FV is sometimes preferable to BGV. This is in agreement with the findings of [20].

Table 6 presents the results of the comparison for plaintext modulus $t = 256$. Again, for most values of L, the ciphertext sizes were the same for both BGV and FV. However, for $L \in \{2, 4, 8, 28, 30\}$ we see from Table 6 that BGV required a larger parameter set than FV. Indeed, the computation for $L \in \{28, 30\}$ could not be supported for BGV using any parameter set. The results for plaintext modulus $t = 32768$, presented in Table 7, are similar. This would again suggest that FV is sometimes preferable to BGV, even for plaintext moduli larger than the crossover point reported in [20]. In Appendix H, we give further tables for larger plaintext moduli and see that BGV slightly outperforms FV after somewhere between $t = 2^{32}$ and $t = 2^{64}$. In most cases, even for plaintext moduli as large as $t = 2^{128}$, we can conclude from our results that the two schemes present only minor performance differences in terms of supporting a specific homomorphic evaluation.

We stress that this is a comparison of how the noise growth behaviour impacts correctness in the BGV and FV schemes: we ignore correctness issues coming from decoding failure. Our comparison is naturally limited in several other aspects. For example, we only consider a certain specific computation, for which we

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	4.75	6.77	8.77	8.77	10.8	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8
FV	4.75	6.77	8.77	8.77	8.77	10.8	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8

Table 5. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 3$.

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	6.77	8.77	8.77	10.8	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	-	-
FV	4.75	6.77	8.77	8.77	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8	12.8

Table 6. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 256$. The symbol '-' indicates that the computation was too large to be supported by any parameter set.

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	6.77	8.77	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	-	-	-	-
FV	6.77	8.77	8.77	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	-	-	-

Table 7. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 32768$. The symbol '-' indicates that the computation was too large to be supported by any parameter set.

do not attempt to make any scheme-specific optimisations that may be possible. Also, we note that while the choice of plaintext modulus $t = 3$ is optimal for integral bases, recent work has demonstrated the benefits of using non-integral bases [7, 13] or using t a polynomial rather than an integer [16].

6 Improving the heuristic-to-practical gap

In this section, we present additional supporting evidence for the statement "The worst-case heuristic approach is inadequate".

Different definitions of noise result in a similar gap. In a fresh FV encryption, the message m is scaled up by $\Delta = \lfloor q/t \rfloor$ to put it in the high-order bits. In decryption, we cancel Δ by multiplying by t/q, but this introduces a rounding term of the form $r_t(q) \cdot m$, since typically q is not exactly divisible by t. The invariant noise, defined such that $t/q \cdot (ct(s)) = m + v + at$, folds this rounding term into the noise. However, notice that this $r_t(q) \cdot m$ term is only introduced by the decryption process: this term is not a part of the noise that the ciphertext carries before a decryption is performed. Therefore, including this term in every intermediate ciphertext will lead to overestimates that compound. This discussion motivates us to define (see Appendix I) the *scaled inherent noise* for FV[10]. We determined heurisitic bounds for the scaled inherent noise and conducted experiments in SEAL similar to those described in Section 4. We found that while using the scaled inherent noise represents a slight improvement for modelling the noise in fresh ciphertexts, it does not significantly improve the heuristic-to-practical gap.

Bounding Gaussian variables inherently leads to loose bounds. Our approach to obtain heuristic bounds requires us to bound Gaussian random variables in the canonical embedding. For example, a Gaussian random variable e, with mean zero and standard deviation σ is bounded as $\|e\|^{\text{can}} \leq B \cdot \sigma_e$, for some B, where $\sigma_e = \sigma\sqrt{n}$. Following [20], we use $B = 6$, while HElib uses $B = 10$ as a default [28]. On the one hand, we never see $\|e\|^{\text{can}}$ this large in experiments, which is not surprising because the probability of $\|e\|^{\text{can}} > B \cdot \sigma_e$ is extremely low. On the other hand, to prove a heuristic bound of this type in theory, we need to ensure B is large enough (such as $B = 5$ or $B = 6$) to obtain a 'reasonable' failure probability. For example, we have $\text{erfc}(5) \approx 2^{-40}$, while $\text{erfc}(6) \approx 2^{-50}$. This means that we necessarily end up with looser bounds than we will observe in practice, in order to retain correctness in theory.

Alternative approaches also require Gaussian tail bounds. We could consider other approaches for obtaining heuristic noise bounds. For example, a δ-subgaussian approach [37] could be used, as was done in [35] for a BGV-like scheme. It was shown in [38] that tighter correctness bounds than presented in [35] could potentially be obtained by using a Central Limit Theorem (CLT) argument. Both the δ-subgaussian and CLT approaches ultimately show that due to the high dimensionality we can expect noise terms to have Gaussian tail bounds. Using these approaches we will encounter the same issue of requiring loose bounds in order to ensure correctness in the worst case.

An average-case analysis would be complicated by nonlinearity. The TFHE scheme [18] uses an appealing average-case approach to estimate noise growth, rather than worst-case bounds. In this approach, the coefficients of the noise in a TFHE ciphertext are modelled as independent subgaussians, and the variance of these subgaussians is traced through the homomorphic evaluation operations. This heuristic has been experimentally verified for the gate bootstrapping operation [18, Figure 10], showing in this case the noise in an output ciphertext can be modelled as a Gaussian of a certain variance. Moreover, every elementary operation in TFHE can be implemented via gate bootstrapping on a linear combination of ciphertexts [18, Table 1]. Hence, by linearity, all noises

[10] The noise is so named as it is equal to a scaling by t/q of the *inherent noise* [23].

in TFHE ciphertexts can be modelled as subgaussian and it is easy to follow through the analysis of the variances.

In contrast, in the case of BGV and FV, we have a nonlinear noise growth in multiplication and the situation becomes more complicated, as is noted for a similar scheme in [38, 39]. In [39] it was investigated how well the noise in a ciphertext obtained after several multiplications can be modelled as a Gaussian, for a BGV-like scheme. It was found that while a Central Limit argument could be used to approximate the noise in such a ciphertext as Gaussian, the quality of such an approximation would tend to decrease after many multiplications because the true distribution of the noise would have heavier and heavier tails. Hence it is not clear if an average-case approach as used in [18] would tightly model the noise growth in BGV or FV after many multiplications. Resolving this would be an interesting direction for future work.

References

[1] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, S. Lokam, D. Micciancio, D. Moody, T. Morrison, A. Sahai, and V. Vaikuntanathan. Homomorphic encryption security standard. Technical report, HomomorphicEncryption.org, 2018.

[2] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Mathematical Cryptology*, 9(3):169–203, 2015.

[3] Martin R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 103–129. Springer, Heidelberg, April / May 2017.

[4] Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Heidelberg, August 2016.

[5] A. Al Badawi, Y. Polyakov, K. Mi Mi Aung, B. Veeravalli, and K. Rohloff. Implementation and performance evaluation of RNS variants of the BFV homomorphic encryption scheme. Cryptology ePrint Archive, Report 2018/589, 2018.

[6] Jean-Claude Bajard, Julien Eynard, M. Anwar Hasan, and Vincent Zucca. A full RNS variant of FV like somewhat homomorphic encryption schemes. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016*, volume 10532 of *LNCS*, pages 423–442. Springer, Heidelberg, August 2016.

[7] Charlotte Bonte, Carl Bootland, Joppe W. Bos, Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Faster homomorphic function evaluation using non-integral base encoding. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 579–600. Springer, Heidelberg, September 2017.

[8] Joppe W. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In Martijn Stam, editor, *14th IMA International Conference on Cryptography and Coding*, volume 8308 of *LNCS*, pages 45–64. Springer, Heidelberg, December 2013.

[9] C. Boura, N. Gama, and M. Georgieva. Chimera: a unified framework for B/FV, TFHE and HEAAN fully homomorphic encryption and predictions for deep learning. Cryptology ePrint Archive, Report 2018/758, 2018.

[10] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 868–886. Springer, Heidelberg, August 2012.

[11] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012.

[12] M. Brenner, W. Dai, S. Halevi, K. Han, A. Jalali, M. Kim, K. Laine, A. Malozemoff, P. Paillier, Y. Polyakov, K. Rohloff, E. Savaş, and B. Sunar. A standard API for RLWE-based homomorphic encryption. Technical report, HomomorphicEncryption.org, 2017.

[13] Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Homomorphic SIM^{2}D operations: Single instruction much more data. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 338–359. Springer, Heidelberg, April / May 2018.

[14] H. Chen, K. Laine, and R. Player. Simple encrypted arithmetic library - SEAL v2.2. Technical report, 2017.

[15] Hao Chen, Kim Laine, and Rachel Player. Simple encrypted arithmetic library - SEAL v2.1. In Michael Brenner, Kurt Rohloff, Joseph Bonneau, Andrew Miller, Peter Y. A. Ryan, Vanessa Teague, Andrea Bracciali, Massimiliano Sala, Federico Pintore, and Markus Jakobsson, editors, *FC 2017 Workshops*, volume 10323 of *LNCS*, pages 3–18. Springer, Heidelberg, April 2017.

[16] Hao Chen, Kim Laine, Rachel Player, and Yuhou Xia. High-precision arithmetic in homomorphic encryption. In Nigel P. Smart, editor, *CT-RSA 2018*, volume 10808 of *LNCS*, pages 116–136. Springer, Heidelberg, April 2018.

[17] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song. Homomorphic encryption for arithmetic of approximate numbers. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 409–437. Springer, Heidelberg, December 2017.

[18] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène. TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, Apr 2019.

[19] A. Costache. *On the Practicality of Ring-Based Fully Homomorphic Encryption Schemes*. PhD thesis, University of Bristol, 2018.

[20] Ana Costache and Nigel P. Smart. Which ring based somewhat homomorphic encryption scheme is best? In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 325–340. Springer, Heidelberg, February / March 2016.

[21] Anamaria Costache, Nigel P. Smart, Srinivas Vivek, and Adrian Waller. Fixed-point arithmetic in SHE schemes. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016*, volume 10532 of *LNCS*, pages 401–422. Springer, Heidelberg, August 2016.

[22] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, Heidelberg, August 2012.

[23] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. `http://eprint.iacr.org/2012/144`.

[24] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.

[25] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 465–482. Springer, Heidelberg, April 2012.

[26] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867. Springer, Heidelberg, August 2012.

[27] Shai Halevi, Yuriy Polyakov, and Victor Shoup. An improved RNS variant of the BFV homomorphic encryption scheme. Cryptology ePrint Archive, Report 2018/117, 2018. `https://eprint.iacr.org/2018/117`.

[28] HElib. `https://github.com/shaih/HElib`, January 2019.

[29] I. Iliashenko. *Optimisations of fully homomorphic encryption*. PhD thesis, KU Leuven, 2019.

[30] M. Kim and K. Lauter. Private genome analysis through homomorphic encryption. *BMC Medical Informatics and Decision Making*, 15(5):S3, Dec 2015.

[31] Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched NTRU parameters. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 3–26. Springer, Heidelberg, April / May 2017.

[32] Tancrède Lepoint and Michael Naehrig. A comparison of the homomorphic encryption schemes FV and YASHE. In David Pointcheval and Damien Vergnaud, editors, *AFRICACRYPT 14*, volume 8469 of *LNCS*, pages 318–335. Springer, Heidelberg, May 2014.

[33] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, Heidelberg, February 2011.

[34] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Howard J. Karloff and Toniann Pitassi, editors, *44th ACM STOC*, pages 1219–1234. ACM Press, May 2012.

[35] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. Cryptology ePrint Archive, Report 2013/293, 2013. `http://eprint.iacr.org/2013/293`.

[36] C. Aguilar Melchor, M.-O. Killijian, C. Lefebvre, and T. Ricosset. A comparison of the homomorphic encryption libraries HElib, SEAL and FV-NFLlib. In J.-L. Lanet and C. Toma, editors, *Innovative Security Solutions for Information Technology and Communications - 11th International Conference, SecITC 2018*, volume 11359 of *LNCS*, pages 425–442. Springer, 2018.

[37] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

[38] S. Murphy and R. Player. A central limit framework for Ring-LWE decryption. Cryptology ePrint Archive, Report 2019/452, 2019.

[39] S. Murphy and R. Player. Discretisation and product distributions in ring-lwe. MathCrypt 2019, to appear, 2019.

[40] R. Player. *Parameter selection in lattice-based cryptography*. PhD thesis, Royal Holloway, University of London, 2018.

[41] Microsoft SEAL (release 3.4). `https://github.com/Microsoft/SEAL`, October 2019. Microsoft Research, Redmond, WA.

A The BGV scheme

In this section we introduce the BGV scheme [11]. The BGV scheme is comprised of the `SecretKeyGen`, `PublicKeyGen`, `EvaluationKeyGen`, `Encrypt`, `Decrypt`, `Add`, `Multiply`, `Relinearize`, and `ModSwitch` algorithms.

In the `ModSwitch` algorithm, we describe switching from a modulus q to a modulus p where, for correctness, we require that $p = q = 1 \mod t$ [11, 25]. For the algorithm as described here, we also need $p \mid q$, which will be the case when moving down the chain of moduli.

- $\texttt{SecretKeyGen}(\lambda)$: Sample $s \leftarrow S$ and output $\texttt{sk} = s$.
- $\texttt{PublicKeyGen}(\texttt{sk})$: Set $s = \texttt{sk}$ and sample $a \leftarrow R_q$ uniformly at random and $e \leftarrow \chi$. Output $\texttt{pk} = ([-(as + te)]_q, a)$.
- $\texttt{EvaluationKeyGen}(\texttt{sk}, w)$: Set $s = \texttt{sk}$. For $i \in \{0, \ldots, \ell\}$, sample $a_i \leftarrow R_q$ uniformly at random and $e_i \leftarrow \chi$. Output $\texttt{evk} = \big([-(a_i s + te_i) + w^i s^2]_q, a_i\big)$.
- $\texttt{Encrypt}(\texttt{pk}, m)$: For the message $m \in R_t$. Let $\texttt{pk} = (p_0, p_1)$, sample $u \leftarrow S$ and $e_1, e_2 \leftarrow \chi$. Output $\texttt{ct} = ([m + p_0 u + te_1]_q, [p_1 u + te_2]_q)$.
- $\texttt{Decrypt}(\texttt{sk}, \texttt{ct})$: Let $s = \texttt{sk}$ and $\texttt{ct} = (c_0, c_1)$. Output $m' = \left[[c_0 + c_1 s]_q\right]_t$.
- $\texttt{Add}(\texttt{ct}_0, \texttt{ct}_1)$: Output $\texttt{ct} = ([\texttt{ct}_0[0] + \texttt{ct}_1[0]]_q, [\texttt{ct}_0[1] + \texttt{ct}_1[1]]_q)$.
- $\texttt{Multiply}(\texttt{ct}_0, \texttt{ct}_1)$: Set $c_0 = \left[\texttt{ct}_0[0]\texttt{ct}_1[0]\right]_q$, $c_1 = \left[\texttt{ct}_0[0]\texttt{ct}_1[1] + \texttt{ct}_0[1]\texttt{ct}_1[0]\right]_q$, and $c_2 = \left[\texttt{ct}_0[1]\texttt{ct}_1[1]\right]_q$. Output $\texttt{ct} = (c_0, c_1, c_2)$.
- $\texttt{Relinearize}(\texttt{ct}, \texttt{evk})$: Let $\texttt{ct}[0] = c_0$, $\texttt{ct}[1] = c_1$ and $\texttt{ct}[2] = c_2$. Let $\texttt{evk}[i][0] = [-(a_i s + te_i) + w^i s^2]_q$ and $\texttt{evk}[i][1] = a_i$. Express c_2 in base w as $c_2 = \sum_{i=0}^{\ell} c_2^{(i)} w^i$. Set $c_0' = c_0 + \sum_{i=0}^{\ell} \texttt{evk}[i][0] c_2^{(i)}$, and $c_1' = c_1 + \sum_{i=0}^{\ell} \texttt{evk}[i][1] c_2^{(i)}$. Output $\texttt{ct}' = (c_0', c_1')$.
- $\texttt{ModSwitch}(\texttt{ct}, p)$: Let $\texttt{ct} = (c_0, c_1)$. Fix δ_i such that $\delta_i = -c_i \pmod{\frac{q}{p}}$ and $\delta_i = 0 \pmod t$. Set $c_0' = \frac{p}{q}(c_0 + \delta_0)$ and $c_1' = \frac{p}{q}(c_1 + \delta_1)$. Output $\texttt{ct} = (c_0', c_1')$.

B The FV scheme

In this section we introduce the FV scheme [23]. To simplify presentation we follow the 'textbook' FV scheme as presented by Fan and Vercauteren [23], in which ciphertexts are of size 2: that is, they are a tuple of 2 elements in R_q. This is denoted $\texttt{ct} = (\texttt{ct}[0], \texttt{ct}[1])$. In particular, we will always assume that any output[11] of `Multiply` is immediately given as an input to `Relinearize`, and so we only define the other algorithms for ciphertexts of size 2. This is in contrast to, for example, the SEAL [41] implementation which allows ciphertexts to grow in size and uses generalisations of algorithms accordingly.

We do however deviate from the original description of FV by also defining a modulus switching operation, as was done in Costache and Smart [20]. In particular, we describe switching from a modulus q to a modulus p.

We now define the `SecretKeyGen`, `PublicKeyGen`, `EvaluationKeyGen`, `Encrypt`, `Decrypt`, `Add`, `Multiply`, `Relinearize`, and `ModSwitch` algorithms. In

[11] Abusing notation, we still denote such an output by `ct`.

order to define Encrypt, we must first define $\Delta = \left\lfloor \frac{q}{t} \right\rfloor$, where q is the current ciphertext modulus, and t is the plaintext modulus. We also define $r_t(q)$ as the remainder of q on division by t, so that $q = \Delta t + r_t(q)$.

- $\mathtt{SecretKeyGen}(\lambda)$: Sample $s \leftarrow S$ and output $\mathtt{sk} = s$.
- $\mathtt{PublicKeyGen}(\mathtt{sk})$: Set $s = \mathtt{sk}$ and sample $a \leftarrow R_q$ uniformly at random and $e \leftarrow \chi$. Output $\mathtt{pk} = ([-(as + e)]_q, a)$.
- $\mathtt{EvaluationKeyGen}(\mathtt{sk}, w)$: Set $s = \mathtt{sk}$. For $i \in \{0, \dots, \ell\}$, sample $a_i \leftarrow R_q$ uniformly at random and $e_i \leftarrow \chi$. Output $\mathtt{evk} = \big([-(a_i s + e_i) + w^i s^2]_q, a_i\big)$.
- $\mathtt{Encrypt}(\mathtt{pk}, m)$: For the message $m \in R_t$. Let $\mathtt{pk} = (p_0, p_1)$, sample $u \leftarrow S$ and $e_1, e_2 \leftarrow \chi$. Output $\mathtt{ct} = ([\Delta m + p_0 u + e_1]_q, [p_1 u + e_2]_q)$.
- $\mathtt{Decrypt}(\mathtt{sk}, \mathtt{ct})$: Let $s = \mathtt{sk}$ and $\mathtt{ct} = (c_0, c_1)$. Output $m' = \left[\left\lfloor \frac{t}{q}[c_0 + c_1 s]_q \right\rceil\right]_t$.
- $\mathtt{Add}(\mathtt{ct}_0, \mathtt{ct}_1)$: Output $\mathtt{ct} = ([\mathtt{ct}_0[0] + \mathtt{ct}_1[0]]_q, [\mathtt{ct}_0[1] + \mathtt{ct}_1[1]]_q)$.
- $\mathtt{Multiply}(\mathtt{ct}_0, \mathtt{ct}_1)$: Compute $c_0 = \left[\left\lfloor \frac{t}{q} \mathtt{ct}_0[0]\mathtt{ct}_1[0] \right\rceil\right]_q$,
  $c_1 = \left[\left\lfloor \frac{t}{q} \left(\mathtt{ct}_0[0]\mathtt{ct}_1[1] + \mathtt{ct}_0[1]\mathtt{ct}_1[0]\right) \right\rceil\right]_q$, and $c_2 = \left[\left\lfloor \frac{t}{q} \mathtt{ct}_0[1]\mathtt{ct}_1[1] \right\rceil\right]_q$.
  Output $\mathtt{ct} = (c_0, c_1, c_2)$.
- $\mathtt{Relinearize}(\mathtt{ct}, \mathtt{evk})$: Let $\mathtt{ct}[0] = c_0$, $\mathtt{ct}[1] = c_1$ and $\mathtt{ct}[2] = c_2$. Let $\mathtt{evk}[i][0] = [-(a_i s + e_i) + w^i s^2]_q$ and $\mathtt{evk}[i][1] = a_i$. Express c_2 in base w as $c_2 = \sum_{i=0}^{\ell} c_2^{(i)} w^i$. Set $c_0' = [c_0 + \sum_{i=0}^{\ell} \mathtt{evk}[i][0] c_2^{(i)}]_q$, and $c_1' = [c_1 + \sum_{i=0}^{\ell} \mathtt{evk}[i][1] c_2^{(i)}]_q$. Output $\mathtt{ct}' = (c_0', c_1')$.
- $\mathtt{ModSwitch}(\mathtt{ct}, p)$: Let $\mathtt{ct}[0] = c_0$ and $\mathtt{ct}[1] = c_1$. Set $c_0' = \left[\left\lfloor \frac{p}{q} c_0 \right\rceil\right]_p$ and $c_1' = \left[\left\lfloor \frac{p}{q} c_1 \right\rceil\right]_p$. Output $\mathtt{ct}' = (c_0', c_1')$.

C BGV relinearization

Let $\mathtt{ct}$ be a BGV ciphertext encrypting m and having noise v. Let $\mathtt{ct}_{\mathrm{relin}}$ be the ciphertext obtained by the relinearization of $\mathtt{ct}$. The noise is given by $v_{\mathrm{relin}} = v + t\sum_{i=0}^{\ell} e_i c_2^{(i)}$. We can bound

$$\|v_{\mathrm{relin}}\|^{\mathrm{can}} \leq \|v\|^{\mathrm{can}} + t \left\| \sum_{i=0}^{\ell} e_i c_2^{(i)} \right\|^{\mathrm{can}} .$$

Let $y = \sum_{i=0}^{\ell} e_i c_2^{(i)}$. Note that the polynomials e_i and $c_2^{(i)}$ have the same variance V_e and $V_{c_2^{(i)}}$ for all i. Then y is a polynomial with coefficients of variance

$$V_y = \sum_{i=0}^{\ell} V_{e_i c_2^{(i)}} = (\ell+1) V_{e_i c_2^{(i)}} = (\ell+1) n V_e V_{c_2^{(i)}} = (\ell+1) n \sigma^2 \frac{w^2}{12} .$$

So, by the same argument as in [20],

$$\|v_{\mathrm{relin}}\|^{\mathrm{can}} \leq \|v\|^{\mathrm{can}} + 6t\sqrt{n V_y} = \|v\|^{\mathrm{can}} + 6t\sqrt{n}\sqrt{(\ell+1) n \sigma^2 \frac{w^2}{12}} .$$

D FV heuristic bounds

Encrypt [29]: Let `ct` be a fresh FV encryption of a message $m \in R_t$. With high probability, the invariant noise v in `ct` satisfies

$$\|v\|^{\mathrm{can}} \leq 6 \cdot \sqrt{n} \cdot \frac{t}{q} \cdot \sqrt{\frac{r_t(q)^2}{12} + \sigma^2 \left(\frac{4n}{3} + 1\right)}\,. \tag{1}$$

Let $V_s = \frac{2}{3}$ be the variance of a coefficient of a polynomial chosen from the secret distribution. Let $V_e = \sigma^2$ be the variance of a coefficient of a polynomial chosen from the error distribution. Let $V_m = \frac{t^2}{12}$ be the variance of a coefficient of a plaintext polynomial (i.e. we model plaintexts as being chosen uniformly at random from the plaintext space). As shown in e.g. [29, 40], the invariant noise v in a fresh ciphertext is $v = \frac{t}{q}\left(\frac{-r_t(q)}{t} m - eu + e_1 + e_2 s\right)$. This is a polynomial with coefficients of variance

$$\begin{aligned} V_v &= \frac{r_t(q)^2}{q^2} V_m + \frac{t^2}{q^2}\left(nV_eV_u + V_{e_1} + nV_{e_2}V_s\right) \\ &= \frac{r_t(q)^2}{q^2} \cdot \frac{t^2}{12} + \frac{t^2}{q^2}\left(n \cdot \frac{2}{3} \cdot \sigma^2 + \sigma^2 + n \cdot \frac{2}{3} \cdot \sigma^2\right) \\ &= \frac{t^2}{q^2}\left(\frac{r_t(q)^2}{12} + \sigma^2\left(\frac{4n}{3} + 1\right)\right)\,. \end{aligned}$$

Hence

$$\|v\|^{\mathrm{can}} \leq 6 \cdot \sqrt{n} \cdot \frac{t}{q} \cdot \sqrt{\frac{r_t(q)^2}{12} + \sigma^2 \left(\frac{4n}{3} + 1\right)}\,.$$

Add [14, 29]: Let $\mathtt{ct}_1$ and $\mathtt{ct}_2$ be two FV ciphertexts encrypting $m_1, m_2 \in R_t$, and having invariant noises v_1, v_2, respectively. Then the invariant noise v_{add} in their sum $\mathtt{ct}_{\mathrm{add}}$ satisfies $\|v_{\mathrm{add}}\|^{\mathrm{can}} \leq \|v_1\|^{\mathrm{can}} + \|v_2\|^{\mathrm{can}}$.

Multiply [29]: Let $\mathtt{ct}_1$ be an FV ciphertext of size 2 encrypting m_1 with invariant noise v_1, and let $\mathtt{ct}_2$ be an FV ciphertext of size 2 encrypting m_2 with invariant noise v_2. With high probability, the invariant noise v_{mult} in the product $\mathtt{ct}_{\mathrm{mult}}$ satisfies

$$\begin{aligned} \|v_{\mathrm{mult}}\|^{\mathrm{can}} &\leq 3\,\|v_1\|^{\mathrm{can}} \cdot \|v_2\|^{\mathrm{can}} \\ &\quad + \left(\|v_1\|^{\mathrm{can}} + \|v_2\|^{\mathrm{can}}\right) \cdot t \cdot \left(\sqrt{3n + 2n^2}\right) \\ &\quad + \frac{t}{q} \cdot \sqrt{3n + 2n^2 + (4/3)n^3}\,. \end{aligned}$$

As shown in e.g. [29, 40], the invariant noise v_{mult} in the product $\mathtt{ct}_{\mathrm{mult}}$ is given by

$$v_{\mathrm{mult}} = v_1 \cdot v_2 + m_1 \cdot v_2 + m_2 \cdot v_1 + tv_1 \cdot a_2 + tv_2 \cdot a_1 + \frac{t}{q} \cdot \sum_{i=0}^{2} \epsilon_i s^i$$

$$= v_1 \cdot v_2 + v_1 \left(m_2 + t \cdot a_2\right) + v_2 \left(m_1 + t \cdot a_1\right) + \frac{t}{q} \cdot \sum_{i=0}^{2} \epsilon_i s^i.$$

Its norm can be bounded as

$$\begin{aligned} \|v_{\mathrm{mult}}\|^{\mathrm{can}} \leq \|v_1\|^{\mathrm{can}} \cdot \|\cdot v_2\|^{\mathrm{can}} + \|v_1\|^{\mathrm{can}} \cdot \|m_2 + t \cdot a_2\|^{\mathrm{can}} \\ + \|v_2\|^{\mathrm{can}} \cdot \|m_1 + t \cdot a_1\|^{\mathrm{can}} + \left\| \frac{t}{q} \cdot \sum_{i=0}^{2} \epsilon_i s^i \right\|^{\mathrm{can}}. \end{aligned}$$

We now bound the term $m_2 + t \cdot a_2$. We know that

$$a_i t + m_i = \frac{t}{q} \cdot \mathtt{ct}_i(s) - v_i.$$

Consider the polynomial $z = \frac{t}{q}\mathtt{ct}_i(s)$. Let $V_c = \frac{q^2}{12}$ be the variance of a coefficient of a ciphertext polynomial and $V_m = \frac{t^2}{12}$ be the variance of a coefficient of a plaintext polynomial. Let $V_s = \frac{2}{3}$ be the variance of a coefficient of a polynomial chosen from the secret distribution. The polynomial z has coefficients of variance

$$V_z = \frac{t^2}{q^2}(V_{c_0} + nV_{c_1}V_s) = \frac{t^2}{q^2}\left(\frac{q^2}{12} + n \cdot \frac{q^2}{12} \cdot \frac{2}{3}\right) = t^2\left(\frac{1}{12} + \frac{n}{18}\right).$$

Hence

$$\left\|\frac{t}{q}\mathtt{ct}_i(s)\right\|^{\mathrm{can}} \leq 6t\sqrt{\frac{n}{12} + \frac{n^2}{18}} = t\sqrt{3n + 2n^2}.$$

It follows that

$$\begin{aligned} \|tv_1 \cdot a_2\|^{\mathrm{can}} &\leq \|v_1\|^{\mathrm{can}} \|ta_2\|^{\mathrm{can}} \\ &\leq \|v_1\|^{\mathrm{can}} \left(\left\|\frac{t}{q}\mathtt{ct}_2(s)\right\|^{\mathrm{can}} + \|v_2\|^{\mathrm{can}}\right) \\ &\leq \|v_1\|^{\mathrm{can}} \left(t\sqrt{3n^2n^2} + \|v_2\|^{\mathrm{can}}\right) \\ &= \left(t\sqrt{3n + 2n^2}\right) \|v_1\|^{\mathrm{can}} + \|v_1\|^{\mathrm{can}} \cdot \|v_2\|^{\mathrm{can}}. \end{aligned}$$

By the same argument as for the previous term, we bound $\|tv_2 \cdot a_1\|^{\mathrm{can}}$ as

$$\|tv_2 \cdot a_1\|^{\mathrm{can}} \leq \left(t\sqrt{3n + 2n^2}\right) \|v_2\|^{\mathrm{can}} + \|v_1\|^{\mathrm{can}} \cdot \|v_2\|^{\mathrm{can}}.$$

Finally, we look at the term $\left\|\frac{t}{q}\sum_{i=0}^{2}\epsilon_i s^i\right\|^{\mathrm{can}}$. Consider the polynomial $A = \frac{t}{q}\sum_{i=0}^{2}\epsilon_i s^i$. Let $V_\epsilon = \frac{1}{12}$ be the variance of a coefficient of a polynomial chosen uniformly at random from $(-1/2, 1/2]$. The polynomial A has coefficients with variance:

$$V_A = \frac{t^2}{q^2}\left(V_\epsilon + nV_\epsilon V_s + n^2 V_\epsilon V_s V_s\right)$$

$$= \frac{t^2}{q^2}\left(\frac{1}{12} + n \cdot \frac{1}{12} \cdot \frac{2}{3} + n^2 \cdot \frac{1}{12} \cdot \frac{2}{3} \cdot \frac{2}{3}\right)$$
$$= \frac{t^2}{12q^2}\left(1 + \frac{2n}{3} + \frac{4n^2}{9}\right).$$

Hence we have

$$\|A\|^{\text{can}} \leq 6\sqrt{n} \cdot \frac{t}{q} \cdot \sqrt{\frac{1}{12}} \cdot \sqrt{1 + \frac{2n}{3} + \frac{4n^2}{9}}$$
$$= \sqrt{3n} \cdot \frac{t}{q} \cdot \sqrt{1 + \frac{2n}{3} + \frac{4n^2}{9}}$$
$$= \frac{t}{q}\sqrt{3n + 2n^2 + \frac{4n^3}{3}}.$$

Relinearize [29]: Let $\texttt{ct}$ be an FV ciphertext encrypting m and having invariant noise v. Let $\texttt{ct}_{\text{relin}}$ be the ciphertext obtained by the relinearization of $\texttt{ct}$. Then with high probability, the invariant noise v_{relin} in $\texttt{ct}_{\text{relin}}$ satisfies

$$\|v_{\text{relin}}\|^{\text{can}} \leq \|v\|^{\text{can}} + \frac{t}{q} n\sigma w\sqrt{3(\ell+1)}.$$

The working is analogous to the one for the BGV relinearization formula.

E Additional experiments for SEAL in the batching setting

In Section 4.2 we observe a heuristic-to-practical gap in the SEAL experiments in the batching setting (Table 4). One reason why this may exist is that the noise bounds assume a random plaintext, but in each slot we encrypt values between 0 and n only. We always have $n < \frac{t}{4}$ and so these values do not 'cover' the whole possible plaintext space. As such, they cannot be considered as random. We repeat the experiment, generating the two plaintexts differently. In each trial, random values r_1 and r_2 are chosen from the interval $[0, t]$ and the values $(r_1, r_1 + 1, \ldots, r_1 + n)$ and $(r_2, r_2 + 1, \ldots, r_2 + n)$ respectively are batch encoded into the plaintexts. All parameters, including the plaintext modulus $t = 65537$, are kept the same as for the previous experiments, as described in Section 4.2. Table 8 reports on the results of this experiment for 10000 trials. We see that there is essentially no difference from Table 4.

F Flexible modulus switching for FV

In this section we comment on the analysis of modulus switching in [20]. Recall that p_i are list of primes forming the chain of moduli and t is the plaintext modulus. For the BGV scheme all the p_i must be chosen so that $p_i = 1 \pmod t$,

n	Enc			Add			Mult			ModSwitch		
	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$	[14]	E	$\overline{x}$
2048	19.0	21.0	27.3	18.0	20.0	27.0	0.000	0.00	0.943	-	-	-
4096	71.0	71.0	82.0	70.0	70.0	81.1	32.0	41.0	54.3	23.0	25.0	42.6
8192	179	179	190	178	178	190	139	148	161	131	133	143
16384	398	398	410	397	397	409	356	366	381	350	352	357

Table 8. Batching setting with random plaintexts. The column $\overline{x}$ gives the observed mean of the invariant noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$. The column E gives an estimate of the noise growth using heuristic bounds obtained following our analysis. The remaining column gives an estimate of the noise growth using heuristic bounds obtained following an analysis as in [14].

to ensure correctness in the modulus switching operation. Choosing such p_i is not necessary for FV, yet was listed as a requirement in [20]. The experimental results in [20] were only concerned with the bit size of the primes in the chain of moduli, and so in Section 5 we cannot explicitly improve their comparison in this aspect. However, we note that this constraint on the p_i could make the FV performance with modulus switching look poor compared to BGV. Specifically, imposing this constraint may result in unfairly large parameters for FV. In fact, an application implemented with BGV may end up requiring larger parameters than the same application implemented with FV, because of the restricted choice of p_i.

G Recommended parameter sets in the HE Standard

n	2048	4096	8192	16384	32768
$\log Q$	54	109	218	438	881

Table 9. Pairs of the parameters n and Q (given as its bitsize $\log Q$) used in our comparison (Section 5). These parameters achieve a security level $\lambda = 128$ according to the Homomorphic Encryption Standard [1, Table 1], when $\sigma = 3.2$ and the secret follows a uniform ternary distribution.

H Comparison of BGV and FV for very large plaintext moduli

In Tables 10, 11 and 12 we present the results of the comparison of BGV and FV in Section 5 for plaintext moduli $t = 2^{32}$, $t = 2^{64}$ and $t = 2^{128}$ respectively. For $t = 2^{32}$, depending on the choice of level, sometimes BGV outperforms FV and

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	6.77	8.77	10.8	12.8	12.8	12.8	12.8	-	-	-	-	-	-	-	-
FV	8.77	10.8	10.8	10.8	12.8	12.8	12.8	12.8	-	-	-	-	-	-	-

Table 10. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 2^{32}$. The symbol '-' indicates that the computation was too large to be supported by any parameter set.

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	8.77	10.8	12.8	12.8	12.8	-	-	-	-	-	-	-	-	-	-
FV	10.8	10.8	12.8	12.8	-	-	-	-	-	-	-	-	-	-	-

Table 11. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 2^{64}$. The symbol '-' indicates that the computation was too large to be supported by any parameter set.

Scheme	Level L														
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
BGV	10.8	12.8	-	-	-	-	-	-	-	-	-	-	-	-	-
FV	10.8	12.8	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 12. Logarithm to base 2 of the minimal ciphertext size in kilobytes required in the BGV and FV schemes to support the described homomorphic evaluation for L levels, for plaintext modulus $t = 2^{128}$. The symbol '-' indicates that the computation was too large to be supported by any parameter set.

sometimes vice versa. For $t = 2^{64}$, we see that BGV outperfoms FV. In all cases there is only a slight difference between the schemes. Indeed, for $t = 2^{128}$ we saw no difference, although for this very large t very few levels could be supported.

I The scaled inherent noise for FV

Definition 4 (Scaled inherent noise). *Let $q = \Delta t + r_t(q)$. Let $\mathtt{ct} = (c_0, c_1)$ be an FV ciphertext encrypting the message $m \in R_t$. Its* scaled inherent noise *v is the polynomial with the smallest infinity norm such that, for some integer coefficient polynomial a,*

$$\frac{t}{q}\mathtt{ct}(s) = \frac{t}{q}\left(c_0 + c_1 s\right) = \frac{t}{q}\Delta m + v + at\,.$$