Homomorphic noise growth in practice: comparing BGV and FV

Anamaria Costache[1], Kim Laine[2], and Rachel Player[3]

[1] Intel AI Research, San Diego, USA ** anamaria.costache@intel.com
[2] Microsoft Research, USA kim.laine@microsoft.com
[3] Royal Holloway, University of London, UK rachel.player@rhul.ac.uk

Abstract. The purpose of this paper is to provide a comprehensive analysis and side-by-side comparison of the noise growth behaviour in the BGV and FV somewhat homomorphic encryption schemes, both heuristically and in their implementations in the libraries HElib and SEAL, respectively. We run extensive experiments in HElib and SEAL to compare the heuristic noise growth to the noise growth in practice. From the experiments, we observe that for both schemes, the heuristic bounds are not tight. We attempt to improve the tightness of the bounds in a number of ways, including the definition of new notions of noise, such as the invariant noise for BGV and the scaled inherent noise for FV. This does not significantly tighten the bounds, thus we conclude that the current heuristic bounds are the best possible in terms of a theoretical analysis. As an additional contribution, we update the comparison between the two schemes presented by Costache and Smart [22], and find that BGV has a slight advantage over FV. Thus, the conclusions of [22] still hold, although the differences between BGV and FV are less dramatic.

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 [29], 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

** Part of this work was done at University of Bristol, UK

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

In this work, we consider the noise growth behaviour of the somewhat homomorphic encryption schemes BGV [12] and FV[4] [28]. We present the heuristic upper bounds for the noise growth in these schemes that have appeared in the literature, and then perform experiments to determine how tightly these bound the noise growth in practice. We consider other possible definitions of noise and we argue that these bounds, while loose compared to the average noise observed in ciphertexts in implementations, are the best possible in terms of a theoretical analysis. We also perform an updated comparison of BGV and FV, following the methodology of Costache and Smart [22].

In the first part of this paper, we compare heuristic noise estimates for the noise growth of both the BGV and FV schemes with the actual noise observed in ciphertexts in their implementation in the HElib [33] and SEAL [46] libraries, respectively. The first HElib noise results concern the growth of the *critical quantity* [22] 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* [16] for FV that is implemented in SEAL. The results in terms of this 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.

In these experiments, for both BGV and FV, we notice a rather large gap between the heuristic estimates for the noise growth and the practical noise growth. We consider three possible ways to reduce this gap, which we refer to as the *heuristic-to-practical* gap. The first is to try to improve the methodology used to obtain the heuristic bounds themselves, for example by tightening the bound on a Gaussian error distribution. In particular, following [22], we had bounded the Gaussian errors by 6σ, while HElib uses 10σ in their heuristic analysis. We argue in Section 7.2 that we cannot significantly improve on the 6σ bound without losing correctness.

Secondly, we introduce a new notion of noise for BGV: the *invariant noise*, so-called as it is the analogue of the invariant noise for FV. We update all the noise analyses (encryption, addition, multiplication, relinearization and modulus

[4] FV is based on a scheme of Brakerski [11] and hence is sometimes referred to as BFV.

switching) for the invariant noise. We then define and implement an invariant noise budget for BGV in HElib, and investigate whether this better models noise behaviour in HElib. We present the results in Table 5, and find that using the invariant noise rather than the critical quantity does not significantly reduce the heuristic-to-practical gap.

Thirdly, we consider the FV scheme. In an FV encryption with ciphertext modulus q and plaintext modulus t, the message is multiplied by a factor Δ, which is such that $q = \Delta t + r_t(q)$. We notice from previous theoretical analyses that the remainder term $r_t(q)$ can introduce cross terms in the operation bounds, which are not found in practice and do not impact the noise, except for the final decryption. Indeed, we can think of the contribution from this term as noise introduced by the decryption process. To remedy the situation, we introduce a new notion of noise for FV. We call this the *scaled inherent noise*, as it is a scaling by t/q of the *inherent noise* as originally defined in [28]. Again, we update all the noise analyses, and implement this new noise in SEAL. We investigate whether this better models noise behaviour in SEAL, and present the results in Tables 6 and 7, for the binary encoding and batch settings, respectively. While we do observe a slight improvement in the heuristic-to-practical gap, it is only slight, and the gap remains significant.

We conclude that any improvement of this gap, and by extension any accurate noise growth analysis has to not only be scheme-specific, but also implementation-specific. That is, these heuristic bounds are the best possible in terms of a theoretical analysis, and we must take a different approach in order to obtain bounds that more closely model the noise growth behaviour that we see in practice.

In the second part of the paper, we perform a comparison of BGV and FV using the methodology of Costache and Smart [22]. We update the work [22] in several aspects, the most notable of which is an up-to-date security analysis conforming to HE standards [1]. Secondly, we use a different notion of noise for FV, namely the invariant noise. Thirdly, our analysis 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. Finally, we no longer consider the NTRU-based schemes YASHE [9] and NTRU [27, 38]. It has since been shown that such schemes may be vulnerable to attacks in 'overstretched' parameter settings of interest [3, 18, 35] and as a consequence most implementations of NTRU-based homomorphic encryption schemes are not currently maintained. In contrast, the BGV and FV schemes are implemented in several actively maintained homomorphic encryption libraries, including PALISADE [44] as well as SEAL and HElib.

Our results, summarised in Table 9, show that while the conclusions of [22] still hold, they are far less dramatic. We recall that a comparison of the two schemes as implemented in their libraries is limited, therefore it seems that the difference could simply be due to the different choices made in the two libraries. In particular, the secret key distribution (ternary uniform for SEAL, binary with small Hamming weight in HElib) could very well account for this small discrepancy.

We conclude that the two schemes in their implementation present only minor performance differences in terms of supporting a specific homomorphic evaluation. That is, we expect that a computation supported in SEAL by a particular parameter set would be supported in HElib with the same parameter set. 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 what library to use.

The main contribution of this paper is to show that the updated theoretical analysis we present is the best possible. We have investigated many possible definitions of noise, including new ones, for both schemes BGV and FV. The fact that these do not significantly improve the heuristic-to-practical gap is evidence for our main conclusion: that any hope of improving the heuristic-to-practical gap lies in an analysis that is *both* scheme and implementation-specific.

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[5]. Indeed, the security standard [1] from the standardisation consortium explicitly mentions the comparison of BGV and FV as an open problem, and motivates the present work.

After completing the RLWE 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 [13]. 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, including BEHZ-FV [6] and HPS-FV [32]. Al Badawi *et al.* [5] conclude from experiments that BEHZ-FV has worse noise growth in practice than HPS-FV, and call for further study on BEHZ-FV noise growth, which further motivates the present work.

Heuristic upper bounds for the invariant noise growth in FV that are similar to those presented here have also been presented by Chen *et al.* [17] and in documentation [16] for previous versions of SEAL.

Apart from that of Costache and Smart [22], other previous comparisons of homomorphic encryption schemes include a work of Lepoint and Naehrig [36] comparing FV and YASHE, and a work of Kim and Lauter [34] comparing BGV and YASHE.

[5] HomomorphicEncryption.org

In the present work we do not consider newer schemes such as CKKS [20] or TFHE [21], which come with entirely different trade-offs. Of these, CKKS seems more feasible to include in a meaningful comparison with BGV and FV, but TFHE is quite fundamentally different. Chimera [10] 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

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, plaintext modulus t and the chain of moduli correspond to 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 $R_t = \mathbb{Z}_t[x]/(x^n + 1)$.

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 as originally described [28] the distribution S is the uniform distribution on the space $R_2 = \{0, 1\}[x]/(x^n + 1)$. In the SEAL implementation [46] the distribution S is the uniform distribution on the space $R_3 = \{-1, 0, 1\}[x]/(x^n + 1)$. In the BGV scheme as originally described [12], the distribution S is the same as the error distribution χ. In the HElib implementation [33], S is parameterised by a Hamming weight h and chooses a vector uniformly among the set of polynomials in the space R_3 having exactly h nonzero coefficients.

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.

2.2 Canonical embedding norm

Following previous work [22, 30, 31], we will present heuristic bounds for the noise growth behaviour of FV and BGV with respect to the canonical embedding norm $\|\cdot\|^{\text{can}}$. Throughout this work, the notation $\|a\|$ refers to the infinity norm of a, while $\|a\|^{\text{can}}$ 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 of a, so $\|a\|^{\text{can}} = \|\sigma(a)\|$.

We will use the following properties of the canonical embedding norm (see [39] for further discussion). For any polynomial a we have $\|a\| \leq c_m \|a\|^{\text{can}} \leq \|a\|_1$

where c_m is a constant known as the ring expansion factor (see [25]). We have $c_m = 1$ when m is a power of two [25]. In this case, it suffices for correctness to ensure that $\|v\|^{\text{can}}$ is less than the maximal value of $\|v\|$ such that decryption succeeds. For any polynomials a, b we have $\|ab\|^{\text{can}} \leq \|a\|^{\text{can}} \|b\|^{\text{can}}$.

Let a, b, c be such that their canonical embeddings have standard deviations σ_a, σ_b and σ_c respectively. Following Costache and Smart [22], we use the following estimates: $\|a\|^{\text{can}} \leq 6\,\sigma_a$ and $\|ab\|^{\text{can}} \leq 16\,\sigma_a\,\sigma_b$ and $\|abc\|^{\text{can}} \leq 40\,\sigma_a\,\sigma_b\,\sigma_c$.

The standard deviations in situations of interest for this paper are as follows. A polynomial f with coefficients distributed uniformly in $[-\frac{k}{2}, \frac{k}{2}]$ is such that the canonical embedding of f has standard deviation $\sigma_f = \frac{k\sqrt{n}}{\sqrt{12}}$. A polynomial e drawn from an error distribution χ, which has standard deviation σ, is such the canonical embedding of e has standard deviation $\sigma_e = \sigma\sqrt{n}$. A polynomial s drawn from the FV secret key distribution as implemented in SEAL [46] is such that the canonical embedding of s has standard deviation $\sigma_s = \sqrt{2n/3}$. A polynomial s drawn from the BGV secret key distribution as implemented in HElib [33] is such that the canonical embedding of s has standard deviation $\sigma_s = \sqrt{h}$, where h is the Hamming weight of s.

2.3 The BGV scheme

In this section we introduce the BGV scheme [12]. 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 $p = q = 1 \pmod t$ [12, 30]. 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 $s \xleftarrow{\$} S$ and output $\texttt{sk} = s$.
- $\texttt{PublicKeyGen}(\texttt{sk})$: Set $s = \texttt{sk}$ and sample $a \xleftarrow{\$} R_q$ 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 $a_i \xleftarrow{\$} R_q$ 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 $u \xleftarrow{\$} S$ 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 $m' = [[c_0 + c_1 s]_q]_t$.
- $\texttt{Add}(\texttt{ct}_0, \texttt{ct}_1)$: 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 $c_0 = [\texttt{ct}_0[0]\texttt{ct}_1[0]]_q$, $c_1 = [\texttt{ct}_0[0]\texttt{ct}_1[1] + \texttt{ct}_0[1]\texttt{ct}_1[0]]_q$, and $c_2 = [\texttt{ct}_0[1]\texttt{ct}_1[1]]_q$. 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 $\texttt{ct} = (c_0', c_1')$.

2.4 The FV scheme

In this section we introduce the FV scheme [28]. To simplify presentation we follow the 'textbook' FV scheme as presented by Fan and Vercauteren [28], in which ciphertexts are of size 2: that is, they are a tuple of 2 elements in R_q. This is denoted $\mathtt{ct} = (\mathtt{ct}[0], \mathtt{ct}[1])$. In particular, we will always assume that any output[6] 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 [46] 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 [22]. 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 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 $s \xleftarrow{\$} S$ and output $\mathtt{sk} = s$.
- $\mathtt{PublicKeyGen}(\mathtt{sk})$: Set $s = \mathtt{sk}$ and sample $a \xleftarrow{\$} R_q$ and $e \leftarrow \chi$. Output $\mathtt{pk} = ([-(as + e)]_q, a)$.
- $\mathtt{EvaluationKeyGen}(\mathtt{sk}, w)$: Set $s = \mathtt{sk}$. For $i \in \{0, \ldots, \ell\}$, sample $a_i \xleftarrow{\$} R_q$ and $e_i \leftarrow \chi$. Output $\mathtt{evk} = \left([-(a_i s + e_i) + w^i s^2]_q, a_i\right)$.
- $\mathtt{Encrypt}(\mathtt{pk}, m)$: For the message $m \in R_t$. Let $\mathtt{pk} = (p_0, p_1)$, sample $u \xleftarrow{\$} S$ 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 $\mathtt{ct}' = (c_0', c_1')$.

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

3 BGV noise growth in practice

3.1 Noise growth behaviour

In this section we reproduce the heuristic bounds on the noise growth behaviour of BGV presented in [22]. These use the *critical quantity* [22] definition of noise.

Definition 1 (BGV critical quantity [22]). *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 [22]). *A BGV ciphertext* $\mathtt{ct}$ *encrypting a message* m *can be correctly decrypted if the critical quantity* v *satisfies* $\|v\| < q/2$*.*

Lemma 2 (Encrypt [22]). *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*

$$\|v\|^{can} \leq t\left(\sqrt{3n} + 2\sigma\sqrt{n}(16\sqrt{h} + 3)\right) .$$

Lemma 3 (Add [22]). *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* $\mathtt{ct}_{add}$ *satisfies* $\|v_{add}\|^{can} \leq \|v_1\|^{can} + \|v_2\|^{can}$*.*

Lemma 4 (Mult [22]). *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* $\mathtt{ct}_{mult}$ *satisfies* $\|v_{mult}\|^{can} \leq \|v_1\|^{can} \cdot \|v_2\|^{can}$*.*

Lemma 5 (Relinearize [22]). *Let* $\mathtt{ct}$ *be a BGV ciphertext encrypting* m *and having noise* v*. Let* $\mathtt{ct}_{relin}$ *be the ciphertext obtained by the relinearization of* $\mathtt{ct}$*. Then with high probability, the critical quantity* v_{relin} *in* $\mathtt{ct}_{relin}$ *satisfies*

$$\|v_{relin}\|^{can} \leq \|v\|^{can} + \frac{8}{\sqrt{3}} t(\ell + 1)\sigma n w .$$

Lemma 6 (ModSwitch [22]). *Let* $\mathtt{ct}$ *be a BGV ciphertext encrypting* m *with critical quantity* v *with respect to a modulus* q*. Let* $\mathtt{ct}_{mod}$ *be the ciphertext encrypting* m *obtained by modulus switching to the modulus* p*. Then with high probability, the critical quantity* v_{mod} *in* $\mathtt{ct}_{mod}$ *satisfies*

$$\|v_{mod}\|^{can} \leq \frac{p}{q}\|v\|^{can} + t\sqrt{n}\left(\sqrt{3} + \frac{8\sqrt{h}}{\sqrt{3}}\right) .$$

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 and standard deviation of the observed noise. Separately, we compute an estimate of the noise growth using the heuristic bounds.

HElib offers a debugging function[7] 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 Section 2.3 above is not implemented in HElib. Instead, a different variant is implemented (see [31]). For this reason, we do not investigate the noise growth behaviour during a HElib relinearization.

Table 1 gives the results of this experiment for 10000 trials, using the following HElib default parameters: $\sigma = 3.2$, $w = 64$, $c = 2$, $k = 80$. We set the plaintext modulus as $t = 3$ by choosing the HElib parameters $p = 3$ and $r = 1$. We set $s = 1$ as we did not require batching functionality. We used dimension $n \in \{2048, 4096, 8192, 16384\}$ by choosing the HElib parameter $m \in \{4096, 8192, 16384, 32768\}$, and we verified that our other choices allowed for these m using the function `FindM`. 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 [46]. The parameters for $n = 2048$ were not large enough to perform modulus switching.

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 [16]. In order to facilitate an 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* $\mathtt{ct}$ *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 [16] as $-\log_2(2 \cdot \|v\|) = \log_2(q) - \log_2(q \cdot \|v\|) - 1$.

[7] `decryptAndPrint`

n	Enc			Add			Mult			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	16.8	5.12	1.68	17.8	5.63	2.16	34.6	14.7	11.3	-	-	-
4096	17.3	5.19	1.65	18.3	5.69	2.11	35.6	15.3	11.7	12.9	8.21	4.65
8192	17.8	5.25	1.55	18.8	5.75	2.03	36.6	15.8	12.1	13.4	8.28	4.57
16384	18.3	5.31	1.53	19.3	5.81	2.00	37.6	16.4	12.5	13.9	8.34	4.53

Table 1. Logarithm to base 2 of the observed mean $\overline{x}$ and of the standard deviation σ (to 3 significant figures) of the noise in HElib ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$, together with the logarithm to base 2 of the estimation E of the noise growth in this evaluation obtained using heuristic bounds.

n	Enc			Add			Mult			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	37.0	48.7	0.127	36.0	48.2	0.127	19.0	39.1	0.130	-	-	-
4096	93.0	106	0.120	92.0	105	0.118	75.0	95.5	0.116	42.0	46.5	0.119
8192	205	217	0.108	204	216	0.109	186	206	0.106	153	158	0.108
16384	427	440	0.102	426	440	0.102	408	429	0.097	376	381	0.098

Table 2. Observed mean $\overline{x}$ and of the standard deviation σ (to 3 significant figures) of the noise budget in HElib ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$, together with the estimation E of the noise budget growth in this evaluation obtained using heuristic bounds.

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 observed noise budgets follow narrow distributions, and 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. Nevertheless, the heuristic bounds are not tight. For example, for fresh ciphertexts, the heuristic bound predicts 11 to 13 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 20 to 21 bits. The gap narrows after modulus switching, to around 5 bits.

We can conclude 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 heuristic upper bounds for the noise growth in FV with respect to the canonical norm $\|\cdot\|^{\text{can}}$. We use the *invariant noise* definition for noise following Chen *et al.* [16], as opposed to the *critical quantity* used by Costache and Smart [22] or the *inherent noise* as used in the original presentation of Fan and Vercauteren [28].

Definition 3 (FV invariant noise [16]). *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 $\|v\| < \frac{1}{2}$.

Lemma 7 (Maximal noise [16]). *An FV ciphertext $\mathtt{ct}$ encrypting a message m can be correctly decrypted if the invariant noise v satisfies $\|v\| < 1/2$.*

We now present heuristic bounds on the noise growth in each homomorphic operation. In general, these bounds are as presented in [16, 45], so we omit the proofs for brevity. We additionally present in Lemma 12 a bound for modulus switching. We use $\|m\|^{\text{can}} \leq t\sqrt{3n}$, as in [22]. We assume that the secret distribution S is implemented as in SEAL [46].

Lemma 8 (Encrypt [16, 45]). *Let $\mathtt{ct}$ be a fresh FV encryption of a message $m \in R_t$. With high probability, the invariant noise v in $\mathtt{ct}$ satisfies*

$$\|v\|^{can} \leq \frac{r_t(q)}{q} \cdot t\sqrt{3n} + \frac{t}{q} \cdot 2\sigma \left(\frac{16\sqrt{2}}{\sqrt{3}} n + 3\sqrt{n} \right) .$$

Lemma 9 (Add [16, 45]). *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 $\mathtt{ct}_{add}$ satisfies $\|v_{add}\|^{can} \leq \|v_1\|^{can} + \|v_2\|^{can}$.*

Lemma 10 (Multiply [16, 45]). *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 $\mathtt{ct}_{mult}$ satisfies*

$$\|v_{mult}\|^{can} \leq 3\,\|v_1\|^{can}\,\|v_2\|^{can} + \frac{2t\sqrt{n}}{q\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right)$$

$$+\left(2t\sqrt{3n}+\frac{2t\sqrt{n}}{\sqrt{12}}\left(3+\frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n}+\frac{40}{3}n\right)\right)\|v_1\|^{can}$$
$$+\left(2t\sqrt{3n}+\frac{2t\sqrt{n}}{\sqrt{12}}\left(3+\frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n}+\frac{40}{3}n\right)\right)\|v_2\|^{can}.$$

Lemma 11 (Relinearize [16, 45]). *Let $\mathtt{ct}$ be an FV ciphertext encrypting m and having invariant noise v. Let $\mathtt{ct}_{relin}$ be the ciphertext obtained by the relinearization of $\mathtt{ct}$. Then with high probability, the invariant noise v_{relin} in $\mathtt{ct}_{relin}$ satisfies*

$$\|v_{relin}\|^{can} \leq \|v\|^{can} + \frac{t}{q}\ (\ell+1)\frac{8}{\sqrt{3}}\,\sigma\, n\, w\,.$$

Lemma 12 (ModSwitch). *Let $\mathtt{ct}$ be an FV ciphertext encrypting m with invariant noise v with respect to a modulus q. Let $\mathtt{ct}_{mod}$ be the ciphertext encrypting m obtained by modulus switching to the modulus p. Then with high probability, the invariant noise v_{mod} in $\mathtt{ct}_{mod}$ satisfies*

$$\|v_{mod}\|^{can} \leq \|v\|^{can} + \frac{t}{p}\left(\sqrt{3n}+\frac{8\sqrt{2}}{3}n\right).$$

Proof. 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 $c_1' = \left[\left\lfloor \frac{p}{q}c_1 \right\rceil\right]_p$. By definition of invariant noise (with respect to q) in $\mathtt{ct}$,

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

where ϵ_0, ϵ_1 are terms introduced from the rounding. Hence by definition of invariant noise (with respect to p), $v_{\mathrm{mod}} = v + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right)$. We can bound this as follows:

$$\begin{aligned}
\|v_{\mathrm{mod}}\|^{\mathrm{can}} &= \left\|v + \frac{t}{p}\left(\epsilon_0 + \epsilon_1 s\right)\right\|^{\mathrm{can}}\\
&\leq \|v\|^{\mathrm{can}} + \frac{t}{p}\left\|\epsilon_0\right\|^{\mathrm{can}} + \frac{t}{p}\left\|\epsilon_1 s\right\|^{\mathrm{can}}\\
&\leq \|v\|^{\mathrm{can}} + \frac{t}{p}\left(6\cdot\frac{\sqrt{n}}{\sqrt{12}} + 16\cdot\frac{\sqrt{n}}{\sqrt{12}}\cdot\sqrt{\frac{2n}{3}}\right),
\end{aligned}$$

which simplifies to the stated bound. □

n	Enc			Add			Mult			Relin			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	27.0	35.4	0.484	26.0	35.0	0.046	0.000	17.0	0.222	0.000	16.9	0.232	-	-	-
4096	81.0	90.0	0.075	80.0	89.1	0.260	51.0	69.7	0.500	51.0	69.7	0.500	31.0	39.0	0.175
8192	189	198	0.118	188	198	0.088	157	178	0.309	157	178	0.321	139	148	0.441
16384	408	418	0.120	407	417	0.014	375	396	0.300	375	396	0.301	358	367	0.028

Table 3. Binary encoding setting. Observed mean $\overline{x}$ and standard deviation σ (to 3 significant figures) of the invariant noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for SEAL default parameters with dimension $n \in \{2048, 4096, 8192, 16384\}$, together with the estimation E of the noise budget growth in this evaluation obtained using heuristic bounds.

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 and standard deviation 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* [16], which is defined as $-\log_2(2 \cdot \|v\|)$.

The evaluation is as follows in the i-th trial. 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 $\mathtt{ct}_2$. Next, generate $\mathtt{ct}_5$ by relinearizing $\mathtt{ct}_4$. Finally, generate $\mathtt{ct}_6$ by modulus switching $\mathtt{ct}_5$ 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 $n \in \{2048, 4096, 8192, 16384\}$. We use decomposition bit count $\log w = 16$ and plaintext modulus $t = 256$. 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. 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 results of this experiment for 10000 trials.

n	Enc			Add			Mult			Relin			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	19.0	27.1	0.281	18.0	27.0	0.143	0.000	0.17	0.374	0.000	0.17	0.374	-	-	-
4096	71.0	79.0	0.000	70.0	78.0	0.000	32.0	50.0	0.045	32.0	50.0	0.045	23.0	31.0	0.000
8192	179	188	0.067	178	187	0.000	139	157	0.391	139	157	0.391	131	140	0.335
16384	398	407	0.000	397	406	0.000	356	376	0.026	356	376	0.026	350	359	0.000

Table 4. Batching setting. Observed mean $\overline{x}$ and standard deviation σ (to 3 significant figures) of the invariant noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for SEAL default parameters with dimension $n \in \{2048, 4096, 8192, 16384\}$, together with the estimation E of the noise budget growth in this evaluation obtained using heuristic bounds.

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. The small standard deviations in Tables 3 and 4 show that the observed noises follow very narrow distributions. This suggests that it is hard to generate a SEAL ciphertext with noise much larger than that of any other ciphertext formed through the same chain of homomorphic operations, in the same parameter setting. In particular, 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 8 to 10 (respectively 8 or 9) 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 16 to 21 (respectively 18 to 20) 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 gaps are all similar to the HElib case.

Consider for example the results for $n = 2048$, after the multiplication operation. In Table 3 the bounds predict that there is no noise budget remaining, and such a ciphertext should be considered completely corrupted with no hope of correct decryption or use in further operations. This could cause the user to choose larger parameters than may be necessary, given that the average observed ciphertext has 17 bits of noise budget remaining. This would result in worse performance. However in Table 4 we see that with very similar parameters (only t has been changed), in the batching setting, the bound accurately predicts that there is almost no noise budget left.

5 Towards an improved understanding of noise growth

5.1 Invariant noise: a new notion of noise for BGV

In Section 3 we looked at the noise growth behaviour of BGV in terms of the critical quantity (Definition 1). We saw that the values for the critical quantity observed in ciphertexts in HElib were only loosely bounded by the estimates given by heuristic upper bounds on the critical quantity. One possible explanation for this suboptimal modelling of the noise growth behaviour is that we are using an inaccurate notion of noise. In this section, we introduce a new notion of noise for BGV, which we call the *invariant noise*. We will then present heuristic bounds on its growth, and investigate its appropriateness for estimating BGV noise growth in practice.

Definition 4. *Let* $\mathtt{ct}$ *be a BGV ciphertext encrypting the message* $m \in R_t$*. Its* invariant noise v *is the polynomial such that*

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

The motivation for the definition of the BGV invariant noise can be seen as analogous to one motivating idea for the definition of invariant noise for FV [16]. This idea is that the noise should be the part of the ciphertext that can cause decryption failure; indeed Definition 4 can be seen as a partial decryption. In a successful BGV decryption, the invariant noise v will be removed when $[\mathtt{ct}(s)]_q$ is taken modulo t, leaving the message m modulo t. This should be contrasted with the critical quantity (Definition 1), which is the whole of $[\mathtt{ct}(s)]_q$, including the underlying message.

Lemma 13 (Maximal noise). *A BGV ciphertext* $\mathtt{ct}$ *encrypting a message* $m \in R_t$ *can be correctly decrypted if the invariant noise* v *satisfies* $\|v\| < q/2 - t$*.*

Proof. During decryption, we compute $[[\mathtt{ct}(s)]_q]_t$. If there is no wraparound modulo q when computing $[\mathtt{ct}(s)]_q$ then for some integer polynomial k, we have that $c_0 + c_1 s = m + tk$ and the reduction modulo t returns m. Hence for correctness, we must have $\|m + v\| = \|c_0 + c_1 s\| \leq q/2$. Since $\|m\| \leq t$ and $\|m + v\| \leq \|m\| + \|v\|$ it is sufficient to require $\|v\| \leq q/2 - t$. □

We can also define the *invariant noise budget* for the BGV invariant noise, analogous to Definition 2 for the critical quantity.

Definition 5. *Let* $\mathtt{ct}$ *be a BGV ciphertext encrypting a message* $m \in R_t$ *with respect to modulus* q *having invariant noise* v*. The* invariant noise budget *for this ciphertext is defined as*

$$\log_2 (q/2 - t) - \log_2 (\|v\|) .$$

We now present heuristic bounds on the invariant noise growth.

Lemma 14 (Encrypt). *Let* $\mathtt{ct}$ *be a fresh BGV encryption of a message* $m \in R_t$*. The invariant noise* v *in* $\mathtt{ct}$ *is given by* $v = t(-eu + e_1 + e_2 s)$ *and can be bounded as* $\|v\|^{can} \leq 2t\sigma\sqrt{n}(16\sqrt{h}+3)$*.*

Proof. In a fresh ciphertext, we have $\mathtt{ct} = (c_0, c_1)$ such that

$$\begin{aligned}[c_0 + c_1 s]_q &= m - asu - teu + te_1 + aus + tse_2 \\ &= m + t(-eu + e_1 + se_2)\,.\end{aligned}$$

Hence the invariant noise is $v = t(-eu+e_1+se_2)$. The heuristic bound on $\|v\|^{\text{can}}$ follows from Lemma 2. □

Lemma 15 (Add). *Let* $\mathtt{ct}_1$ *and* $\mathtt{ct}_2$ *be two BGV ciphertexts encrypting the messages* $m_1, m_2 \in R_t$*, and having invariant noises* v_1*,* v_2*, respectively. Then the invariant noise* v_{add} *in their sum* $\mathtt{ct}_{add}$ *is given by* $v_{add} = v_1 + v_2$ *and can be bounded as* $\|v_{add}\|^{can} \leq \|v_1\|^{can} + \|v_2\|^{can}$*.*

Proof. The argument is the same as for the proof of Lemma 3. □

Lemma 16 (Mult). *Let* $\mathtt{ct}_1$ *and* $\mathtt{ct}_2$ *be two BGV ciphertexts encrypting* $m_1, m_2 \in R_t$*, and having invariant noises* v_1*,* v_2*, respectively. Then the invariant noise* v_{mult} *in their product* $\mathtt{ct}_{mult}$ *is given by* $v_{mult} = m_1 v_2 + m_2 v_1 + v_1 v_2$ *and can be bounded as* $\|v_{mult}\|^{can} \leq t\sqrt{3n}\,(\|v_1\|^{can} + \|v_2\|^{can}) + \|v_1\|^{can} \cdot \|v_2\|^{can}$*.*

Proof. Let $\mathtt{ct}_1 = (c_0, c_1)$ and $\mathtt{ct}_2 = (c_0', c_1')$. By the definitions of $\mathtt{ct}_{\text{mult}}$ and of the invariant noise in $\mathtt{ct}_1$ and $\mathtt{ct}_2$, we have

$$\begin{aligned}\mathtt{ct}_{\text{mult}}(s) &= c_0 c_0' + (c_0 c_1' + c_1 c_0')s + c_1 c_1' s^2 \pmod q \\ &= \mathtt{ct}_1(s)\mathtt{ct}_2(s) \pmod q \\ &= (m_1 + v_1)(m_2 + v_2) \\ &= m_1 m_2 + m_1 v_2 + m_2 v_1 + v_1 v_2\,.\end{aligned}$$

Hence the invariant noise is given by $v_{\text{mult}} = m_1 v_2 + m_2 v_1 + v_1 v_2$. This can be bounded as follows, again using $\|m\|^{\text{can}} \leq t\sqrt{3n}$ as in [22]:

$$\begin{aligned}\|v_{\text{mult}}\|^{\text{can}} &= \|m_1 v_2 + m_2 v_1 + v_1 v_2\|^{\text{can}} \\ &\leq \|m_1 v_2\|^{\text{can}} + \|m_2 v_1\|^{\text{can}} + \|v_1 v_2\|^{\text{can}} \\ &\leq \|m_1\|^{\text{can}} \cdot \|v_2\|^{\text{can}} + \|m_2\|^{\text{can}} \cdot \|v_1\|^{\text{can}} + \|v_1\|^{\text{can}} \cdot \|v_2\|^{\text{can}} \\ &\leq t\sqrt{3n}\,(\|v_1\|^{\text{can}} + \|v_2\|^{\text{can}}) + \|v_1\|^{\text{can}} \cdot \|v_2\|^{\text{can}}\,.\end{aligned}$$

□

Lemma 17 (Relinearize). *Let* $\mathtt{ct} = (c_0, c_1, c_2)$ *be a BGV ciphertext encrypting* m *with invariant noise* v*. Let* $\mathtt{ct}_{relin}$ *be the ciphertext encrypting* m*, obtained by the relinearization of* $\mathtt{ct}$*. Then, the invariant noise* v_{relin} *in* $\mathtt{ct}_{relin}$ *is given by* $v_{relin} = v - t\sum_{i=0}^{\ell} e_i c_2^{(i)}$ *and can be bounded as*

$$\|v_{relin}\|^{can} \leq \|v\|^{can} + \frac{8}{\sqrt{3}} t(\ell+1)\sigma n w\,.$$

n	Enc			Add			Mult			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	37.0	48.7	0.129	36.0	48.2	0.129	19.0	39.1	0.133	-	-	-
4096	93.0	106	0.118	92.0	105	0.117	75.0	95.5	0.117	42.0	46.5	0.118
8192	205	217	0.112	204	216	0.109	186	206	0.107	153	158	0.107
16384	427	440	0.104	426	440	0.101	408	429	0.100	376	381	0.099

Table 5. Observed mean $\overline{x}$ and of the standard deviation σ (to 3 significant figures) of the invariant noise budget in HElib ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$, together with the estimation E of the invariant noise budget growth in this evaluation obtained using heuristic bounds.

Proof. The argument is the same as for the proof of Lemma 5. □

Lemma 18 (ModSwitch). *Let* $\mathtt{ct}$ *be a BGV ciphertext encrypting* m *with invariant noise* v *with respect to a modulus* q*. Let* $\mathtt{ct}_{mod}$ *be the ciphertext encrypting* m *obtained by modulus switching to the modulus* p*. Then, the invariant noise* v_{mod} *in* $\mathtt{ct}_{mod}$ *is given by* $v_{mod} = -\frac{q-p}{q}m + \frac{p}{q}v + \frac{p}{q}(\delta_0 + \delta_1 s)$ *and can be bounded as* $\|v_{mod}\|^{can} \leq \frac{p}{q}\|v\|^{can} + t\sqrt{n}\left(\frac{\sqrt{3}(q-p)}{q} + \sqrt{3} + \frac{8\sqrt{h}}{\sqrt{3}}\right)$.

We defer the proof of Lemma 18 to Appendix A.

We implemented a function in HElib to measure the invariant noise budget, and a function to estimate the invariant noise budget using the heuristic bounds. We then ran the same experiment as in Section 3.2 to compare the growth of the observed invariant noise budget in HElib ciphertexts with that predicted from the heuristic bounds. Table 5 gives the results of this experiment for 10000 trials.

Comparing Table 5 and Table 2, we see that the invariant and critical quantity noise budgets observed in HElib ciphertexts are very similar, for all operations, for all n. Moreover, the estimates derived from the heuristic bounds are exactly the same for all operations, for all n. We can conclude that using the invariant noise rather than the critical quantity does not make a significant difference.

Indeed, it is reasonable to expect that the critical quantity (that is, v such that $[\mathtt{ct}(s)]_q = v$) and invariant noise (that is, v such that $[\mathtt{ct}(s)]_q = m + v$) would have similar behaviour, because the only difference is the contribution of the message term m. If this was a significant term in the critical quantity, then we could potentially distinguish an encryption of zero, which would violate IND-CPA security.

5.2 A new notion of noise for FV

The behaviour we observed for FV in Section 4 was similar to that which we observed for BGV in Section 3: the invariant noises in SEAL ciphertexts were only loosely bounded by the estimates by from heuristic upper bounds. We can therefore equally argue that is possible that the invariant noise is the wrong notion of noise for FV.

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. One advantage of this is that many cross terms disappear in the analysis, compared to the original definition of *inherent noise* [28]. Indeed, this simplification was one motivation in [16] for the introduction of the invariant noise definition. However, notice that this $r_t(q) \cdot m$ term is only introduced by the decryption process: in other words, we can regard this as the decryption process itself introducing noise. This term is not a part of the noise that the ciphertext carries before the decryption, and should not be counted in the intermediate ciphertexts. Including this term in every ciphertext, including all the intermediate ones, will lead to overestimates that compound. Motivated by this observation, that the invariant noise leads to simpler, but looser, bounds, we introduce a new notion of noise, to try to more closely model the noise growth behaviour. We call this the *scaled inherent noise* as it is equal to a scaling by t/q of the inherent noise.

Definition 6 (Scaled inherent noise). *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,*

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

where $q = \Delta t + r_t(q)$.

Lemma 19 (Maximal noise). *An FV ciphertext $\mathtt{ct}$ encrypting a message m can be correctly decrypted if the scaled inherent noise v satisfies*

$$\left\| \frac{-r_t(q)}{q} m + v \right\| < \frac{1}{2}.$$

Proof. Decryption is correct if and only if $m = \left[\left\lfloor \frac{t}{q} \Delta m + v + at \right\rceil \right]_t$. Consider

$$\begin{aligned} m' &= \left[\left\lfloor \frac{\Delta t}{q} m + v + at \right\rceil \right]_t \\ &= \left[\left\lfloor \frac{q - r_t(q)}{q} m + v + at \right\rceil \right]_t \\ &= \left[m + \left\lfloor \frac{-r_t(q)}{q} m + v \right\rceil + at \right]_t \\ &= m \mod t, \end{aligned}$$

if $\left\lfloor \frac{-r_t(q)}{q} m + v \right\rceil = 0$. Hence we require that $\left\| \frac{-r_t(q)}{q} m + v \right\| < \frac{1}{2}$. □

Note that $\left\| \frac{-r_t(q)}{q} m + v \right\| \leq \frac{r_t(q)}{q} \|m\| + \|v\| \leq \frac{t^2}{q} + \|v\|$. Therefore for correctness it is sufficient to require that $\|v\| \leq \frac{1}{2} - \frac{t^2}{q}$. This motivates the following definition of noise budget.

Definition 7. *Let* $\mathtt{ct}$ *be an FV ciphertext encrypting a message* $m \in R_t$ *with respect to modulus* q *having scaled inherent noise* v*. The* scaled inherent noise budget *for this ciphertext is defined as* $\log_2\left(\frac{q}{2} - t^2\right) - \log_2\left(q \cdot \|v\|\right)$.

We now present the noise growth in each homomorphic operation. We again follow [22] and bound a message polynomial as $\|m\|^{\text{can}} \leq t\sqrt{3n}$.

Lemma 20 (Encrypt). *Let* $\mathtt{ct}$ *be a fresh FV encryption of a message* $m \in R_t$*. The scaled inherent noise* v *in* $\mathtt{ct}$ *is given by* $v = \frac{t}{q}\left(-eu + e_1 + e_2 s\right)$*, and can be bounded as* $\|v\|^{can} \leq \frac{t}{q} \cdot 2\sigma\left(\frac{16\sqrt{2}}{\sqrt{3}} n + 3\sqrt{n}\right)$.

Proof. By definition a fresh ciphertext $\mathtt{ct} = (c_0, c_1)$ encrypting $m \in R_t$ under public key $\mathtt{pk} = (p_0, p_1) = ([-(as + e)]_q, a)$ satisfies, for some integer polynomials k_0, k_1, k_2,

$$\begin{aligned}
\frac{t}{q}\mathtt{ct}(s) &= \frac{t}{q}\left(\Delta m + p_0 u + e_1 + k_0 q + p_1 us + e_2 s + k_1 q s\right) \\
&= \frac{t}{q}\left(\Delta m\right) + \frac{t}{q}\left((-as - e + k_2 q)u + e_1 + aus + e_2 s\right) + t(k_0 + k_1 s) \\
&= \frac{t}{q}\left(\Delta m\right) + \frac{t}{q}\left(-asu - eu + e_1 + aus + e_2 s\right) + t(k_0 + k_1 s + k_2 u)\,,
\end{aligned}$$

hence the scaled inherent noise in this ciphertext is $v = \frac{t}{q}\left(-eu + e_1 + e_2 s\right)$. The bound follows from the same argument as in the proof of Lemma 8 (see [45]). □

Lemma 21 (Add). *Let* $\mathtt{ct}_1$ *and* $\mathtt{ct}_2$ *be two FV ciphertexts encrypting* $m_1, m_2 \in R_t$*, and having scaled inherent noises* v_1*,* v_2*, respectively. Let* $[m_1 + m_2]_t = m_1 + m_2 + a_0 t$ *for some integer polynomial* a_0*. Then the scaled inherent noise* v_{add} *in their sum* $\mathtt{ct}_{add}$ *is given by*

$$v_{add} = v_1 + v_2 + \frac{t \cdot r_t(q)}{q} a_0\,,$$

and can be bounded as $\|v_{add}\|^{can} \leq \|v_1\|^{can} + \|v_2\|^{can} + \frac{3\sqrt{3n} \cdot t \cdot r_t(q)}{q}$.

We defer the proof of Lemma 21 to Appendix B.

Lemma 22 (Mult). *Let* $\mathtt{ct}_1$ *be an FV ciphertext of size* 2 *encrypting* m_1 *with scaled inherent noise* v_1*, and let* $\mathtt{ct}_2$ *be an FV ciphertext of size* 2 *encrypting* m_2 *with scaled inherent noise* v_2 *so that for some integer polynomials* a_1*,* a_2*,*

$$\frac{t}{q}\mathtt{ct}_1(s) = \frac{t}{q}\Delta m_1 + v_1 + a_1 t$$

$$\frac{t}{q}\mathtt{ct}_2(s) = \frac{t}{q}\Delta m_2 + v_2 + a_2 t\,.$$

Then the scaled inherent noise v_{mult} in their product $\mathtt{ct}_{mult}$ is given by

$$\begin{aligned} v_{mult} = \frac{r_t(q)}{q} t a_0 - \frac{r_t(q)\Delta t}{q^2}[m_1 m_2]_t + \frac{t^2 r_t(q)\Delta}{q^2} a_0 + \frac{t\Delta}{q}(m_2 v_1 + m_1 v_2) \\ - \frac{r_t(q)}{q}(m_2 a_1 t + m_1 a_2 t) + v_1 v_2 + v_2 a_1 t + v_1 a_2 t + \frac{t}{q}\left(\sum_{i=0}^{2} \epsilon_i s^i\right), \end{aligned}$$

where a_0 is an integer polynomial such that $[m_1 m_2]_t = m_1 m_2 + a_0 t$. The noise can be bounded as

$$v_{mult} \le A \cdot \|v_1\|^{can} \cdot \|v_2\|^{can} + B\left(\|v_1\|^{can} + \|v_2\|^{can}\right) + C\,,$$

where $A = 3$, $B = \frac{t\sqrt{3n}}{q}\left(2\Delta t + r_t(q)\right) + \frac{2t\sqrt{n}}{\sqrt{12}}\left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n\right)$, and

$$\begin{aligned} C = \frac{r_t(q) t\sqrt{n}}{q}\left(\sqrt{3} + \frac{4t\sqrt{n}}{3} + 12t\sqrt{n} + \frac{32t\sqrt{2}}{\sqrt{3}}n + \frac{160}{3}tn\sqrt{n}\right) \\ + \frac{r_t(q)\Delta t^2\sqrt{n}}{q^2}\left(2\sqrt{3} + \frac{4t\sqrt{n}}{3} + 6t\sqrt{n}\right) + \frac{2t\sqrt{n}}{q\sqrt{12}}\left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n\right). \end{aligned}$$

We defer the proof of Lemma 22 to Appendix C.

Lemma 23 (Relinearize). *Let $\mathtt{ct} = (c_0, c_1, c_2)$ be an FV ciphertext encrypting m with scaled inherent noise v. Let $\mathtt{ct}_{relin}$ be the ciphertext encrypting m, obtained by the relinearization of $\mathtt{ct}$. Then, the scaled inherent noise v_{relin} in $\mathtt{ct}_{relin}$ is given by*

$$v_{relin} = v - \frac{t}{q}\sum_{i=0}^{\ell} e_i c_2^{(i)}\,,$$

and can be bounded as $\|v_{relin}\|^{can} \le \|v\|^{can} + \frac{t}{q}\,(\ell+1)\frac{8}{\sqrt{3}}\,\sigma\, n\, w$.

Proof. The proof follows the argument as for Lemma 11. □

Lemma 24 (ModSwitch). *Let $\mathtt{ct}$ be an FV ciphertext encrypting m with scaled inherent noise v with respect to a modulus q. Let $\mathtt{ct}_{mod}$ be the ciphertext encrypting m obtained by modulus switching to the modulus p. Then, the scaled inherent noise v_{mod} in $\mathtt{ct}_{mod}$ is given by*

$$v_{mod} = v + \left(\frac{r_t(p)}{p} - \frac{r_t(q)}{q}\right) m + \frac{t}{p}(\epsilon_0 + \epsilon_1 s)$$

and can be bounded as

$$\|v_{mod}\|^{can} \le \|v\|^{can} + t\sqrt{3n}\left(\frac{r_t(p)}{p} - \frac{r_t(q)}{q}\right) + \frac{t}{p}\left(\sqrt{3n} + \frac{8\sqrt{2}}{3}n\right).$$

Proof. 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 $c_1' = \left[\left\lfloor \frac{p}{q} c_1 \right\rceil\right]_p$. By definition of $\mathtt{ct}_{\mathrm{mod}}$, and by definition of the scaled inherent noise in $\mathtt{ct}$, for some integer polynomials k_0, k_1 and for some polynomials $\|\epsilon_i\| \leq \frac{1}{2}$,

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

since

$$\begin{aligned}
\frac{t}{q}\Delta_q m &= \frac{t}{p}\Delta_p m + \frac{\Delta_q t}{q} m - \frac{\Delta_p t}{p} m \\
&= \frac{t}{p}\Delta_p m + \left(\frac{q - r_t(q)}{q} - \frac{p - r_t(p)}{p}\right) m \\
&= \frac{t}{p}\Delta_p m + \left(1 - \frac{r_t(q)}{q} - 1 + \frac{r_t(p)}{p}\right) m\,.
\end{aligned}$$

Hence the scaled inherent noise is $v_{\mathrm{mod}} = v + \left(\frac{r_t(p)}{p} - \frac{r_t(q)}{q}\right) m + \frac{t}{p}(\epsilon_0 + \epsilon_1 s)$ and the bound follows using Lemma 12. □

We implemented a function in SEAL to measure the scaled inherent noise budget, and a function to estimate the noise budget using the heuristic bounds. We then ran the same experiment as in Section 4.2 to compare the growth of the observed noise budget in SEAL ciphertexts with that predicted from the heuristic bounds. Tables 6 and 7 presents the results of this experiment for 10000 trials in the binary encoder and batch encoder setting respectively.

Comparing Tables 6 and 7 with Tables 3 and 4 we see that the observed invariant noise budget and scaled inherent noise budget in SEAL ciphertexts is extremely similar, especially in the binary encoder setting. Moreover, there remains a significant gap between the observed noises and heuristic estimates. In fresh ciphertexts, the gap in Tables 6 and 7 is between 7 and 9 bits, while it is between 8 and 10 bits in Tables 3 and 4. After multiplication, the gap in Tables 6 and 7 is typically between 17 and 21 bits, the same as in Tables 3 and 4. Similarly there remains a gap of around 8 or 9 bits after modulus switching. We conclude that while the scaled inherent noise represents a slight improvement for modelling the noise in fresh ciphertexts, it suffers from the same issues as the invariant noise in terms of suitability for use in selecting parameters for correctness.

n	Enc			Add			Mult			Relin			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	27.6	35.3	0.472	26.6	35.0	0.048	0.000	17.0	0.207	0.000	17.0	0.207	-	-	-
4096	81.6	90.0	0.054	80.6	89.1	0.350	51.1	69.9	0.392	51.1	69.9	0.394	31.3	39.0	0.017
8192	190	198	0.100	189	198	0.094	158	177.9	0.241	158	177.3	0.240	139	147.9	0.351
16384	409	418	0.139	408	417	0.010	375	396.2	0.388	375	396.2	0.388	358	367	0.000

Table 6. Observed mean $\overline{x}$ and of the standard deviation σ (to 3 significant figures) of the scaled inherent noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$ in the binary encoder setting, together with the estimation E of the noise budget growth in this evaluation obtained using heuristic bounds.

n	Enc			Add			Mult			Relin			ModSwitch		
	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ	E	$\overline{x}$	σ
2048	19.6	27.4	0.482	18.4	27.0	0.177	0.000	0.700	0.459	0.000	0.700	0.459	-	-	-
4096	73.6	82.0	0.062	69.6	78.0	0.000	32.0	50.0	0.111	32.0	50.0	0.111	22.9	31.0	0.125
8192	182	190	0.108	178	187	0.000	139	157.1	0.309	139	157.1	0.309	126	134	0.000
16384	401	410	0.139	397	406	0.000	356	376	0.114	356	376	0.114	344	352	0.000

Table 7. Observed mean $\overline{x}$ and of the standard deviation σ (to 3 significant figures) of the scaled inherent noise budget in SEAL ciphertexts over 10000 trials of a specific homomorphic evaluation for parameter sets with dimension $n \in \{2048, 4096, 8192, 16384\}$ in the batch setting, together with the estimation E of the scaled inherent noise budget growth in this evaluation obtained using heuristic bounds.

6 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 [22]. We make several improvements to the previous work [22]. Most importantly, we select parameters that achieve a security level $\lambda = 128$ according to the Homomorphic Encryption Standard [1]. In contrast, the previous work [22] relied on a security analysis by Lindner and Peikert [37], which has been shown to be incorrect [4, 2]. In fact, as shown in [23], FHE parameters which were estimated by [37] to have 80 bits of security had as little as 51 bits of security according to [2, 4].

Where possible, we make choices in line with the implementations of BGV and FV in HElib and SEAL respectively. To this end, we use the invariant noise for FV, which is a scaling of the *critical quantity* used in [22]. For BGV, we use the critical quantity as in [22]. As is the case throughout this paper, we also use the distributions for the secret key as implemented in HElib and SEAL. In particular, this means BGV is modelled as having a sparse secret while FV is not. We discuss these issues in more detail in Section 7.1.

Our analysis allows for a more flexible modulus switching for FV compared to that in the previous work [22]. We discuss this in Section 6.3.

6.1 Methodology and parameter selection

Our comparison uses the same homomorphic evaluation function as in [22]. 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 [24] 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 [22]. Of course, we could have considered other criteria such as key size. However, 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 [22]: 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 $L \in \{2, 4, 6, \ldots, 30\}$. For BGV, we assume that the secret key has $h = 64$ nonzero coefficients. We always use plaintext modulus $t = 3$, which was shown to be optimal among integral bases for encoding by Costache *et al.* [24]. 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 distribution on R_3. The possible pairs of n and Q are reproduced in Table 8.

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

Table 8. Pairs of the parameters n and Q (given as its bitsize $\log Q$) used in our comparison, extracted from the Homomorphic Encryption Standard [1, Table 1].

6.2 Results, analysis and limitations

Table 9 presents the results of the comparison. 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 $L \in \{8, 16, 18\}$ we see that FV required a larger parameter set than BGV. Similarly, for $L \in \{28, 30\}$, the largest parameter set with $n = 32768$ was not enough to support the computation in FV, while it was for BGV. This would suggest that BGV is sometimes preferable to FV.

The use of a sparse secret in the BGV case may explain its apparent better performance: we expect that a sparser secret will correspond to smaller noise growth. However, such a secret distribution admits additional attacks [2] and no parameter sets using sparse secrets are currently standardised [1].

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	10.8	12.8	12.8	12.8	12.8	12.8	12.8
FV	4.75	6.77	8.77	10.8	10.8	10.8	10.8	12.8	12.8	12.8	12.8	12.8	12.8	-	-

Table 9. 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. The symbol '-' denotes that no suitable parameters could be found.

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 do not attempt to make any scheme-specific optimisations that may be possible. Also, we note that while our choice of plaintext modulus $t = 3$ is optimal for integral bases, recent work has demonstrated the benefits of using non-integral bases [7, 14, 19] or using t a polynomial rather than an integer [8, 17].

6.3 Modulus switching

We conclude this section by commenting on the analysis of modulus switching in [22]. 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 $p_i = 1 \pmod{t}$, to ensure correctness in the modulus switching operation. Choosing such p_i is not necessary for FV, yet was listed as a requirement in [22]. The experimental results in [22] were only concerned with the bit size of the primes in the chain of moduli, and so 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. Furthermore, in practice, this could lead to BGV requiring larger parameters than FV.

7 Discussion

7.1 Fairness

The challenge of ensuring a fair comparison between homomorphic encryption schemes has been noted in previous work [15]. In Section 6, we are concerned primarily with comparing FV as implemented in SEAL and BGV as implemented in HElib. The libraries have different goals and so, for example, provide the user with different default parameters, and the ability to change different subsets of the parameters via the public API. In our comparison we have tried to balance investigating a similar parameter setting with keeping the spirit and goals of the libraries.

Throughout the paper we assumed that the secret distributions in FV and BGV were as they are implemented in SEAL and HElib respectively. An argument for this assumption is that in Sections 3, 4 and 5 we were interested in comparing the noise estimated from the heuristics to the noises seen in ciphertexts in actual implementations, and so it makes sense to consider the secret distributions as implemented. An argument against this assumption is that the secret distributions are different, which could have implications for both security and performance. Indeed, in Section 6 we saw that BGV performed slightly better than FV, which could be partially explained by the use of a sparse secret.

Another difficulty has been with choosing the plaintext space. The most natural choice for HElib plaintext space is $p = 2$, $r = 1$ but the choice $p = 2$ is incompatible with the choice of a power-of-2 cyclotomic. We chose $p = 3$, which is the smallest possible choice that is compatible, as well as being optimal among integral bases for encoding [24]. This is our attempt as a best compromise between keeping the experiments on HElib and SEAL consistent while being true to the intended usage of the libraries. We recognise these may not be the best choices, but we argue that such compromises are unavoidable.

7.2 Improving the heuristics

In Section 3 (respectively Section 4) we saw that there was a significant gap between the noise observed in HElib (respectively SEAL) ciphertexts and the noise predicted using heuristic estimates. In Sections 5.1 and 5.2 we tried to improve the situation by introducing new notions of noise for both BGV and FV. However, further experiments showed that in both cases we did not obtain much of an improvement. The next natural direction, then, is to try to improve the methodology used to obtain the heuristic bounds themselves.

The heuristic bounds include terms bounding 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 [22], we use $B = 6$, while HElib uses $B = 10$ as a default [33]. 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.

One could consider a δ-subgaussian approach [41, 43] to obtain noise bounds, as was done in [40] for a BGV-like scheme. In particular, [40, Lemma 8.7] gives the noise growth for homomorphic multiplication for this scheme. However, this is not an approach typically studied and implemented in homomorphic encryption in practice. Furthermore, it is unclear how to directly translate the asymptotic $\omega(\sqrt{\log n})$ term of [40, Lemma 8.7] into a concrete bound for multiplication.

Another related work [42] showed that by using a Central Limit Theorem (CLT) argument, tighter correctness bounds for the scheme in [40] can potentially be obtained. Whether one arrives at the noise growth analysis via δ-subgaussians or CLT, there is such high dimensionality in homomorphic encryption that we can expect noise terms to have Gaussian tail bounds. Therefore we will encounter the same issue of requiring loose bounds in order to ensure correctness.

7.3 Conclusion and future directions

In this work, we have been unable to significantly lessen the heuristic-to-practical gap between the noise observed in practice and the estimated noise growth from heuristic upper bounds. Furthermore, we cannot improve the heuristic bounds theoretically, because we need a proof that the scheme has a reasonable failure probability. A clear conclusion of our work in is therefore that any reasonable method of predicting noise growth behaviour in the FV and BGV schemes will only loosely upper bound the growth seen in practice in implementations of these schemes. That is, these bounds are the best possible in terms of a theoretical analysis, and we must take a different approach in order to obtain bounds that more closely model the noise growth behaviour that we see in practice.

Moreover, any implementation of a scheme will have differences to the scheme in theory. For example, SEAL implements the BEHZ variant [6] of FV, and as a result many objects in SEAL are always stored and manipulated in an RNS format. In contrast, the bounds for FV presented in Section 4 assume the ciphertext is a pair of polynomials in R_q rather than an isomorphic RNS representation. This could partly explain the discrepancy we have seen between the heuristic bounds and the noise in SEAL ciphertexts. We therefore believe that an important direction to better model the noise growth behaviour is to tailor the analysis to the specific implementation. Future work should focus on obtaining such library-specific heuristic bounds, which would result in the most accurate and tight bounding of ciphertext noise growth in practice. This would be the best step towards automating parameter selection for correctness, which is crucial for the real world deployment of FHE [26, 47] and the ongoing standardisation process [1].

An important contribution of our work is evidence that in their implementations, BGV and FV present only minor performance differences, from the point

of view of possibility to support a specific homomorphic evaluation. In particular, from Section 6 we can conclude that a computation supported in SEAL by a particular parameter set would be supported in HElib with the same parameter set. Indeed, we have seen that the noise growth behaviour (for all definitions of noise) of BGV and FV is very similar. 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). We conclude that a preference for BGV over FV (or vice versa) should not be the deciding factor for the implementor choosing which library to use.

In practice, many things will impact the decision to use one scheme or library over another, such as computational performance, support for different programming languages, robustness and quality of available implementations, and the relative ease-of-use. In future work it would be interesting to include such aspects into a comparative analysis, although they can be hard to quantify.

Acknowledgements

Rachel Player was supported by the French Programme d'Investissement d'Avenir under national project RISQ P141580 and by the European Union PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701). We thank Shai Halevi for helpful comments on HElib and Nigel Smart for sharing the experimental code used in [22].

References

[1] Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption security standard. Technical report, HomomorphicEncryption.org, Toronto, Canada, November 2018.

[2] 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.

[3] 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.

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

[5] Ahmad Al Badawi, Yuriy Polyakov, Khin Mi Mi Aung, Bharadwaj Veeravalli, and Kurt Rohloff. Implementation and performance evaluation of RNS variants of the BFV homomorphic encryption scheme. Cryptology ePrint Archive, Report 2018/589, 2018. `https://eprint.iacr.org/2018/589`.

[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] Carl Bootland, Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Efficiently Processing Complex-Valued Data in Homomorphic Encryption. Proceedings of MathCrypt 2018, Journal of Mathematical Cryptology, to appear, 2018.

[9] 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.

[10] Christina Boura, Nicolas Gama, and Mariya 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. `https://eprint.iacr.org/2018/758`.

[11] 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.

[12] 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.

[13] Michael Brenner, Wei Dai, Shai Halevi, Kyoohyung Han, Amir Jalali, Miran Kim, Kim Laine, Alex Malozemoff, Pascal Paillier, Yuriy Polyakov, Kurt Rohloff, Erkay Savaş, and Berk Sunar. A standard API for RLWE-based homomorphic encryption. Technical report, HomomorphicEncryption.org, Redmond WA, USA, July 2017.

[14] Wouter Castryck, Ilia Iliashenko, and Frederik Vercauteren. Homomorphic $\mathrm{SIM}^2\mathrm{D}$ 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.

[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, and Rachel Player. Simple encrypted arithmetic library - SEAL v2.2. Technical report, 2017.

[17] 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.

[18] Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low level encoding of zero. Cryptology ePrint Archive, Report 2016/139, 2016. `http://eprint.iacr.org/2016/139`.

[19] Jung Hee Cheon, Jinhyuck Jeong, Joohee Lee, and Keewoo Lee. Privacy-preserving computations of predictive medical models with minimax approximation and non-adjacent form. 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 53–74. Springer, Heidelberg, April 2017.

[20] 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.

[21] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2016.

[22] 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.

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

[24] 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.

[25] 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.

[26] Roshan Dathathri, Olli Saarikivi, Hao Chen, Kim Laine, Kristin E. Lauter, Saeed Maleki, Madanlal Musuvathi, and Todd Mytkowicz. CHET: compiler and runtime for homomorphic evaluation of tensor programs. *CoRR*, abs/1810.00845, 2018.

[27] Yarkın Doröz, Yin Hu, and Berk Sunar. Homomorphic AES evaluation using the modified LTV scheme. *Designs, Codes and Cryptography*, 80(2):333–358, Aug 2016.

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

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

[30] 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.

[31] 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.

[32] 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`.

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

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

[35] 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.

[36] 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.

[37] 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.

[38] 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.

[39] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May / June 2010.

[40] 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`.

[41] 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.

[42] Sean Murphy and Rachel Player. A central limit framework for ring-lwe decryption. Cryptology ePrint Archive, Report 2019/452, 2019. `https://eprint.iacr.org/2019/452`.

[43] Sean Murphy and Rachel Player. δ-subgaussian random variables in cryptography. In J. Jang-Jaccard and F. Guo, editors, *ACISP 2019: The 24th Australasian Conference on Information Security and Privacy*, 2019. Available at `https://eprint.iacr.org/2017/698`.

[44] PALISADE v1.0. `https://git.njit.edu/palisade/PALISADE`, 2017. New Jersey Institute of Technology (NJIT).

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

[46] Simple Encrypted Arithmetic Library (release 3.1.0). `https://github.com/Microsoft/SEAL`, December 2018. Microsoft Research, Redmond, WA.

[47] Alexander Viand and Hossein Shafagh. Marble: Making fully homomorphic encryption accessible to all. In *Proceedings of the 6th Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, WAHC '18, pages 49–60, New York, NY, USA, 2018. ACM.

A Proof of Lemma 18

In this section, we provide the proof of Lemma 18.

Proof. Let $\mathtt{ct} = (c_0, c_1)$. By definition of modulus switching,

$$\mathtt{ct}_{\mathrm{mod}} = \left(\frac{p}{q}(c_0 + \delta_0), \frac{p}{q}(c_1 + \delta_1)\right),$$

for δ_i such that $\delta_i = -c_i \pmod{\frac{q}{p}}$ and $\delta_i = 0 \pmod{t}$. Then by definition of invariant noise in $\mathtt{ct}$,

$$\begin{aligned}
\mathtt{ct}_{\mathrm{mod}}(s) &= \frac{p}{q}(c_0 + \delta_0) + \left(\frac{p}{q}(c_1 + \delta_1)\right) s \\
&= \frac{p}{q}(c_0 + c_1 s) + \frac{p}{q}(\delta_0 + \delta_1 s) \\
&= \frac{p}{q}(m + v) + \frac{p}{q}(\delta_0 + \delta_1 s) \\
&= m - \frac{q-p}{q} m + \frac{p}{q} v + \frac{p}{q}(\delta_0 + \delta_1 s) .
\end{aligned}$$

Hence the invariant noise is given by $v_{\mathrm{mod}} = -\frac{q-p}{q} m + \frac{p}{q} v + \frac{p}{q}(\delta_0 + \delta_1 s)$ and can be bounded as

$$\begin{aligned}
\|v_{\mathrm{mod}}\|^{\mathrm{can}} &= \left\| -\frac{q-p}{q} m + \frac{p}{q} v + \frac{p}{q}(\delta_0 + \delta_1 s) \right\|^{\mathrm{can}} \\
&\leq \frac{q-p}{q} \|-m\|^{\mathrm{can}} + \left\| \frac{p}{q} v + \frac{p}{q}(\delta_0 + \delta_1 s) \right\|^{\mathrm{can}} \\
&\leq \frac{p}{q} \|v\|^{\mathrm{can}} + t\sqrt{n} \left(\frac{\sqrt{3}(q-p)}{q} + \sqrt{3} + \frac{8\sqrt{h}}{\sqrt{3}} \right),
\end{aligned}$$

using Lemma 6 and the bound $\|m\|^{\mathrm{can}} \leq t\sqrt{3n}$. □

B Proof of Lemma 21

In this section, we provide the proof of Lemma 21.

Proof. Firstly, note that we can bound $\|a_0\|^{\mathrm{can}}$ as follows:

$$\begin{aligned}
\|a_0\|^{\mathrm{can}} &= \frac{1}{t} \|[m_1 + m_2]_t - m_1 - m_2\|^{\mathrm{can}} \\
&\leq \frac{1}{t} \left(\|[m_1 + m_2]_t\|^{\mathrm{can}} + \|m_1\|^{\mathrm{can}} + \|m_2\|^{\mathrm{can}} \right) \\
&\leq \frac{1}{t} \left(3 \cdot t\sqrt{3n} \right) = 3\sqrt{3n} .
\end{aligned}$$

Let $\mathtt{ct}_1 = (c_0, c_1)$ and $\mathtt{ct}_2 = (d_0, d_1)$. By definition of $\mathtt{ct}_{\mathrm{add}}$, and by definition of noise in $\mathtt{ct}_1$ and $\mathtt{ct}_2$, for some integer polynomials k_0, k_1, a_1, a_2,

$$\frac{t}{q} \mathtt{ct}_{\mathrm{add}}(s) = \frac{t}{q} (c_0 + d_0 + k_0 q + c_1 s + d_1 s + k_1 q s)$$

$$
\begin{aligned}
&= \frac{t}{q}(c_0 + c_1 s) + \frac{t}{q}(d_0 + d_1 s) + t(k_0 + k_1 s) \\
&= \frac{t}{q}\Delta m_1 + v_1 + a_1 t + \frac{t}{q}\Delta m_2 + v_2 + a_2 t + t(k_0 + k_1 s) \\
&= \frac{t}{q}\Delta(m_1 + m_2) + v_1 + v_2 + t(a_1 + a_2 + k_0 + k_1 s) \\
&= \frac{t}{q}\Delta([m_1 + m_2]_t - a_0 t) + v_1 + v_2 + t(a_1 + a_2 + k_0 + k_1 s) \\
&= \frac{t}{q}\Delta([m_1 + m_2]_t - \frac{\Delta t^2}{q} a_0 + v_1 + v_2 + t(a_1 + a_2 + k_0 + k_1 s) \\
&= \frac{t}{q}\Delta([m_1 + m_2]_t - \frac{t(q - r_t(q))}{q} a_0 + v_1 + v_2 + t(a_1 + a_2 + k_0 + k_1 s) \\
&= \frac{t}{q}\Delta([m_1 + m_2]_t + \frac{r_t(q) \cdot t}{q} a_0 + v_1 + v_2 + t(-a_0 + a_1 + a_2 + k_0 + k_1 s)\,.
\end{aligned}
$$

Hence by definition the scaled inherent noise in this ciphertext is $v_{\text{add}} = v_1 + v_2 + \frac{t \cdot r_t(q)}{q} a_0$. This can be bounded as

$$
\begin{aligned}
\|v_{\text{add}}\|^{\text{can}} &= \left\| v_1 + v_2 + \frac{r_t(q)}{q} a_0 t \right\|^{\text{can}} \\
&\leq \|v_1\|^{\text{can}} + \|v_2\|^{\text{can}} + \frac{t \cdot r_t(q)}{q} \|a_0\|^{\text{can}}\,,
\end{aligned}
$$

and the result follows. □

C Proof of Lemma 22

In this section, we provide the proof of Lemma 22.

Proof. Throughout the proof we model a message as a polynomial with random coefficients in $[-\frac{t}{2}, \frac{t}{2}]$ as in [22], so that $\|m\|^{\text{can}} \leq t\sqrt{3n}$. We will frequently use that fact that $q = \Delta t + r_t(q)$, so that $\frac{\Delta t}{q} = \frac{q - r_t(q)}{q} = 1 - \frac{r_t(q)}{q}$. We first establish some bounds on certain terms that will appear later.

Let a_0 be an integer polynomial such that $[m_1 m_2]_t = m_1 m_2 + a_0 t$. We can bound a_0 as follows:

$$
\begin{aligned}
\|a_0\|^{\text{can}} &= \frac{1}{t}\|[m_1 m_2]_t - m_1 m_2\|^{\text{can}} \\
&\leq \frac{1}{t}\left(\|[m_1 m_2]_t\|^{\text{can}} + \|m_1 m_2\|^{\text{can}}\right) \\
&\leq \frac{1}{t}\left(t\sqrt{3n} + 16 \cdot \frac{t\sqrt{n}}{\sqrt{12}} \cdot \frac{t\sqrt{n}}{\sqrt{12}}\right) \\
&\leq \sqrt{3n} + \frac{4tn}{3}\,.
\end{aligned}
$$

For $i \in \{0,1,2\}$ let ϵ_i be polynomials with coefficients uniformly distributed in $[-\frac{1}{2}, \frac{1}{2}]$. We can bound $\left\| \frac{t}{q} \sum_{i=0}^{2} \epsilon_i s^i \right\|^{\text{can}}$ as follows [45]:

$$\begin{aligned}
\left\| \frac{t}{q} \sum_{i=0}^{2} \epsilon_i s^i \right\|^{\text{can}} &\leq \frac{t}{q} \left(\|\epsilon_0\|^{\text{can}} + \|\epsilon_1 \cdot s\|^{\text{can}} + \|\epsilon_2 \cdot s \cdot s\|^{\text{can}} \right) \\
&\leq \frac{t}{q} \left(6 \cdot \frac{\sqrt{n}}{\sqrt{12}} + 16 \cdot \frac{\sqrt{n}}{\sqrt{12}} \cdot \sqrt{\frac{2}{3}n} + 40 \cdot \frac{\sqrt{n}}{\sqrt{12}} \cdot \sqrt{\frac{2}{3}n} \cdot \sqrt{\frac{2}{3}n} \right) \\
&= \frac{2t\sqrt{n}}{q\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) .
\end{aligned}$$

For $j \in \{1,2\}$ we can bound $\|\mathtt{ct}_j(s)\|^{\text{can}}$ as follows [45]:

$$\begin{aligned}
\|\mathtt{ct}_j(s)\|^{\text{can}} &= \left\| c_{j,0} + c_{j,1}s + c_{j,2}s^2 \right\|^{\text{can}} \\
&\leq \|c_{j,0}\|^{\text{can}} + \|c_{j,1}s\|^{\text{can}} + \left\| c_{j,2}s^2 \right\|^{\text{can}} \\
&\leq 6 \cdot \frac{q\sqrt{n}}{\sqrt{12}} + 16 \cdot \frac{q\sqrt{n}}{\sqrt{12}} \cdot \sqrt{\frac{2}{3}n} + 40 \cdot \frac{q\sqrt{n}}{\sqrt{12}} \cdot \sqrt{\frac{2}{3}n} \cdot \sqrt{\frac{2}{3}n} \\
&= \frac{2q\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) .
\end{aligned}$$

This enables us to bound $\|a_j t\|^{\text{can}}$ as follows:

$$\begin{aligned}
\|a_j t\|^{\text{can}} &= \left\| \frac{t}{q}\mathtt{ct}_j(s) - \frac{t}{q}\Delta m_j - v_j \right\|^{\text{can}} \\
&\leq \frac{t}{q} \|\mathtt{ct}_j(s)\|^{\text{can}} + \frac{\Delta t}{q} \|m_j\|^{\text{can}} + \|v_j\|^{\text{can}} \\
&\leq \frac{2t\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{\Delta t^2 \sqrt{3n}}{q} + \|v_j\|^{\text{can}} .
\end{aligned}$$

Let $\mathtt{ct}_1 = (c_0, c_1)$ and $\mathtt{ct}_2 = (d_0, d_1)$. By definition of $\mathtt{ct}_{\text{mult}}$ and by the definition of noise in $\mathtt{ct}_1$ and $\mathtt{ct}_2$ we have, for some polynomials $\|\epsilon_i\| \leq \frac{1}{2}$ and for some integer polynomials k_i,

$$\begin{aligned}
\frac{t}{q}\mathtt{ct}_{\text{mult}}(s) &= \frac{t}{q} \left(\frac{t}{q} c_0 d_0 + \epsilon_0 + k_0 q + \frac{t}{q}(c_0 d_1 + c_1 d_0)s + \epsilon_1 s + k_1 q s \right) \\
&\quad + \frac{t}{q} \left(\frac{t}{q}(c_1 d_1)s^2 + \epsilon_2 s^2 + k_2 q s^2 \right) \\
&= \frac{t}{q}(c_0 + c_1 s) \cdot \frac{t}{q}(d_0 + d_1 s) + \frac{t}{q} \left(\sum_{i=0}^{2} \epsilon_i s^i \right) + t \left(\sum_{i=0}^{2} k_i s^i \right) \\
&= \left(\frac{t}{q}\Delta m_1 + v_1 + a_1 t \right) \cdot \left(\frac{t}{q}\Delta m_2 + v_2 + a_2 t \right) + \frac{t}{q} \left(\sum_{i=0}^{2} \epsilon_i s^i \right)
\end{aligned}$$

$$
\begin{aligned}
&\quad + t\left(\sum_{i=0}^{2} k_i s^i\right) \\
&= \frac{(t\Delta)^2}{q^2} m_1 m_2 + \frac{t\Delta}{q}(m_2 v_1 + m_1 v_2) + \frac{t\Delta}{q}(m_2 a_1 t + m_1 a_2 t) + v_1 v_2 \\
&\quad + v_2 a_1 t + v_1 a_2 t + \frac{t}{q}\left(\sum_{i=0}^{2} \epsilon_i s^i\right) + t\left(\sum_{i=0}^{2} k_i s^i + a_1 a_2 t\right) .
\end{aligned}
$$

Considering the first term, we have that

$$
\begin{aligned}
\frac{(t\Delta)^2}{q^2} m_1 m_2 &= \frac{(t\Delta)^2}{q^2}\left([m_1 m_2]_t - a_0 t\right) \\
&= \frac{t\Delta}{q}\left(1 - \frac{r_t(q)}{q}\right)\left([m_1 m_2]_t - a_0 t\right) \\
&= \frac{t\Delta}{q}\left([m_1 m_2]_t - a_0 t - \frac{r_t(q)}{q}[m_1 m_2]_t + \frac{r_t(q)}{q} a_0 t\right) \\
&= \frac{t}{q}\Delta[m_1 m_2]_t - \frac{t^2\Delta}{q} a_0 - \frac{r_t(q)\Delta t}{q^2}[m_1 m_2]_t + \frac{t^2 r_t(q)\Delta}{q^2} a_0 .
\end{aligned}
$$

We also note that

$$
-\frac{t^2\Delta}{q} a_0 = -a_0 t + \frac{r_t(q)}{q} t a_0
$$

and

$$
\frac{t\Delta}{q}(m_2 a_1 t + m_1 a_2 t) = m_2 a_1 t + m_1 a_2 t - \frac{r_t(q)}{q}(m_2 a_1 t + m_1 a_2 t).
$$

Hence, $\frac{t}{q}\mathtt{ct}_{\mathrm{mult}}(s) = \frac{t}{q}\Delta[m_1 m_2]_t + v_{\mathrm{mult}} + at$, for the integer polynomial

$$
a = \sum_{i=0}^{2} k_i s^i + a_1 a_2 t - a_0 + m_2 a_1 + m_1 a_2 .
$$

and the noise

$$
\begin{aligned}
v_{\mathrm{mult}} = &\frac{r_t(q)}{q} t a_0 - \frac{r_t(q)\Delta t}{q^2}[m_1 m_2]_t + \frac{t^2 r_t(q)\Delta}{q^2} a_0 + \frac{t\Delta}{q}(m_2 v_1 + m_1 v_2) \\
&- \frac{r_t(q)}{q}(m_2 a_1 t + m_1 a_2 t) + v_1 v_2 + v_2 a_1 t + v_1 a_2 t + \frac{t}{q}\left(\sum_{i=0}^{2} \epsilon_i s^i\right) .
\end{aligned}
$$

We previously established a bound for the final term in the expression. Let us bound the remaining terms in turn. Firstly,

$$
\left\| \frac{r_t(q)}{q} t a_0 \right\|^{\mathrm{can}} = \frac{r_t(q)\cdot t}{q} \left\| a_0 \right\|^{\mathrm{can}} \leq \frac{r_t(q)\cdot t\sqrt{3n}}{q} + \frac{4\cdot r_t(q)\cdot t^2 n}{3q} .
$$

Next,

$$\left\| -\frac{r_t(q)\Delta t}{q^2}[m_1 m_2]_t \right\|^{\text{can}} = \frac{r_t(q)\Delta t}{q^2} \left\| [m_1 m_2]_t \right\|^{\text{can}} \leq \frac{r_t(q) \cdot \Delta t^2 \sqrt{3n}}{q^2} .$$

Next,

$$\left\| \frac{t^2 r_t(q)\Delta}{q^2} a_0 \right\|^{\text{can}} = \frac{t^2 r_t(q)\Delta}{q^2} \left\| a_0 \right\|^{\text{can}} \leq \frac{t^2 \cdot r_t(q) \cdot \Delta\sqrt{3n}}{q^2} + \frac{4t^3 n \cdot r_t(q) \cdot \Delta}{3q^2} .$$

Next,

$$\begin{aligned} \left\| \frac{t\Delta}{q}(m_2 v_1 + m_1 v_2) \right\|^{\text{can}} &= \frac{t\Delta}{q} \left(\left\| m_2 \right\|^{\text{can}} \left\| v_1 \right\|^{\text{can}} + \left\| m_1 \right\|^{\text{can}} \left\| v_2 \right\|^{\text{can}} \right) \\ &\leq \frac{\Delta t^2 \sqrt{3n}}{q} \left(\left\| v_1 \right\|^{\text{can}} + \left\| v_2 \right\|^{\text{can}} \right) . \end{aligned}$$

Next,

$$\begin{aligned} &\left\| -\frac{r_t(q)}{q}(m_2 a_1 t + m_1 a_2 t) \right\|^{\text{can}} \leq \frac{r_t(q)}{q} \left(\left\| m_2 \right\|^{\text{can}} \left\| a_1 t \right\|^{\text{can}} + \left\| m_1 \right\|^{\text{can}} \left\| a_2 t \right\|^{\text{can}} \right) \\ &\leq \frac{r_t(q) t \sqrt{3n}}{q} \left(\frac{2t\sqrt{n}}{\sqrt{3}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{2\Delta t^2 \sqrt{3n}}{q} + \left\| v_1 \right\|^{\text{can}} + \left\| v_2 \right\|^{\text{can}} \right) . \end{aligned}$$

Next,

$$\left\| v_1 v_2 \right\|^{\text{can}} \leq \left\| v_1 \right\|^{\text{can}} \left\| v_2 \right\|^{\text{can}} .$$

Finally,

$$\begin{aligned} \left\| v_2 a_1 t + v_1 a_2 t \right\|^{\text{can}} &\leq \left\| v_2 \right\|^{\text{can}} \left(\frac{2t\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{\Delta t^2 \sqrt{3n}}{q} + \left\| v_1 \right\|^{\text{can}} \right) \\ &\quad + \left\| v_1 \right\|^{\text{can}} \left(\frac{2t\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{\Delta t^2 \sqrt{3n}}{q} + \left\| v_2 \right\|^{\text{can}} \right) \\ &\leq \left(\left\| v_1 \right\|^{\text{can}} + \left\| v_2 \right\|^{\text{can}} \right) \left(\frac{2t\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{\Delta t^2 \sqrt{3n}}{q} \right) \\ &\quad + 2 \left\| v_1 \right\|^{\text{can}} \cdot \left\| v_2 \right\|^{\text{can}} . \end{aligned}$$

We therefore have $v_{\text{mult}} \leq A \left\| v_1 \right\|^{\text{can}} \left\| v_2 \right\|^{\text{can}} + B \left(\left\| v_1 \right\|^{\text{can}} + \left\| v_2 \right\|^{\text{can}} \right) + C$, where $A = 3$, and B and C are as follows:

$$\begin{aligned} B &= \frac{\Delta t^2 \sqrt{3n}}{q} + \frac{r_t(q) t \sqrt{3n}}{q} + \frac{2t\sqrt{n}}{\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{\Delta t^2 \sqrt{3n}}{q} , \\ C &= \frac{r_t(q) t \sqrt{3n}}{q} + \frac{4 r_t(q) t^2 n}{3q} + \frac{r_t(q) \Delta t^2 \sqrt{3n}}{q^2} + \frac{t^2 r_t(q) \Delta \sqrt{3n}}{q^2} \\ &\quad + \frac{4t^3 n r_t(q) \cdot \Delta}{3q^2} + \frac{r_t(q) t \sqrt{3n}}{q} \left(\frac{4t\sqrt{n}}{\sqrt{3}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) + \frac{2\Delta t^2 \sqrt{3n}}{q} \right) \end{aligned}$$

$$+ \frac{2t\sqrt{n}}{q\sqrt{12}} \left(3 + \frac{8\sqrt{2}}{\sqrt{3}}\sqrt{n} + \frac{40}{3}n \right) .$$

These expressions can be simplified to give the stated noise bound. □