Optimisations and Tradeoffs for HElib

Anamaria Costache[1], Lea Nürnberger[1], and Rachel Player[2]

[1] Norwegian University of Science and Technology (NTNU), Norway
[2] Royal Holloway, University of London, UK
anamaria.costache@ntnu.no, lea.nurnberger@ntnu.no, rachel.player@rhul.ac.uk

Abstract. In this work, we investigate the BGV scheme as implemented in HElib. We begin by performing an implementation-specific noise analysis of BGV. This allows us to derive much tighter bounds than what was previously done. To confirm this, we compare our bounds against the state of the art. We find that, while our bounds are at most 1.8 bits off the experimentally observed values, they are as much as 29 bits tighter than previous work. Finally, to illustrate the importance of our results, we propose new and optimised parameters for HElib. In HElib, the special modulus is chosen to be k times larger than the current ciphertext modulus Q_i. For a ratio of subsequent ciphertext moduli $\log(\frac{Q_i}{Q_{i-1}}) = 54$ (a very common choice in HElib), we can optimise k by up to 26 bits. This means that we can either enable more multiplications without having to switch to larger parameters, or reduce the size of the evaluation keys, thus reducing on communication costs in relevant applications. We argue that our results are near-optimal.

1 Introduction

Fully Homomorphic Encryption (FHE) is a type of encryption that allows to compute on encrypted data. An open problem for nearly three decades, the first construction came in 2009 from Gentry [18]. Since then, the field has seen some spectacular advances, and there are now several widely used and implemented schemes, each with various tradeoffs. Loosely speaking, these all fit into four generations. The first generation refers to the original construction [18] and its variants. The second generation includes the BGV [4] and BFV [3, 17] schemes. The third generation includes the CGGI scheme [6, 7], which was developed from the line of work [21, 16]. Finally, the fourth generation consists of the approximate homomorphic scheme CKKS [5] and its numerous variants. The above named schemes all base their security on variants of the Learning With Errors problem (LWE) [32], and are currently being standardised.

In this work, we focus on the BGV scheme [4], which has been implemented in several open source libraries, including HElib [23], PALISADE [31], SEAL [33] and Lattigo [27]. The implementation in HElib was the first public implementation of BGV, and remains actively maintained. It has been used in several applications [1, 10, 15, 20].

BGV does not follow the Gentry blueprint [18] of building a somewhat homomorphic encryption scheme and then bootstrapping it to obtain a fully homomorphic scheme. Instead, it uses *levels*, which can be thought of as layers of the ciphertext ring. We encrypt at the top level, and switch down one level after each multiplication, until we reach a final level where no more multiplications are possible without incorrect decryption. In this setting, the circuit to be evaluated must be fixed in advance, and large enough parameters must be chosen so that there are enough levels to support the required depth of the circuit.

The levelled approach is proposed in [4] as a *noise management* technique. Noise is a feature of all ciphertexts in all LWE-based homomorphic encryption schemes, and is essential for security. The noise grows with each homomorphic operation, particularly so with multiplication, and if it becomes too large then decryption will fail. A good understanding of noise growth is therefore necessary to balance correctness, security and performance requirements.

Several noise analyses of BGV have been presented in prior work [11, 13, 19, 20, 22, 25, 30]. Most approaches give a worst-case bound on the canonical norm [11, 13, 19, 20] (defined below) or infinity norm [25] of the noise after each BGV operation. In [13], it was observed that there can be a large gap between the noise predicted by such bounds and the actual observed noise in BGV ciphertexts as implemented in HElib. This can be explained by the inherent looseness of the bounds compounding as we move through the circuit.

To mitigate this, an *average-case* approach for BGV noise anaylsis was presented in [30], that built upon a similar analysis for the CKKS scheme that was presented in [12], in analogue to the approach taken for the CGGI scheme in [8, 9]. The main idea is to track the variance of the noise through each operation, arriving at a variance for the noise in the output ciphertext, which can then be bounded. Experiments in [30], using implementations of BGV in HElib and in SEAL, showed that, while the gap identified in [13] between the predicted and observed noise is narrowed when using this average-case approach, it is not completely closed. Moreover, the gap was seen to be wider for HElib than for SEAL. It was suggested in [30] that this could be explained by the different implementation choices in HElib and SEAL, but providing and evaluating an implementation-specific noise analysis of BGV was left as an open problem.

1.1 Our Contributions

In this paper, we give for the first time a noise analysis for BGV that is specifically adapted to its implementation in HElib, as described in [22]. It follows a similar approach as in [8, 9, 12, 30], in that we present results for how the variance of the noise develops through the stages of homomorphic multiplication. However, in contrast to [30], we focus not just on BGV ciphertext noise, but on BGV as implemented in HElib. Further, we evaluate the efficacy of our approach, and discuss its utility and applicability.

In more detail, we confirm that our analysis resolves the open question posed in [30], by experimentally verifying that our theoretical results for the variance of the noise (Corollaries 2 and 3) empirically match the variance of the noise observed in HElib ciphertexts (Tables 1 and 3). We thereby demonstrate that our theoretical analysis of the variance is tight and any eventual loss in the tightness comes from the final bounding step.

Additionally, we present a detailed comparison to prior noise analyses for BGV. The results show that our approach leads to closer modelling of the noise and consequently tighter bounds. This applies both for prior works using bounds on the canonical norm (Table 4) and the infinity norm (Table 6). We see for example in Table 4, for a ring size $n = 32768$, that our theoretical bounds are up to 29 bits tighter than those in [22] and up to 9 bits tighter than those in [13], whilst being at most 1.8 bits off the observed experimental values.

An interesting finding of our comparison was that applying previous analyses for BGV, such as the work [25] that was developed considering PALISADE [31], may underestimate the observed HElib noise. This means that relying on such analyses to estimate the noise growth in HElib ciphertexts might lead to decryption errors. This observation further emphasises the value of implementation specific noise analyses.

Finally, we use our results to propose new parameters in HElib. Specifically, we demonstrate that our analysis allows to optimize the ratio between ciphertext moduli in the moduli chain that express how the levels are made up in HElib. In HElib, the special modulus is chosen to be k times larger than the current ciphertext modulus Q_i. In Section 6 we show that, for a ratio of subsequent ciphertext moduli $\log(\frac{Q_i}{Q_{i-1}}) = 54$ (a very common choice in HElib), we can optimise k by up to 26 bits. Our work enables the following tradeoff. On the one hand, it could be used to allow more moduli to be included in the chain, and thus we can permit a greater multiplicative depth for a fixed parameter set. This means we can evaluate higher-depth computations without having to switch to a larger parameter set and incurring a consequent performance slow down. On the other hand, it could be used to reduce the size of evaluation keys, and hence represents an improvement in communication costs.

1.2 Structure of the Paper

In Section 2 we introduce notation and the necessary background. In Section 3 we present our implementation-specific noise analysis for BGV as implemented in HElib. In Section 4 we experimentally verify the theoretical analysis that we have developed. In Section 5 we compare our approach with prior analyses of BGV noise growth. In Section 6 we demonstrate how our analysis can be applied to optimize parameter selection in HElib.

2 Preliminaries

2.1 Notation

Vectors are denoted by a small bold letter $\mathbf{z}$, where z_i denotes its i^{th} component. In a slight abuse of notation, for a polynomial $a \in \mathcal{R}$, where $\mathcal{R}$ is a polynomial ring of degree n, we denote by $a[i]$ the i-th coefficient of a. It can be thought of as the i-th element in the coefficient vector of a. The notation $[\cdot]_q$ denotes reduction modulo q (coefficient wise, when applied to a polynomial). The notation $\lceil \cdot \rfloor$ denotes rounding to the nearest integer (coefficient wise, when applied to a polynomial). Unless otherwise specified, log denotes $\log_2$.

We denote by σ^2 a variance, σ a standard deviation and μ the mean of any distribution, while σ^2_{est}, σ_{est} and μ_{est} denote their point estimators. Let $\mathcal{N}(\mu, \sigma)$ be the normal distribution with mean μ and standard deviation σ. For any distribution $\mathcal{D}$ we denote by $x \leftarrow \mathcal{D}$ the fact that x has been drawn from $\mathcal{D}$. For any set S, $x \xleftarrow{\$} S$ denotes the fact that x has been sampled uniformly at random from S.

2.2 Point Estimators for Variance and Standard Deviation

Let $x_i \leftarrow \mathcal{D}(\sigma^2)$ for $1 \leq i \leq w$ be samples drawn from an unknown distribution, with unknown variance σ^2 and let $\overline{x}$ be their mean. We can estimate the variance and standard deviation of $\mathcal{D}$ as follows. The (biased) sample variance is defined as:

$$\sigma^2_{\text{biased}} = \frac{1}{w} \sum_{i=1}^{w} (x_i - \overline{x})^2 \,.$$

It can be shown that the expectation $\mathbb{E}[\sigma^2_{\text{biased}}] = \frac{w-1}{w}\sigma^2$ and hence the obtained estimation is biased. To avoid this, we will use the unbiased sample variance

$$\sigma^2_{\text{est}} = \frac{w}{w-1} \cdot \sigma^2_{\text{biased}} = \frac{1}{w-1} \sum_{i=1}^{w} (x_i - \overline{x})^2 \,.$$

From this, the standard deviation σ is estimated via $\sigma_{\text{est}} = \sqrt{\sigma^2_{\text{est}}}$. Since σ_{est} is obtained from σ^2_{est} through a non-linear operation, it is no longer unbiased. For a big enough sample size, the bias is however negligible.

2.3 Algebraic Background

We let $\mathcal{R} = \mathbb{Z}[x]/(x^m + 1)$, the cyclotomic ring of dimension $n = \phi(m)$, where $\phi(\cdot)$ is Euler's Totient Function. For m is a power of two, we have $\phi(m) = m/2$.

To represent polynomials in $\mathcal{R}$ as vectors we can use both the coefficient embedding and the canonical embedding. For a polynomial $a \in \mathcal{R}$, expressed as $a = a_0 + \ldots + a_{n-1}x^{n-1}$, its coefficient embedding is the vector $(a_0, \ldots, a_{n-1})$.

To define the canonical embedding, let ζ_m be a primitve m^{th} root of unity and $\mathbb{Q}(\zeta_m)$ the m^{th} cyclotomic number field obtained as a field extension of $\mathbb{Q}$ by adjoining ζ_m. There are n ring embeddings $\sigma_1, \ldots, \sigma_n : \mathbb{Q}(\zeta_m) \hookrightarrow \mathbb{C}$ given by $\zeta_m \mapsto \zeta_m^k$ for $k \in \{1, \ldots, n\}$. The canonical embedding of an element $p \in \mathbb{Q}(\zeta_m)$ is given via $p \mapsto (\sigma_1(p), \ldots, \sigma_n(p))^T$.

The canonical norm of an element $p \in \mathbb{Q}(\zeta_m)$ is denoted as $\|p\|^{\text{can}}$ and is the infinity norm of the embedded vector. The following bound on the canonical norm of a random polynomial is proved in Section 2.8 of [24].

Lemma 1 ([24]). *Let $a \leftarrow \mathcal{R}_q$ be a random polynomial and let $\sigma^2_{a[i]}$ be the variance of each coefficient in the powerful basis $(\zeta_m, \ldots, \zeta_m^n)$. The random variable $a(\zeta_m^k)$ for $k \in \{1, \ldots, n\}$ has variance $\sigma^2_{a(\zeta_m^k)} = \sigma^2_{a[i]} n$, and the canonical norm of a can be bounded by*

$$\|a\|^{can} \leq 6\sqrt{\sigma^2_{a[i]} n}\,.$$

We denote by $\|p\|_\infty$ the infinity norm of the coefficient embedding of p. For $a, b \in \mathcal{R}$ and for $\gamma_{\mathcal{R}}$ the expansion factor [28] of $\mathcal{R}$, it holds that

$$||ab||_\infty \leq \gamma_{\mathcal{R}} ||a||_\infty ||b||_\infty.$$

For an n-dimensional power of two cyclotomic ring $\mathcal{R}$ we have $\gamma_{\mathcal{R}} = n$. To bound the infinity norm of polynomials whose coefficients are normally distributed, we will use the following well-known fact.

Lemma 2. *Let $v \sim \mathcal{N}(0, \sigma)$ and let erf$(\cdot)$ be the error function. Then v lies in the interval $(-a, a)$ with probability*

$$erf\left(\frac{a}{\sigma\sqrt{12}}\right).$$

For a vector $\mathbf{v}$, whose entries are identically and independently normally distributed with mean 0 and variance σ^2, each entry is smaller than an $a \in \mathbb{R}$, with the above stated probability. That is, we have

$$\mathbb{P}(||\mathbf{v}||_\infty) \leq a) = \operatorname{erf}\left(\frac{a}{\sigma\sqrt{2}}\right).$$

For $a = 10\sigma$, $||\mathbf{v}||_\infty > 10\sigma$ is true with probability smaller than 2^{-75}.

2.4 The BGV Scheme

The BGV scheme [4] is a levelled FHE scheme based on the Ring-LWE problem [29]. The ciphertext space is $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^m + 1)$, where q is the ciphertext modulus. The plaintext space is $\mathcal{R}_t = \mathbb{Z}_t[x]/(x^m + 1)$, where t is the plaintext modulus. Messages and ciphertexts will be considered as polynomials in $\mathcal{R}_t$ and $\mathcal{R}_q$, respectively.

The BGV scheme is parametrised by the ring dimension n, the plaintext modulus t; the length L of the moduli chain $Q_L \gg \ldots \gg Q_0$, where $Q_i | Q_{i+1}$ for $i \in \{0, \ldots, L-1\}$; the decomposition base ω; the security parameter λ; the secret key distribution $\mathcal{S}$; and the error distribution χ.

BGV consists of the algorithms `KeyGen`, `Encrypt`, `Decrypt`, `Add`, `PreMult`, `KeySwitch` and `ModSwitch`, defined as follows.

`KeyGen`(1^λ): Draw $s \leftarrow \mathcal{S}$ and set $(1, s) := \mathtt{sk}$ as the secret key. Sample $a \xleftarrow{\$} \mathcal{R}_q$ and $e \leftarrow \chi$. Set $\mathtt{pk} = (\mathtt{pk}[0], \mathtt{pk}[1]) := ([-as - te]_{Q_L}, a)$ as the public key. For $i \in \{0, \ldots, \log_\omega(Q_L)\}$ sample $a_i \xleftarrow{\$} \mathcal{R}_{Q_L}$ and $e_i \leftarrow \chi$ and set $\mathtt{evk} := ([-a_i s - te_i + \omega^i s^2]_{Q_L}, a_i)$. Return $(\mathtt{sk}, \mathtt{pk}, \mathtt{evk})$.

$\texttt{Encrypt}(\texttt{pk}, m)$: Let $m \in \mathcal{R}_t$ be a message. Let $Q_i, i \in \{0, \dots, L\}$ be the modulus in the moduli chain corresponding to the current level. Sample $u \leftarrow \mathcal{S}$ and $e_1, e_2 \leftarrow \chi$. Return $\texttt{ct} = (\texttt{ct}[0], \texttt{ct}[1]) \coloneqq ([m + \texttt{pk}[0]u + te_1]_{Q_i}, [\texttt{pk}[1]u + te_2]_{Q_i})$.

$\texttt{Decrypt}(\texttt{sk},\texttt{ct})$: Return $m' = [[< \texttt{ct}, \texttt{sk} >]_{Q_i}]_t$.

$\texttt{Add}(\texttt{ct}_0, \texttt{ct}_1)$: Return $\texttt{ct} \coloneqq ([\texttt{ct}_0[0] + \texttt{ct}_1[0]]_{Q_i}, [\texttt{ct}_0[1], \texttt{ct}_1[1]]_{Q_i})$.

$\texttt{PreMult}(\texttt{ct}_0, \texttt{ct}_1)$: Return $\texttt{ct}^{pm} = (\texttt{ct}^{pm}[0], \texttt{ct}^{pm}[1], \texttt{ct}^{pm}[2]) \coloneqq ([\texttt{ct}_0[0]\texttt{ct}_1[0]]_{Q_i}, [\texttt{ct}_0[0]\texttt{ct}_1[1] + \texttt{ct}_0[1]\texttt{ct}_1[0]]_{Q_i}, [\texttt{ct}_0[1]\texttt{ct}_1[1]]_{Q_i})$.

$\texttt{KeySwitch}(\texttt{ct},\texttt{evk})$: Let $\texttt{ct} = (\texttt{ct}[0], \texttt{ct}[1], \texttt{ct}[2])$. Set for the decomposition base $\omega^j = D_j^\star = D_1 \dots D_{j-1}$, where the D_h are such that $Q_i = \prod_{h=1}^{\ell} D_h$. Define $\texttt{ct}_j[2]$ such that

$$\texttt{ct}[2] = \sum_{j=1}^{\ell} \texttt{ct}_j[2] D_j^\star.$$

Define the matrix A_i to switch keys from s_i to s as the matrix whose j^{th} row $a_{ij} = (a_{ij}[0], a_{ij}[1])$ is an encryption of $kQ_{j-1}s_i$ under $\texttt{sk}$ with respect to a bigger ciphertext modulus $Q = kQ_i$, $\gcd(k, Q_i) = 1$. Output

$$\texttt{ct}^{ks} \coloneqq k(\texttt{ct}[0], \texttt{ct}[1]) + \sum_{j=1}^{\ell} (\texttt{ct}_j[2] a_{2j}[1], \texttt{ct}_j[2] a_{2j}[1]).$$

$\texttt{ModSwitch}(\texttt{ct},Q_j)$: Let $\texttt{ct} = (\texttt{ct}[0], \texttt{ct}[1])$. Return $\texttt{ct}^{ms} \coloneqq \left(\left\lfloor \frac{Q_j}{Q_i} \texttt{ct}[0] \right\rceil_t, \left\lfloor \frac{Q_j}{Q_i} \texttt{ct}[1] \right\rceil_t \right)$, where $\left\lfloor \frac{Q_{i-1}}{Q} \texttt{ct}[i] \right\rceil_t$ denotes the rounding of the coefficients of the scaled ciphertext such that it encrypts the same message modulo t as the unscaled ciphertext.

In BGV, one multiplication consists of the following three steps: `PreMult`, `KeySwitch` and `ModSwitch`. When used as super- or subscripts, the notation pm, ks, and ms indicates that the object relates to the result of a BGV `PreMult`, `KeySwitch` or `ModSwitch` operation, respectively.

2.5 The HElib Library

HElib [22] provides a widely used implementation of BGV. In the original presentation of BGV [4], the secret key distribution $\mathcal{S}$ is a discrete gaussian with standard deviation $\sigma = 3.2$. In HElib, $\mathcal{S}$ is the following ternary distribution: for a specified hamming weight h, a coefficient is chosen to be 0 with probability $\frac{n-h}{n}$, and ± 1 with probability $\frac{h}{2n}$. In the case of dense keys and m a power of two, h is set to be $h \coloneqq \frac{n}{2}$. Hence, we have $\mathbb{E}(\mathcal{S}) = 0$ and the variance $\sigma_{\mathcal{S}}^2 = \frac{h}{n}$.

Since version 1.0.0 [23], the moduli chain is parametrised by `bits` and δ, instead of by the number of multiplicative levels L. The parameter `bits` gives the length of the top modulus of the ciphertext moduli in bits. The special modulus used for key switching is then chosen to be about k times the size of the current ciphertext modulus Q_i, where $\gcd(k, Q_i) = 1$. The parameter δ gives the relation in size between the moduli in the modulus chain. The plaintext modulus is given by the exponent $t = p^r$ and the number of plaintext slots by a parameter s. In our experiments, we will use $t = 3$ and $s = 1$. The parameter `c` defines the number of lines in the key switching matrix. The default $\mathtt{c} = 2$ is recommended by HElib.

2.6 Noise Definition

The definition of the noise or error in a BGV ciphertext varies in different sources. HElib uses the *critical quantity*, as defined in [11].

Definition 1 ([11]). *Let* $\mathtt{ct}$ *be a BGV ciphertext, encrypting a message* $m \in \mathcal{R}_t$ *with respect to a ciphertext modulus* q *and secret key* $\mathtt{sk} = (1, s)$*. The* critical quantity *of* $\mathtt{ct}$ *is defined as:*

$$v = [< \mathtt{ct}, \mathtt{sk} >]_q.$$

We will compare our analysis with that of [25], who define the *noise* in a BGV ciphertext as follows.

Definition 2 ([25]). *Let* $\mathtt{ct}$ *be a BGV ciphertext, encrypting a message* $m \in \mathcal{R}_t$ *with respect to a ciphertext modulus* q *and secret key* $\mathtt{sk}$*. The noise* e *of* $\mathtt{ct}$ *is defined as*

$$e = \frac{1}{t}([< \mathtt{ct}, \mathtt{sk} >]_q - m).$$

The critical quantity determines whether decryption will be correct, since it is an intermediate result in the decryption process. As such, we view it as the more natural definition. On the other hand, the noise as in Definition 2 looks at the ciphertext noise independent of the message and the plaintext modulus. Since both the message and the plaintext modulus are fixed for a fixed ciphertext, both quantities can be computed from one another, therefore the two definitions are essentially equivalent.

3 Noise Heuristics for HElib Ciphertexts

In this section we give heuristics for the variance of the critical quantity after both the `PreMult` and `ModSwitch` operations for BGV as implemented in HElib. We first give expressions for the relevant critical quantities. We then determine the required variances of these critical quantities. Our analysis relies on the following result on the variance of the product of two polynomials.

Lemma 3. *Let* $f, g \in \mathcal{R}$ *be two polynomials of degree* n*, whose coefficients are drawn identically and independently from two distributions* $\mathcal{D}_f$ *and* $\mathcal{D}_g$ *:*

$$f[i] \xleftarrow{i.i.d} \mathcal{D}_f(\mu_f, \sigma_f^2), \quad g[i] \xleftarrow{i.i.d} \mathcal{D}_g(\mu_g, \sigma_g^2),$$

$i \in \{1, \ldots, n\}$, *where* μ_j *is the mean and* σ_j^2 *is the variance of* $\mathcal{D}_j$ *respectively. Let* $\mathbb{E}(\mathcal{D}_j)$ *denote the expectation of* $\mathcal{D}_j$, $j \in \{f, g\}$. *Then the variance of the distribution of the coefficients of* $f \cdot g$ *is:*

$$\sigma^2_{(fg)[i]} = n(\mathbb{E}(\mathcal{D}_f)^2 \sigma_g^2 + \mathbb{E}(\mathcal{D}_g)^2 \sigma_f^2 + \sigma_g^2 \sigma_f^2).$$

Proof. The coefficients of the product of two polynomials f, $g \in \mathcal{R}$ is given in [24] as

$$(fg)[i] = \sum_{k=0}^{i} f[k]g[i-k] - \sum_{k=i+1}^{n} f[k]g[i+n-k].$$

For the variance of the product XY of two independent random variables X, Y we have that $\sigma^2_{XY} = \mathbb{E}(X)^2\sigma_Y^2 + \mathbb{E}(Y)^2\sigma_X^2 + \sigma_X^2\sigma_Y^2$, where $\mathbb{E}(X)$ and $\mathbb{E}(Y)$ are the expectations of X and Y respectively, whereas for the variance of the sum $X + Y$ we have $\sigma^2_{X+Y} = \sigma_X^2 + \sigma_Y^2$. The coefficients $(fg)[i]$ of fg hence are the sum of n products of the coefficients of f and g. The claimed result follows. □

3.1 Expressions for the Critical Quantities

We next establish the critical quantities after BGV `PreMult`, `KeySwitch` and `ModSwitch`, as implemented in HElib. We consider the multiplication of two ciphertexts, where one is the output of at least one multiplication, and the other is fresh. Let $\mathtt{ct}_0 = (\mathtt{ct}_0[0], \mathtt{ct}_0[1])$ be a ciphertext, which is not fresh, encrypting m_0 at level i with critical quantity $v_0 = [< \mathtt{ct}_0, \mathtt{sk} >]_{Q_i}$. Let $\mathtt{ct}_1 = (\mathtt{ct}_1[0], \mathtt{ct}_1[1])$ be a fresh ciphertext encrypting m_1 with critical quantity $v_1 = [< \mathtt{ct}_1, \mathtt{sk} >]_{Q_L}$. Furthermore, let $(\mathtt{ct}^{pm}[0], \mathtt{ct}^{pm}[1], \mathtt{ct}^{pm}[2]) := \mathtt{PreMult}(\mathtt{ct}_0, \mathtt{ct}_1)$ denote the output of pre-multiplication, $(\mathtt{ct}^{ks}[0], \mathtt{ct}^{ks}[1]) := \mathtt{KeySwitch}(\mathtt{ct}^{pm})$ denote the output of key switching and $(\mathtt{ct}^{ms}[0], \mathtt{ct}^{ms}[1]) := \mathtt{ModSwitch}(\mathtt{ct}^{ks})$ denote the the output of modulus switching. These ciphertexts all encrypt $[m_0 m_1]_t$ with critical quantities v_{pm}, v_{ks} and v_{ms} respectively.

We first determine the BGV critical quantity v_{pm} of $(c_0^{pm}, c_1^{pm}, c_2^{pm})$.

Lemma 4. *With the notation as above, we can express* $v_{pm} = [v_0 v_1]_{Q_i}$.

Proof. For some h_1, $h_2 \in \mathbb{N}$, we have:

$$\begin{aligned}
v_{pm} &= [\mathtt{ct}^{pm}[0] + \mathtt{ct}^{pm}[1]s + \mathtt{ct}^{pm}[2]s^2]_{Q_i} \\
&= [\mathtt{ct}_0[0]\mathtt{ct}_1[0] + (\mathtt{ct}_0[0]\mathtt{ct}_1[1] + \mathtt{ct}_0[1]\mathtt{ct}_1[0])s + \mathtt{ct}_0[1]\mathtt{ct}_1[1]s^2]_{Q_i} \\
&= [(\mathtt{ct}_0[0] + \mathtt{ct}_0[1]s)(\mathtt{ct}_1[0] + \mathtt{ct}_1[1]s)]_{Q_i} \\
&= [([\mathtt{ct}_0[0] + \mathtt{ct}_0[1]s]_{Q_i} + h_1 Q_i)([\mathtt{ct}_1[0] + \mathtt{ct}_1[1]s]_{Q_i} + h_2 Q_i)]_{Q_i} = [v_0 v_1]_{Q_i}.
\end{aligned}$$

□

We next give an expression for the critical quantity v_{ks} of $\mathtt{ct}^{ks}$, specialised to the HElib implementation of BGV. Note that, by the definition of the key switching matrix as given in [22], it holds that: $a_{ij}^{(0)} + a_{ij}^{(1)}s = kD_j^{\star}s_i + te_{ij}$.

Lemma 5. *With the notation as above, we can express*

$$v_{ks} = \left[\frac{Q}{Q_i} v_{pm} + t \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] e_{2j} \right]_Q .$$

Proof. The result follows from:

$$\begin{aligned}
v_{ks} &= [< \mathtt{ct}^{ks}, \mathtt{sk} >]_Q \\
&= \left[k\mathtt{ct}^{pm}[0] + \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] a_{2,j}[0] + \left(k\mathtt{ct}^{pm}[1] + \sum_{j=1}^{\ell} \mathtt{ct}_j^{ks}[2] a_{2,j}[1] \right) s \right]_Q \\
&= \left[k(\mathtt{ct}^{pm}[0] + \mathtt{ct}^{pm}[1]s) + \sum_{j=1}^{\ell} \mathtt{ct}_j[2](kD_j^{\star} s^2 + te_{2j}) \right]_Q \\
&= \left[k(\mathtt{ct}^{pm}[0] + \mathtt{ct}^{pm}[1]s + \mathtt{ct}^{pm}[2]s^2) + t \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] e_{2j} \right]_Q .
\end{aligned}$$

□

In HElib, $k = \frac{Q}{Q_i}$ is chosen to be the product of all the special primes and such that the kv_{pm} term dominates the expression given for v_{ks} in Lemma 5. Its bit length is determined through the following heuristic

$$\log_2 \left(\frac{D_{\max} \cdot m \cdot t \cdot \sigma_0 \cdot \sqrt{12} \cdot \ell}{\sqrt{\phi(m) \ln(\phi(m))} t^2 h} \right).$$

This heuristic is taken from the method `AddSpecialPrimes()` from [23]. Here, $D_{\max} = \max_{j \in \{1,\dots,\ell\}} D_j^{\star}$ is the largest digit used in the decomposition of $\mathtt{ct}^{pm}[2]$, m is the dimension of the cyclotomic ring (if it is a power of 2, then $m = 2n$), t is the plaintext modulus, σ_0 the standard deviation of the error distribution, usually $\sigma_0 = 3.2$, and h is the hamming weight of the secret key. The parameter ℓ is normally set to be 3 by default [22]. This discussion leads to the following corollary.

Corollary 1. *The critical quantity after HElib key switching can be approximated as*

$$v_{ks} \approx \frac{Q}{Q_i} v_{pm}.$$

We next give an expression for the critical quantity v_{ms} in (c_0^{ms}, c_1^{ms}), that is specialised to the HElib implementation of BGV.

Lemma 6. *Let*

$$\tau_i := \frac{Q_{i-1}}{Q} \mathtt{ct}[i] - \left\lfloor \frac{Q_{i-1}}{Q} \mathtt{ct}[i] \right\rceil_t$$

be the rounding error associated with the critical quantity. With the remaining notation as above, we can express

$$v_{ms} = \left[\frac{Q_{i-1}}{Q} v_{ks} + \tau_0 + \tau_1 s \right]_{Q_{i-1}}.$$

Proof. The modulus switching procedure for switching from a modulus Q to a modulus Q_{i-1} scales the ciphertext by the factor $\frac{Q_i}{Q}$ and rounds it to the nearest integer, such that it is again encrypting the same message modulo t as before the modulus switching. We assume τ_i to be uniformly randomly distributed in the interval $\left(-\frac{t}{2}, \frac{t}{2}\right]$, which is in line with previous work [11, 13]. The result then follows from:

$$\begin{aligned} v_{ms} = [< \mathtt{ct}^{ms}, \mathtt{sk} >]_{Q_{i-1}} &= \left[\left\lfloor \frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[0] \right\rceil_t + \left\lfloor \frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[1] \right\rceil_t s \right]_{Q_{i-1}} \\ &= \left[\frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[0] + \tau_0 + \frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[1] s + \tau_1 s \right]_{Q_{i-1}}. \end{aligned}$$

□

3.2 Variance of the Critical Quantities

We now establish the coefficient variance of the critical quantities after BGV `PreMult`, `KeySwitch` and `ModSwitch`, as implemented in HElib. We first determine the coefficient variance of the critical quantity after key switching.

Lemma 7. *Let* $\mathtt{KeySwitch}(\mathtt{ct}^{pm}) = (\mathtt{ct}^{ks}[0], \mathtt{ct}^{ks}[1])$ *be the ciphertext after key switching and* v_{ks} *its critical quantity. Then the random variable describing* v_{ks} *has coefficient variance*

$$\sigma_{ks}^2 = \left(\frac{Q}{Q_i} \right)^2 \sigma_{pm}^2 + \frac{t^2 n \sigma_0^2}{12} \sum_{j=1}^{\ell} (D_j^\star)^2,$$

where σ_{pm}^2 *is the coefficient variance of* v_{pm}*, and* ℓ *is the number of digits.*

Proof. By Lemma 5, we have $v_{ks} = \left[\frac{Q}{Q_i} v_{pm} + t \sum_{j=1}^{\ell} c_{2,j} e_{2j} \right]_Q$. We therefore get for the coefficient variance

$$\sigma_{ks}^2 = \sigma^2_{\frac{Q}{Q_i} v_{pm}[i]} + \sigma^2_{t \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] e_{2j}} = \left(\frac{Q}{Q_i} \right)^2 \sigma^2_{v_{pm}[i]} + t^2 \sum_{j=1}^{\ell} n \sigma^2_{ct_j^{pm}[2]} \sigma^2_{e_{2j}}$$

from which the results follows. □

We next introduce the main result of this section, the coefficient variance of the critical quantity after modulus switching in HElib. Our key observation is that, in this setting, the coefficient variance of the critical quantity after `ModSwitch` is solely dependent on h and t, and not on the input critical quantities of the ciphertexts that are being multiplied. Hence, it is not dependent on the number of multiplications that were carried out previously on each respective ciphertext.

Lemma 8. *In HElib, if $||v_{pm}|| \ll \frac{Q_{i-1}}{Q_i}$, the critical quantity after modulus switching from a modulus Q to a modulus Q_{i-1} for a ciphertext $\mathtt{ct}^{ms}$ encrypting a product m can be closely approximated by the term*

$$v_{ms} = [\tau_0 + \tau_1 s]_{Q_{i-1}}.$$

The variance of the distribution of the coefficients of v_{ms} can be closely approximated by $\sigma_{ms}^2 \approx \frac{t^2}{12}(1+h)$, where h is the hamming weight of the secret key.

Proof. Let $\mathtt{ct}^{ks}$ be the ciphertext and $v_{ks} = [< \mathtt{ct}^{ks}, \mathtt{sk} >]_Q$ the critical quantity of the ciphertext after key switching. By Lemma 6 we have for the critical quantity after modulus switching:

$$v_{ms} = \left[\frac{Q_{i-1}}{Q} v_{ks} + \tau_0 + \tau_1 s \right]_{Q_{i-1}}.$$

Using Lemma 5 we obtain:

$$\begin{aligned} v_{ms} &= \left[\frac{Q_{i-1}}{Q} \left[\frac{Q}{Q_i} v_{pm} + t \sum_{j=1}^{\ell} e_{2,j} \mathtt{ct}_j^{pm}[2] \right]_Q + \tau_0 + \tau_1 s \right]_{Q_{i-1}} \\ &= \left[\frac{Q_{i-1}}{Q_i} v_{pm} + \frac{Q_{i-1}}{Q} t \sum_{j=1}^{\ell} e_{2j} \mathtt{ct}_j^{pm}[2] + \tau_0 + \tau_1 s \right]_{Q_{i-1}} \\ &\approx \left[\frac{Q_{i-1}}{Q} t \sum_{j=1}^{\ell} e_{2j} \mathtt{ct}_j^{pm}[2] + \tau_0 + \tau_1 s \right]_{Q_{i-1}}, \end{aligned}$$

where the last line holds due to the assumption that $||v_{pm}|| \ll \frac{Q_i}{Q_{i-1}}$. We see in [22] that $\log_2\left(\frac{Q_i}{Q_{i-1}}\right) \geq 36$ for all i, and hence the first part of the sum is negligible. We further see in Section 4 that $\log_2(||v_{pm}||_\infty) \leq 22$, for $n \leq 2^{15}$, so this assumption is reasonable. Next, by Corollary 1, $\frac{Q}{Q_i}$ is chosen such that $\frac{Q}{Q_i} v_{pm}$ dominates $t \sum_{j=1}^{\ell} e_{2j} \mathtt{ct}_j^{pm}[2]$. That is, $[\frac{Q}{Q_i} ||v_{pm}|| \geq \left|\left| t \sum_{j=1}^{\ell} e_{2j} \mathtt{ct}_j^{pm}[2] \right|\right|$. Thus,

$$\frac{Q_{i-1}}{Q} \left|\left| t \sum_{j=1}^{\ell} e_{2j} \mathtt{ct}_j^{pm}[2] \right|\right| \leq \frac{Q_{i-1}}{Q} ||v_{pm}|| \leq \frac{Q_{i-1}}{Q} \frac{Q_i}{Q_{i-1}} = \frac{Q_i}{Q},$$

and so this term is also negligible. We obtain the claimed approximation for v_{ms}.

Since the coefficients of τ_j for $j \in \{0,1\}$ are distributed continuously uniformly randomly in the interval $\left(-\frac{t}{2}, \frac{t}{2}\right]$, they have expectation 0 and variance $\sigma^2_{\tau_j[i]} = \frac{t^2}{12}$, for $i \in \{1, \ldots, n\}$. Using Lemma 3, and the variance of the HElib secret distribution established in Section 2.5, we obtain the following for the variance of the coefficients of $\tau_0 + \tau_1 s$:

$$\sigma^2_{ms} = \sigma^2_{(\tau_0+\tau_1 s)[i]} = \sigma^2_{\tau_0[i]} + \sigma^2_{\tau_1 s[i]} = \sigma^2_{\tau_0[i]} + n\sigma^2_{\tau_1[i]}\sigma^2_{s[i]} = \frac{t^2}{12} + n\frac{t^2}{12}\frac{h}{n},$$

from which the claimed result follows. □

We can specialize Lemma 8 to the situation of our experiments.

Corollary 2. *The coefficient standard deviation σ_{ms} of the critical quantity v_{ms} after modulus switching as implemented in HElib, with dense secret key and plaintext modulus $t = 3$, is given by*

$$\sigma_{ms} = \frac{1}{2}\sqrt{3 + \frac{3}{2}n}.$$

We now determine the coefficient variance of the critical quantity after `PreMult` in HElib, when considering the multiplication of two ciphertexts, at least one of which is not fresh.

Lemma 9. *Let $\mathtt{ct}_0$ be a ciphertext after modulus switching to level $0 \le i < L$. Let $\mathtt{ct}_1$ be a ciphertext at level $i < j \le L$. In HElib, the coefficients of the critical quantity v_{pm} of the ciphertext $\mathtt{ct}^{pm} = \mathtt{PreMult}(\mathtt{ct}_0, \mathtt{ct}_1)$ have variance*

$$\sigma^2_{pm} = \frac{t^4 n}{72}(1+h)^2.$$

Proof. Since the ciphertexts $\mathtt{ct}_0$ and $\mathtt{ct}_1$ are at different levels, a common ciphertext modulus is calculated as follows in HElib [22].

Let v_0 and v_1 be the critical quantities and Q_i and Q_j the ciphertext moduli of $\mathtt{ct}_0$ and $\mathtt{ct}_1$ respectively. The new common ciphertext modulus $\overline{Q}$ is chosen such that:

$$\frac{\overline{Q}}{Q_i}v_0 \approx v_{ms} \approx \frac{\overline{Q}}{Q_j}v_1, \tag{1}$$

where v_{ms} is the critical quantity after modulus switching $\mathtt{ct}_0$ and $\mathtt{ct}_1$ to $\overline{Q}$. Since $\mathtt{ct}_1$ has been modulus switched to level j, and the critical quantity after modulus switching is independent of the message, we have $v_1 = v_{ms}$. Hence by Equation 1 we have $\overline{Q} = Q_j$. Let $\overline{v_0}$ be the critical quantity after modulus switching $\mathtt{ct}_0$ to Q_j. Then we have:

$$\overline{v_0} = \left[\left\lfloor\frac{Q_j}{Q_i}\mathtt{ct}_0[0]\right\rceil_t + \left\lfloor\frac{Q_j}{Q_i}\mathtt{ct}_0[1]\right\rceil_t s\right]_{Q_j} = \left[\frac{Q_j}{Q_i}(\mathtt{ct}_0[0] + \mathtt{ct}_0[1]s) + \tau_0 + \tau_1 s\right]_{Q_j}$$

$$= \left[\frac{Q_j}{Q_i} v_0 + v_{ms}\right]_{Q_j} \approx [v_{ms} + v_{ms}]_{Q_j},$$

where the last approximation holds by Equation 1. Using Lemma 4 and Lemma 8, we obtain the claimed variance as follows:

$$\sigma_{pm}^2 = n(\sigma_{ms}^2 + \sigma_{ms}^2)\sigma_{ms}^2 = 2n\sigma_{ms}^4 = 2n\left(\frac{t^2}{12}(1+h)\right)^2 = \frac{t^4 n}{72}(1+h)^2.$$

□

We can specialize Lemma 9 to the situation of our experiments.

Corollary 3. *The coefficient standard deviation σ_{pm} of the critical quantity v_{pm} after* `PreMult` *as implemented in HElib, with dense secret key and plaintext modulus $t = 3$, is given by*

$$\sigma_{pm} = \frac{3}{2}\left(1 + \frac{n}{2}\right)\sqrt{\frac{n}{2}}.$$

4 Experimental Verification

In this section, we confirm the theoretical results that we obtained in Section 3 experimentally. We compare the predicted standard deviation of the critical quantity after HElib operations with the point estimator of the observed standard deviation of the critical quantity of HElib ciphertexts, over a data set of 10000 trials.

In more detail, we evaluated several circuits for various parameter sets in HElib v. 2.2.1 [23]. We evaluated each circuit 10000 times for each parameter set. We considered circuits with γ multiplications, for $1 \leq \gamma \leq 5$ as follows. For one multiplication, we multiplied two fresh ciphertexts, applied key switching to the result and modulus switched to the next level. For two multiplications, we multiplied two fresh ciphertexts, applied key switching to the result, and modulus switched to the next level. We then multiplied the resulting ciphertext with a fresh one, applied key switching and modulus switching. For three, four and five multiplications, we follow the same methodology, so that at each multiplication, we multiply a fresh ciphertext with the output of the previous multiplication.

We recorded the critical quantities of the ciphertext at each stage in the last multiplication in each circuit. That is, in the case of one multiplication, they were calculated directly after the first pre-multiplication, key switching and modulus switching. In the case of two multiplications, they were calculated after the second pre-multiplication, key switching and modulus switching; and so on.

The parameter sets we used are given in abbreviated form in the Tables 1 - 3. The full parameter sets can be found in Appendix A of the eprint version [14], giving the bit length of the moduli in the moduli chain, which is necessary for calculating the key switching heuristics; and estimates of the security (based on the lattice estimator [2]). Our goal was to choose several parameter sets, each with a security level of 128 bits or above. To be able to compare among multiple sets of parameters for a fixed multiplicative depth, some insecure parameter sets were included, if no secure ones could be found. For the parameter sets with $n = 16384$ and $n = 32768$, the same bit length for the moduli chain was set, but δ was varied to observe the effects of the resolution of the moduli chain on the critical quantity.

The experimental results observed for `PreMult` `KeySwitch` and `ModSwitch` can be seen in Tables 1 to 3 respectively. In the tables, the column **Heuristic** gives the theoretically obtained standard deviations for `PreMult` (Corollary 3) `KeySwitch` (Corollary 7) and `ModSwitch` (Corollary 2), and the column $\sigma_{\mathrm{est},op}$ for $op \in \{pm, ks, ms\}$ gives the experimentally obtained sample standard deviation. The column $\Delta_i := \frac{|\sigma_{op} - \sigma_{\mathrm{est},op}|}{\sigma_{op}} \cdot 100$ for $i \in \{1, \ldots, 5\}$ gives the observed difference between theory and practice for each circuit as a percentage. The first line in each table gives the number of multiplications that were evaluated. The results for one pre-multiplication are not presented, since in this case the conditions of Lemma 9 are not satisfied, and hence the theoretical results are not applicable. Indeed, the theoretical results assume that both input ciphertexts have been freshly modulus switched. This is correct from the second multiplication on: one ciphertext is the result of a previous multiplication and therefore was modulus switched just before. The second ciphertext is a fresh encryption and therefore at a higher level as the first. To make levels match this ciphertext is modulus switched, too. The only exception to this is the first multiplication, where to fresh ciphertexts with therefore different initial critical quantities are multiplied. Since the a multiplication is normally followed by a modulus switching and the exact noise estimates of the first multiplication are therefore no very important, we did not include this special case here.

For `PreMult` we see from Table 1 that the experimental results deviate from the theoretical ones by at most 2.1%, and for all but six values the deviation is less than 1%. For`ModSwitch` we see from Table 3 that the experimental results deviate by at most 1.1% and for all but two values the deviation is less than 1%. The standard error tells us to expect a deviation of the experimental from the theoretical results of approximately $\frac{1}{\sqrt{n}}$, where n is the number of trials. Since we have $n = 10000$ for all experiments, this means we are to expect a deviation of about $\frac{1}{\sqrt{10000}} = 1\%$. That is, the deviations of the experimental results from the theoretical ones are what is to be empirically expected. We can hence consider our theoretical results to be experimentally confirmed for pre-multiplication and modulus switching. Further, we conclude that our results are near-optimal.

The experimental results observed for `KeySwitch` can be seen in Table 2. For `KeySwitch` the deviations that we observe are larger, between 0.14% and 16.88%. This can be explained by the fact that we need approximations to obtain a calculable heuristic, for example estimating $D_j^\star$ as the maximal value among all $j \in \{1, \ldots, \ell\}$.

Our experiments consider circuits with up to five multiplications. The results confirm Lemma 8, which shows that the noise after modulus switching is independent of the number of multiplications computed previously. The same result would also apply in a deeper circuit, if a modulus switching were applied after each multiplication. Therefore, experimental results for circuits with more multiplications have not been included since they do not provide new information.

(n, L, δ)	**Heuristic**	**2**		**3**		**4**		**5**	
		$\sigma_{\text{est},pm}$	Δ_2	$\sigma_{\text{est},pm}$	Δ_3	$\sigma_{\text{est},pm}$	Δ_4	$\sigma_{\text{est},pm}$	Δ_5
$(4096, 2, 6)$	17.085	17.095	0.60%	-	-	-	-	-	-
$(8192, 3, 6)$	18.585	18.599	0.96%	18.596	0.77%	-	-	-	-
$(8192, 4, 10)$		18.590	0.35%	18.575	0.70%	18.584	0.12%	-	-
$(16384, 5, 3)$	20.085	20.095	0.66%	20.087	1.35%	20.082	0.12%	20.104	1.33%
$(16384, 5, 6)$		20.054	2.17%	20.101	1.09%	20.071	1.01%	20.105	1.42%
$(32768, 7, 3)$	21.585	21.580	0.37%	21.574	0.77%	21.591	0.40%	21.576	0.66%
$(32768, 7, 6)$		21.576	0.62%	21.590	0.37%	21.592	0.50%	21.586	0.89%

Table 1: Estimated and theoretical standard deviations of the critical quantity after pre-multiplication in bits.

(n, L, δ)	**Heuristic**	**2**		**3**		**4**		**5**	
		$\sigma_{\text{est},ks}$	Δ_2	$\sigma_{\text{est},ks}$	Δ_3	$\sigma_{\text{est},ks}$	Δ_4	$\sigma_{\text{est},ks}$	Δ_5
$(4096, 2, 6)$	62.924	63.13	15.44%	-	-	-	-	-	-
$(8192, 3, 6)$	63.465	63.69	16.88%	63.61	10.92%	-	-	-	-
$(8192, 4, 10)$	66.492	66.549	3.99%	66.540	3.33%	66.520	1.94%	-	-
$(16384, 5, 3)$	121.964	122.076	8.08%	122.081	8.47%	122.044	5.67%	122.013	3.45%
$(16384, 5, 6)$	67.065	67.145	5.67%	67.117	3.65%	67.113	3.38%	67.091	1.84%
$(32768, 7, 3)$	183.388	183.398	0.69%	183.392	0.24%	183.390	0.14%	183.401	0.88%
$(32768, 7, 6)$	125.387	125.445	4.07%	125.449	4.36%	125.443	3.93%	125.425	2.67%

Table 2: Theoretical and experimental standard deviation of the critical quantity after key switching in bits.

5 Comparison with Other Noise Heuristics

In this section, to illustrate the effectiveness of our HElib-specific approach, we compare our noise analysis with the prior heuristic noise analyses of BGV given in [13], [22] and [25]. In particular, these prior works all give bounds on the canonical norm of either the BGV critical quantity ([13, 22]) or the infinity norm of the BGV noise ([25]). In order to compare our results with these works,

(n, L, δ)		**1**		**2**		**3**		**4**		**5**	
	Heur.	$\sigma_{\text{est},ms}$	Δ_1	$\sigma_{\text{est},ms}$	Δ_2	$\sigma_{\text{est},ms}$	Δ_3	$\sigma_{\text{est},ms}$	Δ_4	$\sigma_{\text{est},ms}$	Δ_5
$(2048, 1, 3)$	4.793	4.779	0.97%	-	-	-	-	-	-	-	-
$(4096, 1, 3)$	5.293	5.277	1.12%	-	-	-	-	-	-	-	-
$(4096, 2, 6)$		5.298	0.36%	5.294	0.07%	-	-	-	-	-	-
$(8192, 1, 3)$	5.793	5.806	0.94%	-	-	-	-	-	-	-	-
$(8192, 3, 6)$		5.796	0.24%	5.797	0.31%	5.800	0.55%	-	-	-	-
$(8192, 4, 10)$		5.780	0.87%	5.799	0.47%	5.793	0.02%	5.791	0.13%	-	-
$(16384, 5, 3)$	6.293	6.294	0.11%	6.294	0.13%	6.295	0.14%	6.293	0.02%	6.299	0.47%
$(16384, 5, 6)$		6.300	0.53%	6.280	0.87%	6.301	0.55%	6.295	0.16%	6.299	0.43%
$(32768, 7, 3)$	6.793	6.790	0.19%	6.794	0.09%	6.794	0.13%	6.791	0.14%	6.789	0.23%
$(32768, 7, 6)$		6.782	0.70%	6.793	0.05%	6.792	0.03%	6.793	0.05%	6.793	0.12%

Table 3: Theoretical and experimental standard deviation of the critical quantity after modulus switching in bits.

we therefore also need to derive appropriate bounds on the critical quantity and noise in HElib BGV ciphertexts from the results obtained in Section 3.

We will give the comparison with related work for a circuit consisting of two multiplications. This is done because the first multiplication is a special case, for which Lemma 9 does not apply. If we multiply two ciphertexts which are not at the same level, `ModSwitch` is first applied to the ciphertext at the highest level, in order for both ciphertexts to be at the same level. This means that from the second multiplication onwards, the noise in the input ciphertexts is always the noise resulting from `ModSwitch`. Only in the first multiplication are the input ciphertexts fresh ciphertexts, which leads to a different expression for the standard deviation of the critical quantity after pre-multiplication.

5.1 Bounding the Critical Quantity

We use Iliashenko's approach [24], recalled in Lemma 1, to give a bound on the canonical norm of the critical quantity. To bound the infinity norm of the critical quantity, for pre-multiplication and modulus switching, we show the critical quantity is distributed as a Normal random variable, and use Lemma 2. For key switching, applying the Kolmogorov-Smirnov test [26, 34] to our experimental data indicated that the critical quantity was not Normal (see Appendix B of the eprint version [14]. We obtain a bound on the infinity norm of the critical quantity after key switching using bounds on the infinity norms of the constituent polynomials that make up the critical quantity expression. In particular, since we do not use the standard deviation of the coefficients of the critical quantity after key switching to bound the critical quantity, it does not matter that the theoretical results for the standard deviation as shown in Table 2 are less tight.

In Lemma 10 we show that the distribution of the critical quantity after pre-multiplication and modulus switching can be approximated by a Normal distribution. Similar results were given in [30] for the distribution of the noise after these operations.

Lemma 10. *Let* $\mathtt{ct}^{pm}$ *and* $\mathtt{ct}^{ms}$ *be the ciphertexts after pre-multiplication and modulus switching respectively. Let* $v_{pm} = [\mathtt{ct}^{pm}[0] + \mathtt{ct}^{pm}[1]s + \mathtt{ct}^{pm}[2]s^2]_q$ *and* $v_{ms} = [\mathtt{ct}^{ms}[0] + \mathtt{ct}^{ms}[1]s]_q$ *be their respective critical quantities. Then*

$$v_{pm}[i] \sim \mathcal{N}(0, \sigma_{pm}^2)$$
$$v_{ms}[i] \sim \mathcal{N}(0, \sigma_{ms}^2),$$

for all i*, where* σ_{pm}^2 *and* σ_{ms}^2 *are the coefficient variances given in Lemmas 8 and 9 respectively.*

Proof. Deferred to Appendix C of the eprint version [14]. □

It remains to bound the critical quantity after key switching.

Lemma 11. *The critical quantity after key switching in HElib can be bounded as*

$$||v_{ks}||_\infty \leq 10k\sigma_{pm} + 5t\ell n D_{\max}\sigma_0\,,$$

where $D_{\max} = \max_{j=1,\ldots,\ell} D_j^\star$*, the maximal digit in the decomposition of* $\mathtt{ct}[2]$*.*

Proof. Using the expression for v_{ks} given in Lemma 5, we can bound

$$\begin{aligned} ||v_{ks}||_\infty &= \Big|\Big|\frac{Q}{Q_i}v_{pm} + t\sum_{j=1}^{\ell}\mathtt{ct}_j^{pm}[2]e_{2j}\Big|\Big|_\infty \leq \frac{Q}{Q_i}||v_{pm}||_\infty + t\sum_{j=1}^{\ell} n||\mathtt{ct}_j^{pm}[2]||_\infty ||e_{2j}||_\infty \\ &\leq \frac{Q}{Q_i}10\sigma_{pm} + t\ell n\frac{D_{\max}}{2}10\sigma_0 = k\sigma_{pm} + 5t\ell n D_{\max}\sigma_0, \end{aligned}$$

where for bounds on $||e_{2,j}||_\infty$ and $||v_{pm}||_\infty$, the normality of their distributions, and hence Lemma 2, was used. □

5.2 Bounding the Noise

While our work focuses on the critical quantity, the work [25] uses the noise as in Definition 2. To facilitate comparison, we adapt our heuristics as follows.

Lemma 12. *Let* $\mathtt{ct}^{pm}$*,* $\mathtt{ct}^{ks}$ *and* $\mathtt{ct}^{ms}$ *be the ciphertexts after pre-multiplication, key switching and modulus switching. Let* e_{op} *be their noises, for* $op \in \{pm, ks, ms\}$*. Then we have for the variances* $\sigma_{pm,e}^2, \sigma_{ks,e}^2, \sigma_{ms,e}^2$ *of the noise:*

$$\sigma_{pm,e}^2 = \frac{n}{144}(2t^2(1+h)^2 + 17t + 26)$$
$$\sigma_{ms,e}^2 = \frac{1}{12}(2+h).$$
$$\sigma_{ks,e}^2 = \left(\frac{Q}{Q_i}\right)^2 \sigma_{pm,e}^2 + \frac{n\sigma_0^2}{12}\sum_{j=1}^{\ell}(D_j^\star)^2.$$

Proof. Deferred to Appendix D of the eprint version [14]. □

It is shown in [30] that for pre-multiplication and modulus switching, the noise is distributed as a Normal random variable. We can then use Lemma 2 to give a bound on the infinity norm. It remains to bound the noise after key switching.

Lemma 13. *The noise after key switching in HElib can be bounded as*

$$||e_{ks}||_\infty \leq \frac{Q}{Q_i} 10\sigma_{pm,e} + 5\ell n D_{\max}\sigma_0 \,.$$

Proof. Appendix D of the eprint version [14] shows that $e_{ks} = \frac{Q}{Q_i}e_{pm} + \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2]e_{2j}$. Hence

$$||e_{ks}||_\infty = \Big|\Big|\frac{Q}{Q_i}e_{pm} + \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2]e_{2j}\Big|\Big|_\infty \leq \frac{Q}{Q_i}||e_{pm}||_\infty + \sum_{j=1}^{\ell} n||\mathtt{ct}_j^{pm}[2]||_\infty ||e_{2j}||_\infty,$$

from which the claim follows. □

5.3 Comparison of Critical Quantity Bounds with [13] and [22]

The canonical norm bounds stated in [13] and [22] are recalled in Appendix E of the eprint version [14]. We present in Table 4 (for pre-multiplication and modulus switching) and in Table 5 (for key switching) the results of comparing the bounds in [13] and [22] with our bounds in the infinity and canonical norms developed in Section 5.1. We compare with the experimentally obtained infinity norms after two pre-multiplications, key switches and modulus switches (columns $||\cdot||_\infty$). Note that since the noise after modulus switching does not depend on the input noise, the infinity norm is not dependent on the number of multiplications (see Table 13 in Appendix G.2 in the eprint version [14]).

Tables 4 and 5 show that both our bounds on the infinity norm and on the canonical norm are tighter than the ones given in the two works we compare with. We also note that the key switching bound from [22] seems to underestimate the key switching noise by about 3 bits. This could lead to decryption errors.

5.4 Comparison of Noise Bounds with [25]

We next compare our noise bounds, developed in Section 5.2, with the noise bounds presented in [25]. We present results only for pre-multiplication and modulus switching. We do not compare with the key switching bounds in [25] since they modulus switch from the special modulus to the ciphertext modulus directly after key switching. This reduces the noise significantly and makes it even smaller than the pre-multiplication noise [25]. This is not the case in the HElib implementation, so the comparison would not be very meaningful.

(n, L, δ)	PreMult					ModSwitch				
	$\|\|\cdot\|\|_\infty$	B_∞	B_{can}	[13]	[22]	$\|\|\cdot\|\|_\infty$	B_∞	B_{can}	[13]	[22]
$(4096, 2, 6)$	18.94	20.41	25.67	28.17	44.42	7.15	8.61	13.88	14.09	22.21
$(8192, 3, 6)$	20.52	21.91	27.67	30.17	47.53	7.72	9.11	14.88	15.08	23.76
$(8192, 4, 6)$	20.51					7.73				
$(16384, 5, 3)$	22.08	23.41	29.67	32.17	50.63	8.28	9.61	15.88	16.09	25.31
$(16384, 5, 6)$	22.03					8.29				
$(32768, 7, 3)$	23.07	24.91	31.67	34.17	53.73	8.89	10.11	16.88	17.09	26.86
$(32768, 7, 6)$	23.68					8.89				

Table 4: Comparison of the infinity norm of the experimental results with our theoretical bounds on the infinity norm B_∞ and the canonical norm B_{can} of the critical quantity, with the results from [13] and [22].

(n, L, δ)	$\|\|\cdot\|\|_\infty$	B_∞	B_{can}	[13]	[22]
$(4096, 2, 6)$	65.078	65.407	70.671	71.848	62.435
$(8192, 3, 6)$	65.687	66.907	72.670	73.848	63.493
$(8192, 4, 10)$	68.526	69.907	76.670	76.848	66.493
$(16384, 5, 3)$	124.115	125.407	131.670	131.848	121.546
$(16384, 5, 6)$	69.174	70.407	76.670	76.848	66.546
$(32768, 7, 3)$	185.204	186.907	193.670	193.848	182.596
$(32768, 7, 6)$	127.539	128.907	135.67	135.848	124.596

Table 5: Comparison of the experimentally obtained bound on the infinity norm of the critical quantity after key switching with theoretical bounds on the infinity norm and the canonical norm with [13] and [22]. The values are given in bits.

The noise bounds stated in [25] are recalled in Appendix F of the eprint version [14]. Table 6 gives the results of comparing the bounds in [25] with our bounds in the infinity and canonical norms developed in Section 5.2. The columns $||\cdot||_\infty$ contain the infinity norm after the second pre-multiplication and modulus switching respectively, while results for all multiplications are given in Table 15 in Appendix G.3 of the eprint version [14].

Table 6 shows that our bounds for pre-multiplication are tighter than the ones given by [25]. For modulus switching, the results of [25] are closer to the experimentally obtained values, but are underestimating them. Since their results were developed considering PALISADE [31], the difference may be due to differences in the implementation in these two libraries. The estimation of the ring expansion factor as $\gamma_{\mathcal{R}} \approx 2\sqrt{n}$ may also underestimate the noise polynomial in certain cases.

In summary, our comparisons demonstrate that relying on prior BGV noise analyses to estimate the noise growth in BGV HElib ciphertexts might lead to

(n, L, δ)	PreMult				ModSwitch			
	$\|\| \cdot \|\|_\infty$	B_∞	$B_{\mathbf{can}}$	**[25]**	$\|\| \cdot \|\|_\infty$	B_∞	$B_{\mathbf{can}}$	**[25]**
$(4096, 2, 6)$	17.99	18.82	24.09	15.58	6.22	7.03	12.95	6.01
$(8192, 3, 6)$	19.56	20.32	26.09	16.58	6.77	7.53	13.95	6.51
$(8192, 4, 10)$	19.59				6.80			
$(16384, 5, 3)$	21.13	21.82	28.09	17.58	7.35	8.03	14.95	7.01
$(16384, 5, 6)$	21.16				7.34			
$(32768, 7, 3)$	22.68	23.32	30.09	18.58	7.90	8.53	15.95	7.50
$(32768, 7, 6)$	22.69				7.90			

Table 6: Comparison of the bounds on the infinity norm of the noise after 2 multiplications for pre-multiplications and modulus switching with the results from [25] in bits.

decryption errors. This further emphasises the value of implementation specific noise analyses, as we have presented here for HElib.

6 Optimizations and Tradeoffs

In this section, we show how our analysis can be applied to give an optimized ratio between ciphertext moduli in the moduli chain, and discuss the improvements that this could enable.

The moduli chain in HElib is constructed from three chosen sets of primes: small primes, normal primes and special primes [22]. The ciphertext moduli are formed as products of elements from special primes and normal primes. The product of all the special primes forms the factor k, by which the current ciphertext modulus is multiplied to obtain the modulus for key switching. In contrast to the construction of ciphertext primes, the factor k always consists of all the special primes.

Let δ be the resolution parameter. The default setting is $\delta = 3$, but it can be customized to $\delta \in \{1, \ldots, 10\}$. The normal primes are all of the same bit size b, where $b \in \{54, \ldots, 60\}$. The small primes consist of two primes of bit size $c = \left\lfloor \frac{2b}{3} \right\rceil \in \{36, \ldots, 40\}$ and one prime of size $d = b - \delta 2^t > c$, where $t = 0, 1, \ldots$ can be chosen as needed. Therefore, the ratio $\frac{Q_i}{Q_{i-1}}$ between the ciphertext moduli of two adjacent levels is always at least 36 bits, but is more likely bigger. The smallest ratio of $\frac{Q_i}{Q_{i-1}}$ that was observed in our experiments for different values of δ was 54 bits, where we obtained this ratio by calling `context.productOfPrimes(context.getCtxtPrimes())` after each modulus switching and divided the results. Our experiments used $\delta \in \{3, 6, 10\}$. In these cases, $d \in \{42, \ldots, 57\}$ for $\delta = 3$, $d \in \{42, \ldots, 54\}$ for $\delta = 6$ and $d \in \{44, \ldots, 50\}$ for $\delta = 10$.

The special primes are chosen such that $k||v_{pm}||^{\mathrm{can}} \geq \left|\left|t \sum\limits_{j=1}^{\ell} \mathtt{ct}_j[2]e_{2,j}\right|\right|^{\mathrm{can}}$, in order to keep the modulus switching noise as small as possible. However, as can be seen from Section 3, this condition is sufficient but not necessary. To achieve a constant modulus switching noise, we require

$$\left[\left\lceil \frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[0] \right\rfloor + \left\lceil \frac{Q_{i-1}}{Q} \mathtt{ct}^{ks}[1] \right\rfloor s \right]_{Q_{i-1}} \approx [\tau_0 + \tau_1 s]_{Q_{i-1}}. \tag{2}$$

In the proof of Lemma 8 we have seen that

$$||\frac{Q_{i-1}}{Q} v_{ks}||_\infty \approx ||\frac{Q_{i-1}}{Q} t \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] e_{2j}||_\infty = ||\frac{Q_{i-1}}{Q_i k} t \sum_{j=1}^{\ell} \mathtt{ct}_j^{pm}[2] e_{2j}||_\infty. \tag{3}$$

To fulfill the conditions of Equation 2, this term needs to be smaller than the modulus switching noise. This can be achieved by either making $\frac{Q_i}{Q_{i-1}}$ or k sufficiently large. We will look at both those values, assuming them in turn to be fixed. From Lemma 11 we have

$$||\frac{Q_{i-1}}{Q_i k} t \sum \mathtt{ct}_j^{pm}[2] e_{2j}||_\infty \leq \frac{Q_{i-1}}{Q_i k} t \ell n D_{\max} 5 \sigma_0, \tag{4}$$

where $D_{\max} = \max_{j \in \{1,\dots,\ell\}}(D_j^\star)$ is the maximal digit that is used for decomposition during key switching. As stated in Lemma 2, we have

$$\alpha \sigma_{ms} \leq ||\tau_0 + \tau_1 s||_\infty, \tag{5}$$

with probability $\alpha = 1 - \mathrm{erf}\left(\frac{\beta}{\sqrt{2}}\right)$. Depending on β, we therefore obtain for k by combining Equations 3,4 and 5

$$\frac{Q_{i-1} D_{\max} t \ell n 5 \sigma_0}{Q_i \sigma_{ms}} \leq k. \tag{6}$$

The values we observed for $D_{\max}$ in our experiments can be found in Table 12 in Appendix G.1 of the eprint version [14]. We calculate the values for k needed for our parameter sets based Equation 6 for two values of $\frac{Q_i}{Q_{i-1}}$: 36 bits, since this is the minimal value possible in HElib; and 54 bits, since this was the most common value we observed in practice. The values for k shown in Table 7 are for $\alpha \in \{0.01, 0.001, 0.0001\}$.

We see that we can optimize k for $\alpha = 0.01$ by up to 8 bits if $\log_2\left(\frac{Q_i}{Q_{i-1}}\right) = 36$ but can reach an optimization of up to 26 bits if $\log_2\left(\frac{Q_i}{Q_{i-1}}\right) = 54$.

If we assume k to be constant, then we get from Equation 2

$$\frac{Q_i}{Q_{i-1}} > \frac{D_{\max} t \ell n 5 \sigma_0}{\beta \sigma_{ms} k}.$$

	$\log_2\left(\frac{Q_i}{Q_{i-1}}\right) = 36$			$\log_2\left(\frac{Q_i}{Q_{i-1}}\right) = 54$		
(n, L, δ)	$\alpha = 0.01$	$\alpha = 0.001$	$\alpha = 0.0001$	$\alpha = 0.01$	$\alpha = 0.001$	$\alpha = 0.0001$
$(2048, 1, 3)$	37	41	44	19	22	25
$(4096, 1, 3)$	39	42	45	21	24	27
$(4096, 2, 6)$	39	42	45	21	24	27
$(8192, 1, 3)$	40	43	47	22	25	28
$(8192, 3, 6)$	40	43	47	22	25	28
$(8192, 4, 10)$	43	46	50	25	28	31
$(16384, 5, 3)$	98	101	104	80	83	86
$(16384, 5, 6)$	43	46	49	25	28	31
$(32768, 7, 3)$	166	163	166	141	144	147
$(32768, 7, 6)$	101	105	108	83	86	89

Table 7: Optimized values for k in bits for different failure probabilities α and ciphertext ratios.

The result for the ratio $\frac{Q_i}{Q_{i-1}}$ can be found in Table 8, where we assumed as values for k the values observed in our experiments, as specified in Table 9 in Appendix A of the eprint version [14].

(n, L, δ)	$\alpha = 0.01$	$\alpha = 0.001$	$\alpha = 0.0001$
$(2048, 1, 3)$	29	32	35
$(4096, 1, 3)$	30	33	36
$(4096, 2, 6)$	30	33	36
$(8192, 1, 3)$	32	35	38
$(8192, 3, 6)$	32	35	38
$(8192, 4, 10)$	32	35	38
$(16384, 5, 3)$	33	36	39
$(16384, 5, 6)$	33	36	39
$(32768, 7, 3)$	34	37	40
$(32768, 7, 6)$	34	37	40

Table 8: Ratio between ciphertext moduli in bits for different failure probabilities α.

We see from Table 8 that we can reduce the ratio between ciphertext moduli by a minimum of 2 bits, if the ratio was never bigger than the smallest prime in "small prime". We can reduce the ratio by up to 25 bits compared to the ratios we practically observed in our experiments.

The optimization we propose leads to a trade-off: we can either reduce the size of the special modulus during key switching, or the ratio between ciphertext moduli and hence reach a larger multiplicative depth for the same parameter sets. These two optimizations may be of interest in different applications.

For example, in a non-interactive protocol, bootstrapping represents a bottleneck. In this case, we would like to maximize the number of multiplications before having to bootstrap. Therefore, optimizing the ratio between the ciphertext moduli and thus reaching a larger multiplicative depth for the same parameter set optimizes a circuit. In the somewhat homomorphic encryption setting, increasing the number of ciphertext moduli for a fixed parameter set may permit to perform a higher-depth computation with a smaller parameter set, thus improving performance.

On the other hand, in a client-aided outsourced computation protocol, bootstrapping is replaced by sending the ciphertext to the client for recryption., and is no longer a bottleneck. However, in this scenario, evaluation keys for key switching will have to be generated and exchanged, whose size grows with the size of the special moduli. In such a case, to save on communication costs and to make the key switching procedure more efficient, reducing the size of the special modulus can be of importance. Since in this case the multiplicative depth is less important, the ratio between the ciphertext moduli can be increased, hence allowing for a substantial reduction of the factor k.

Acknowledgements

We would like to thank Leroy Odunlami for insightful discussions on statistics and probability theory.

References

[1] Adi Akavia, Dan Feldman, and Hayim Shaul. Secure search on encrypted data via multi-ring sketch. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 985–1001. ACM Press, October 2018.

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 377–408. Springer, Heidelberg, December 2017.

[8] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: fast fully homomorphic encryption over the torus. *J. Cryptol.*, 33(1):34–91, 2020.

[9] Ilaria Chillotti, Damien Ligier, Jean-Baptiste Orfila, and Samuel Tap. Improved programmable bootstrapping with larger precision and efficient arithmetic circuits for TFHE. LNCS, pages 670–699. Springer, Heidelberg, 2021.

[10] Carlos Cid, John Petter Indrøy, and Håvard Raddum. FASTA - A stream cipher for fast FHE evaluation. In Steven D. Galbraith, editor, *Topics in Cryptology - CT-RSA 2022 - Cryptographers' Track at the RSA Conference 2022, Virtual Event, March 1-2, 2022, Proceedings*, volume 13161 of *Lecture Notes in Computer Science*, pages 451–483. Springer, 2022.

[11] 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.

[12] Anamaria Costache, Benjamin R. Curtis, Erin Hales, Sean Murphy, Tabitha Ogilvie, and Rachel Player. On the precision loss in approximate homomorphic encryption. Cryptology ePrint Archive, Paper 2022/162, 2022. `https://eprint.iacr.org/2022/162`.

[13] Anamaria Costache, Kim Laine, and Rachel Player. Evaluating the effectiveness of heuristic worst-case noise analysis in FHE. In Liqun Chen, Ninghui Li, Kaitai Liang, and Steve A. Schneider, editors, *ESORICS 2020, Part II*, volume 12309 of *LNCS*, pages 546–565. Springer, Heidelberg, September 2020.

[14] Anamaria Costache, Lea Nürnberger, and Rachel Player. Optimisations and trade-offs for HElib. Cryptology ePrint Archive, Paper 2023/104, 2023. `https://eprint.iacr.org/2023/104`.

[15] Jack L. H. Crawford, Craig Gentry, Shai Halevi, Daniel Platt, and Victor Shoup. Doing real work with FHE: the case of logistic regression. In Michael Brenner and Kurt Rohloff, editors, *Proceedings of the 6th Workshop on Encrypted Computing & Applied Homomorphic Cryptography, WAHC@CCS 2018, Toronto, ON, Canada, October 19, 2018*, pages 1–12. ACM, 2018.

[16] Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 335–352. Springer, Heidelberg, August 2014.

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

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

[19] 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.

[20] 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.

[21] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

[22] Shai Halevi and Victor Shoup. Design and implementation of HElib: a homomorphic encryption library. Cryptology ePrint Archive, Report 2020/1481, 2020. `https://eprint.iacr.org/2020/1481`.

[23] Shai Halevi and Victor Shoup. Helib 2.2.1. GitHub Repository homenc/HElib, 2021.

[24] Ilia Iliashenko. *Optimisations of Fully Homomorphic Encryption*. PhD thesis, KU Leuven, 2019.

[25] Andrey Kim, Yuriy Polyakov, and Vincent Zucca. Revisiting homomorphic encryption schemes for finite fields. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 608–639, Cham, 2021. Springer International Publishing.

[26] Andrey Kolmogorov. Sulla determinazione empirica di una lgge di distribuzione. *Inst. Ital. Attuari, Giorn.*, 4:83–91, 1933.

[27] Lattigo v4. Online: `https://github.com/tuneinsight/lattigo`, August 2022. EPFL-LDS, Tune Insight SA.

[28] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *International Colloquium on Automata, Languages, and Programming*, pages 144–155. Springer, 2006.

[29] 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.

[30] 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`.

[31] Palisade lattice cryptography library (release 1.10.6), Dec 2020.

[32] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[33] Microsoft SEAL (release 4.0). `https://github.com/Microsoft/SEAL`, March 2022. Microsoft Research, Redmond, WA.

[34] Nickolay Smirnov. Table for estimating the goodness of fit of empirical distributions. *The annals of mathematical statistics*, 19(2):279–281, 1948.