Note on the noise growth of the RNS variants of the BFV scheme

Jean Claude Bajard[1] Julien Eynard[2] Paulo Martins[3] Leonel Sousa[3]
Vincent Zucca[4]
[1] Sorbonne Université, CNRS, LIP6, Paris, France
jean-claude.bajard@lip6.fr
[2] Université Grenoble Alpes, CEA, LETI, DSYS, CESTI, F-38000 Grenoble
julien.eynard@cea.fr
[3] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa
paulo.sergio@netcabo.pt, las@inesc-id.pt
[4] imec - COSIC, KU Leuven, Belgium
vincent.zucca@kuleuven.be

Abstract

In a recent work, Al Badawi et al. have noticed a different behaviour of the noise growth in practice between the two RNS variants of BFV from Bajard et al. and Halevi et al. Their experiments, based on the PALISADE and SEAL libraries, have shown that the multiplicative depth reached, in practice, by the first one was considerably smaller than the second one while theoretically equivalent in the worst-case. Their interpretation of this phenomenon was that the approximations used by Bajard et al. made the expansion factor behave differently than what the Central Limit Theorem would predict. We have realized that this difference actually comes from the implementation of the `SmMRq` procedure of Bajard et al. in SEAL and PALISADE[1] which is slightly different than what Bajard et al. had proposed. In this note we show that by fixing this small difference, the multiplicative depth of both variants is actually the same in practice.

Keywords: Lattice-based Cryptography, Homomorphic Encryption, BFV, Residue Number Systems, Software Implementation

Over the last few years, the use of the RNS representation (a.k.a CRT representation) has been essential to reach better performance for Ring-LWE based homomorphic encryption schemes. In particular, Bajard et al. ([BEHZ17]) have shown how to overcome the limitations of the RNS representation so that all the computations required by the (B)FV scheme ([FV12]) could be done in RNS, leading to considerably better performance. More recently, Halevi et al. ([HPS19]) proposed another variant using floating point computations to remove the approximations introduced by Bajard et al. and leading to simpler procedures. Intuitively, the HPS variant should have a smaller noise growth than BEHZ since it removes the noise associated with the approximations added by BEHZ. Nonetheless it was proved that, in the worst-case, both versions reach the same multiplicative depth than textbook BFV.

However, in practice, Al Badawi et al. ([QPA$^+$19]) observed that HPS reached considerably higher multiplicative depth than BEHZ. Their interpretation of this phenomenon was that the extra multiples of the ciphertext modulus q appearing during the first fast base

[1] The error has been fixed in PALISADE version 1.7.a.

extension of homomorphic multiplication of BEHZ, although reduced in size by the SmMRq procedure, transformed the ciphertexts in non-centred random variables and increased the correlation between the coefficients. Their conclusion was that the Central Limit Theorem (CLT) could not be applied to estimate the expansion factor $\delta = \sup\{\|\boldsymbol{a}\cdot\boldsymbol{b}\|_\infty / \|\boldsymbol{a}\|_\infty \|\boldsymbol{b}\|_\infty,\ \text{for } \boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}[X]/(X^n+1) \setminus \{\boldsymbol{0}\}\}$ for BEHZ. From their experiments, run on the SEAL ([SEA19]) and PALISADE ([CRRP19]) libraries, they have been able to heuristically estimate: $\delta \in \mathcal{O}(n^{0.7})$ for BEHZ while $\delta \in \mathcal{O}(\sqrt{n})$ for HPS (as per the CLT) leading to much better asymptotic parameters for HPS than BEHZ.

However we have noticed that the SmMRq procedure of BEHZ, used to reduce the aforementioned q-overflow and replicated in Algorithm 1, has been implemented differently in SEAL and PALISADE than what was proposed by Bajard et al. Indeed, the remainder modulo $\tilde{m}$, computed in the first line of Alg. 1, is supposed to be centred in zero $[\cdot]$– i.e. in $[-\tilde{m}/2, \tilde{m}/2)$ – while it has been computed as a positive remainder $|\cdot|$– i.e. in $[0, \tilde{m})$ – in both SEAL and PALISADE.

Algorithm 1 $\mathtt{SmMRq}_{\tilde{m}}((\boldsymbol{c}''_m)_{m\in\mathcal{B}_{\mathtt{sk}}\cup\{\tilde{m}\}})$: Small Montgomery Reduction modulo q ([BEHZ17])

```
Require: $\boldsymbol{c}''$ in $\mathcal{B}_{\mathtt{sk}} \cup \{\tilde{m}\}$
1: $\boldsymbol{r}_{\tilde{m}} \leftarrow [-\boldsymbol{c}''_{\tilde{m}}/q]_{\tilde{m}}$
2: for $m \in \mathcal{B}_{\mathtt{sk}}$ do
3:     $\boldsymbol{c}'_m \leftarrow |(\boldsymbol{c}''_m + q\boldsymbol{r}_{\tilde{m}})\tilde{m}^{-1}|_m$
4: end for
5: return $\boldsymbol{c}'$ in $\mathcal{B}_{\mathtt{sk}}$
```

The difference of $|q(\tilde{m}/2)\tilde{m}^{-1}|_{\mathcal{B}_{\mathtt{sk}}} = |q/2|_{\mathcal{B}_{\mathtt{sk}}} = q/2$ in magnitude is added, in average, to half of the coefficients, and so to the noise, which almost annihilates the benefit of the SmMRq procedure. Actually, the results from [QPA+19] support the results of Section 5.2 in [BEHZ17] that study the impact of $\tilde{m}$ and SmMRq on the noise growth. Note also that $\tilde{m}$ has been chosen of same size than the moduli q_i (usually more than 50-bits) in SEAL and PALISADE instead of 2^8 or 2^{16} in BEHZ. While this contributes to a further, although not necessary, reduction of the magnitude of $\boldsymbol{c}'$, it makes the SmMRq procedure slower.

In order to confirm our analysis, we have measured the multiplicative depths reached in practice by BEHZ with positive and centred remainders on SEAL and PALISADE and also those reached by HPS on PALISADE (HPS is not available in SEAL). Table 1 presents the results of our experiments. Depths were measured using 2^5 different keys, each of them running 2^5 different ciphertexts for a total of 2^{10} tests by squaring encryptions of 1 until they no longer decrypt correctly. The displayed numbers correspond to the smallest depth reached on these 2^{10} tests. The dimension n, the size of the modulus q and the standard deviation of the error distribution σ follow the homomorphic encryption standards[2] for 128 bits of post-quantum security. Secret key distribution is chosen as ternary (uniform in $\{-1, 0, 1\}^n$), plaintext modulus is set to $t = 2$ and $\sigma = 3.2$. Finally, relinearisation was performed using only the CRT decomposition, k represents the number of moduli in the decomposition of $q = q_1 \cdots q_k$ and ω the size of these moduli.

Table 1 shows results similar to those of Al Badawi et al. for positive remainders, while with a centred remainder there is a difference of at most 1 between HPS and BEHZ which is not significant on only 2^{10} tests. Moreover, as shown by Table 2, the difference of performance are very small between the two versions. In particular if we consider that the modulus $\tilde{m}$ of

[2] http://homomorphicencryption.org

n	$\log_2 q$	$k \times \omega$	Library	BEHZ		HPS
				Pos	Cen	
2^{11}	52	2×26	SEAL	1	1	-
			PALISADE	-	-	-
2^{12}	102	2×51	SEAL	2	3	-
			PALISADE	3	4	4
2^{13}	204	4×51	SEAL	7	9	-
			PALISADE	8	10	11
2^{14}	413	7×59	SEAL	15	22	-
			PALISADE	16	23	23
2^{15}	826	14×59	SEAL	32	45	-
			PALISADE	33	47	46

Table 1: Multiplicative depth observed for BEHZ, with centred and positive remainder, and HPS for 128 bits standard security parameters.

BEHZ is not chosen as a small power of two in PALISADE.

n	$\log_2 q$	$k \times \omega$	HPS		BEHZ	
			Dec	Mult	Dec	Mult
2^{12}	102	2×51	0.4	5.5	0.7	5.8
2^{13}	204	4×51	1.2	24.3	1.2	23.7
2^{14}	413	7×59	8	107	5.3	112
2^{15}	826	14×59	39.1	609	22.8	623

Table 2: Timings (in ms) of decryption (for $t = 2$) and homomorphic multiplication of HPS and BEHZ for 128 bits standard security parameters. Timings computed on an average of 2^{10} tests run using the PALISADE library in single-threaded mode on an i7-4810MQ CPU @ 2.80GHz with Turbo-boost turned off.

We can hence conclude that, contrarily to the conclusions in [QPA$^+$19], the noise growth in BEHZ and HPS behaves the same in practice and in theory. This, combined with the fairly equivalent performance, allow us to argue that there is no clear advantage for using one version instead of the other in practice.

Acknowledgments

This work was partially funded by the ANR ARRAND 15-CE39-0002-01, the national funds through Fundação para a Ciência e a Tecnologia (FCT) under project number UID/CEC/50021/2019, the Ph.D. grant with reference SFRH/BD/103791/2014; through the Pessoa/Hubert Curien programme with reference 4335 (FCT)/40832XC (CAMPUSFRANCE) and the IARPA HECTOR project under the solicitation number IARPA-BAA-17-05.

References

[BEHZ17] 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 Heys, editors, *Selected Areas in Cryptography – SAC 2016*, pages 423–442, Cham, 2017. Springer International Publishing.

[CRRP19] David B Cousins, Gerard W. Ryan, Kurt Rohloff, and Yuriy Polyakov. PALISADE Homomorphic Encryption Software Library. `https://gitlab.com/palisade/palisade-release`, September 2019.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive*, 2012.

[HPS19] Shai Halevi, Yuriy Polyakov, and Victor Shoup. An Improved RNS Variant of the BFV Homomorphic Encryption Scheme. In Mitsuru Matsui, editor, *Topics in Cryptology – CT-RSA 2019*, pages 83–105, Cham, 2019. Springer International Publishing.

[QPA$^+$19] A. Qaisar Ahmad Al Badawi, Y. Polyakov, K. M. M. Aung, B. Veeravalli, and K. Rohloff. Implementation and Performance Evaluation of RNS Variants of the BFV Homomorphic Encryption Scheme. *IEEE Transactions on Emerging Topics in Computing*, pages 1–1, 2019.

[SEA19] Microsoft SEAL (release 3.4). `https://github.com/Microsoft/SEAL`, October 2019. Microsoft Research, Redmond, WA.