On the concrete hardness of the Approximate GCD problem

Éricles A. A. B. Lima[1], **Hilder V. L. Pereira**[2]

[1]IMECC – Universidade Estadual de Campinas (Unicamp)
Cidade Universitária Zeferino Vaz - Barão Geraldo, Campinas - SP, 13083-970

[2]Istituto de Computação – Universidade Estadual de Campinas (Unicamp)
Cidade Universitária Zeferino Vaz - Barão Geraldo, Campinas - SP, 13083-970

e271814@dac.unicamp.br, hilder@unicamp.br

Abstract. *The problem of the Approximate Greatest Common Divisor (AGCD) has proven to be an important theoretical foundation for many cryptographic constructions, mostly due to its use in homomorphic encryption schemes and post-quantum cryptography. This work analyzes the concrete hardness of the AGCD, showing the cost of known algorithms with respect to the number of basic operations. Then, based on this analysis, we provide a tool to estimate the concrete hardness of any AGCD instance. Finally, we use our tool to estimate the security level of existing AGCD-based schemes.*

Resumo. *O problema do Máximo Divisor Comum Aproximado (AGCD, na sigla em inglês) tem se mostrado um fundamento teórico importante para várias construções criptográficas, principalmente devido ao seu uso em esquemas de criptografia homomórfica e de criptografia pós-quântica. Este trabalho analisa a dificuldade concreta do AGCD, mostrando como o custo dos algoritmos conhecidos varia em relação ao número de operações básicas. Em seguida, com base nessa análise, fornecemos uma ferramenta para estimar a dificuldade concreta de qualquer instância do AGCD. Por fim, usamos nossa ferramenta para estimar o nível de segurança de esquemas existentes baseados em AGCD*

1. Introduction

Modern cryptographic schemes are typically proven secure within a specific computational model under the assumption that a particular mathematical problem is computationally hard to solve. Consequently, evaluating the practical efficiency of a cryptographic scheme requires analyzing the most effective known algorithms for solving its underlying hard problem, since these algorithms directly influence the choice of parameters, key sizes, ciphertext sizes, and overall performance. More precisely, for a target security level λ — say, $\lambda = 128$ — the scheme's parameters are selected so that the best known attack against the underlying problem is expected to require at least 2^λ computational operations.

For well-established schemes, such as the RSA and schemes based on the discrete log problem (DLP), the algorithms are already well-studied and choosing the parameters to achieve a desired security level is straightforward. For instance, for schemes

Acknowledgment: This work was funded, in part, by the São Paulo Research Foundation (FAPESP), Brazil, under Grant No. 2023/12755-8.

This work was funded, in part, by the Coordination for the Improvement of Higher Education Personnel, CAPES, Brazil, under Grant No. 88882.329069/2014-01.

based on the DLP over elliptic curves, such as the Elliptic Curve Digital Signature Algorithm (ECDSA), one just has to choose a group with prime order larger than 2λ [Johnson et al. 2001]. However, for new post-quantum secure schemes, the situation is more complicated, since many of them are based on problems that have not been widely used before in cryptography, e.g., computational problems related to isogenies [Feo 2017]. In some cases, it is not clear what the best algorithm is; in others, it is not clear how to estimate the computational cost of the best algorithm.

For instance, lattice-based cryptography [Peikert 2016] is a promising direction to construct post-quantum secure schemes, especially when based on the Learning With Errors problem (LWE), which became popular due to its versatility, allowing one to construct primitives ranging from key-exchange protocols [Bos et al. 2018] to advanced schemes like fully homomorphic encryption (FHE) [Brakerski and Vaikuntanathan 2011]. However, deciding what is the best algorithm to solve the LWE problem is not easy, as it may depend on the different parameter regimes one instantiates the LWE. Moreover, even estimating the running time of such algorithms is complicated, as it depends on how one models the internal building blocks of the algorithms.

Thus, as a way of separating the tasks of projecting cryptographic schemes and estimating their securities, in [Albrecht et al. 2015] the concrete security of the LWE problem was analyzed, and the Lattice Estimator program was published [0], which receives the parameters used in a LWE-based scheme, tests the feasible algorithms against the LWE problem and returns the security level in bits. Currently, the corresponding tool also allows for the evaluation of security for schemes based on the NTRU and SIS problems, and it has become a standard reference for researchers and practitioners in lattice-based cryptography.

Another hard problem that can serve as the foundation for lattice-based cryptosystems is the Approximate Greatest Common Divisor (AGCD) problem [van Dijk et al. 2010]. Although AGCD-based schemes share several important characteristics with LWE, including being hard even for quantum computers and applicability to FHE, there is no widely adopted publicly available program for assessing their security. This paper aims to address that gap by consolidating the existing knowledge on the hardness of AGCD into a tool that capable of estimating the concrete security of schemes based on the AGCD problem.

1.1. Contributions

We provide a tool that can be used to estimate the parameters of AGCD-based cryptographic schemes. Using that tool, we observed that the celebrated OL attacks are not always applicable, and when applicable might be less effective than previously considered. We provide bounds for the applicability of OL attacks and also use the tool to estimate the parameters of several AGCD-based schemes and report the findings.

2. Background

For any set X, we write $x \leftarrow X$ when x is sampled uniformly at random from X. If $n_1, \cdots, n_\ell$ are pairwise coprime integers, setting $n := \prod_i n_i$, we denote by

[0]Github reposiroty of the Lattice Estimator: `https://github.com/malb/lattice-estimator/`

$CRT_{(n_1,\cdots,n_\ell)}(a_1,\cdots,a_\ell)$ (or in its abbreviation $CRT_{(n_i)}(a_i)$), the unique integer $x \in [\![0, n-1]\!]$ that satisfies $x = a_i \ mod \ n_i$ for $1 \leqslant i \leqslant \ell$. This map defines an isomorphism between $\mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_\ell}$ and $\mathbb{Z}_n$, where $\mathbb{Z}_k$ denotes the ring of integers $\bmod k$ (this result is known as the Chinese Remainder Theorem, or CRT).

2.1. The AGCD problem

Like the LWE, the AGCD arises by adding noise to a problem that is classically easy to solve, namely, finding the GCD of a list of numbers, making it difficult.

Definition 1 (AGCD)**.** *For $\rho < \eta < \gamma$, the (ρ, η, γ)-AGCD (approximate greatest common divisor) problem, (aka ACD), consists of retrieving the η-bit integer p given many γ-bit integers $x_i := pq_i + r_i$, where each r_i is uniformly sampled from the integer interval $[\![-2^\rho, 2^\rho]\!]$ and q_i is uniformly sampled from $[\![0, 2^\gamma/p[\![$.*

The set of AGCD samples is then defined as

$$D_{\gamma,\rho}(p) := \{x = pq_i + r_i : q_i \leftarrow [\![0, 2^\gamma/p[\![,\ r \leftarrow [\![-2^\rho, 2^\rho]\!]\}\,. \tag{1}$$

The (ρ, η, γ)-AGCD is defined, then, as the problem of finding p given a arbitrarily many samplings of $D_{\gamma,\rho}(p)$.

The AGCD can also be defined as a decisional problem, instead of a computational one. In this case, one has to distinguish between $D_{\gamma,\rho}$ and the uniform distribution $U([\![0, 2^\gamma]\!])$. In [Coron et al. 2014], it was proved the computational and decisional versions are polynomially equivalent, thus, some subsequent schemes were based on the decisional version, as it makes the security proofs easier. Moreover, in [Cheon and Stehlé 2015] a (classical) reduction from the LWE to a decisional variant of the AGCD (with different distributions for the noise terms) was shown. Thus, (that variant of) AGCD is at least as difficult as the LWE, supporting the confidence that AGCD is a suitable problem at least in terms of computational complexity.

2.2. Variations of the AGCD problem

This section presents commonly used variants of the AGCD problem.

2.2.1. Partial AGCD

In the Partial AGCD problem (AGCD), in addition to the AGCD samples, one also has a multiple of the secret integer p, that is, given $x_0 := p \cdot q_0$ for some q_0 and values $x_i \leftarrow D_{\gamma,\rho}(p)$ for $i \geqslant 1$, one has to find p [Coron and Pereira 2019].

Clearly, PACD cannot be more difficult than the AGCD, and appears to be easier, as one has now two ways of recovering p, either by factoring x_0 or by ignoring x_0 and solving the associate AGCD problem. However, apart from the factorization, all the algorithms to solve PAGCD also work for the the AGCD, i.e., there is no clear way to use x_0 to improve the existing algorithms for the AGCD problem.

It is of note that, since PAGCD assumes that factoring is hard, schemes based on it are vulnerable to quantum computers.

2.2.2. CRT-ACD

In order to reduce data expansion when using the AGCD, it was proposed to encode many AGCD samples into a single one via the Chinese Remainder Theorem (CRT). In this variant, instead of a single secret value p, one has a list of secret integers $p_1, ..., p_n$, where $n = \lfloor \gamma/\eta \rfloor$. The idea is that a sample x_i is a different AGCD sample modulo each p_j, i.e., $x_i \equiv e_{i,j} \pmod{p_i}$ where $e_{i,j} \in [\![-2^\rho, 2^\rho]\!]$ is a small noise term. This problem is known as CRT-ACD (from *Chinese Remainder Theorem ACD*), and also as *multi-prime AGCD*, or CCK-ACD [Cheon et al. 2013], and uses the Chinese remainder theorem to encode several p_i into a single p.

Formally, the CRT-ACD problem is defined as follows.

Definition 2 (CRT-ACD)**.** *For parameters (γ, η, ρ, n), let $p_1, \cdots, p_n$ be η-bit pairwise coprime integers, $P = \prod_{i=1}^{n} p_i$ and $D_{\gamma,\eta,\rho,n}(p_i)$ be the following distribution on γ-bit integers*

$$D_{\gamma,\eta,\rho,n}(p_i) = \{T \cdot P + CRT_{(p_i)}(r_i) : T \leftarrow [\![2^{\gamma-1}/P, 2^\gamma/P]\!],\ r_i \leftarrow [\![-2^\rho, 2^\rho]\!]\}. \quad (2)$$

The problem (γ, η, ρ, n)-CRT-ACD problem consists of finding each p_i given arbitrarily many samples of $D_{\gamma,\eta,\rho,n}(p_1, \cdots, p_n)$.

The definition above is given as posed in [Lima Pereira 2020], but definitions in [Kim et al. 2013] and [Cheon et al. 2013] are slightly different.

2.3. Lattices

A lattice is a discrete set of vectors generated by integer linear combinations of some basis vectors. In other words, given a set of linear independent vectors $\{b_1, \cdots, b_n\} \subset \mathbb{R}^m$, we say that $B = [b_1 \ \cdots \ b_n] \in \mathbb{R}^{m \times n}$ is a basis matrix of the lattice $L = L(B) := \{B \cdot u : u \in \mathbb{Z}^n\}$. In this case, the dimension of L is n. Two matrices B_1 and B_2 are bases of the same lattice if there is a unimodular matrix $U \in \mathbb{Z}^{n \times n}$ such that $B_1 = B_2 \cdot U$. The determinant of the lattice, denoted $\det(L)$, is a measure of how dense the lattice is and is defined as $\sqrt{\det(B \cdot B^T)}$. It is easy to see that $\det(L)$ does not depend on the particular basis one chooses to compute it on, since if $\tilde{B}$ is another basis, then $\tilde{B} = B \cdot U$ where $\det(U) = \pm 1$, thus, $\sqrt{\det(\tilde{B} \cdot \tilde{B}^T)} = \sqrt{\det(B)\det(U \cdot U^T)\det(B^T)} = \sqrt{\det(B \cdot B^T)}$.

Since lattices are discrete, their nonzero vectors cannot be arbitrarily short. Thus there is a minimum nonzero norm, typically denoted $\lambda_1(L)$ - that is, we say that a vector $v \in L$ is a shortest (nonzero) vector if $\|v\|_2 = \lambda_1(L)$. The problem of finding such v given a basis of L is known as the shortest vector problem (SVP) and it is an NP-hard problem (under randomized reductions) [Ajtai 1998]. Below, we define the approximate version of SVP with approximation factor γ. The exact version is obtained by setting $\gamma = 1$.

Definition 3. *γ-SVP (The approximate shortest vector problem)*

Given a lattice basis B, find a non-zero shortest vector v such that $\|v\|_2 \leqslant \gamma \cdot \lambda_1(L)$.

Another important problem defined over lattices is the problem of finding the vector in the lattice that is the closest to a given vector. Again, we present the approximate version of this problem.

Definition 4. *γ-CVP (The approximate closest vector problem)*

Given a basis B of a lattice L and a (target) vector t, find a $v \in L$ closest to t up to a factor γ, that is, a $v \in L$ such that $\|v - t\|_2 \leqslant \gamma \cdot \|u - t\|_2$ for all $u \in L$.

2.4. Lattice reduction

Lattices basis are invariant by unimodular transformations, that is, by integer matrices U that satisfy $det(U) \in \{-1, 1\}$. Thus, the reduction process consists in finding a unimodular matrix U such that the columns of BU satisfy some reduction criterion. There are several such criteria, the first of which was put forward in 1850 by Hermite [Hermite 1850]. In 1873, Korkine and Zolotarev proposed a stronger version of this reduction [Korkine and Zolotareff 1873, Korkine and Zolotareff 1877], which became known as the HKZ reduction. The HKZ reduction obtains a shortest vector, therefore existing at one extreme of the reduction algorithms, while, at the other, there is the LLL reduction, which finds a basis whose shortest vector is at an exponentially large distance from the shortest vector of the lattice. Between these two extremes, there is the BKZ family of reductions, which is indexed by a parameter β - when $\beta = 2$, BKZ corresponds to LLL and, when $\beta = n$, to HKZ. These methods are not exactly disjoint, however, since, for example, BKZ uses HKZ reduction on smaller blocks as a subroutine and also runs LLL on the entire basis as a preprocessing step, and algorithms for HKZ generally use both BKZ and LLL to "massage the basis", before applying more specific methods. For end-use cryptographer purposes, BKZ reduction is the most relevant, but it is important to keep in mind that BKZ makes use of both HKZ and LLL.

2.5. Hermite factor

With these various notions of reduction, one might ask how to evaluate the quality of a basis in more general terms. Minkowsky's theorem guarantees that $\exists x$ such that

$$\|x\|_2 \leqslant \sqrt{n} \cdot det(L)^{1/n}, \tag{3}$$

which suggests we should look for a vector that is small compared to $det(L)^{1/n}$. Therefore, for a given basis $B = \{b_1, \cdots, b_n\}$, we define the so-called "Hermite factor", $\delta_0^n = \frac{\|b_1\|}{det(L)^{1/n}}$, where b_1 is a smallest vector in that basis. In practice, since the Hermite factor grows exponentially with the dimension, the root Hermite factor is used, defined as the nth root of δ_0^n, i.e., δ_0, which allows comparing the performance of reduction algorithms in different dimensions[1].

The interpretation of this term is that δ_0 being close to the geometric mean of the basis vectors, indicated by the determinant, indicates that the algorithm found a vector b_1 that is close to the average distance between the points of the lattice, indicating a good reduction quality.

Made popular by [Gama and Nguyen 2008], the Hermite factor is the standard "ruler" for measuring cryptographic security in lattice-based schemes - as in these cases the reduction of the lattice is the main computational bottleneck. If an attacker can obtain a sufficiently small Hermite factor, they may be able to break lattice-based cryptographic schemes. To put that in perspective, LLL typically obtains $\delta_0 \approx 1.021$, although the worst case predicted by theory is ≈ 1.075 [Ding et al. 2018], whereas BKZ typically reaches $\delta_0 \approx 1.01$, depending on the block size [Zhao and Ding 2023].

[1]In this work, except when explicitly stated otherwise, δ_0 refers to the root Hermite factor.

2.6. Solving AGCD by orthogonal lattices

The following shows how lattice techniques can be used to attack AGCD-based schemes.

The use of lattices in AGCD-based schemes occurs mainly through so-called *orthogonal lattice attacks*. The first such orthogonal lattice attack (OL) was proposed in [van Dijk et al. 2010], using a lattice first proposed in [Nguyen and Stern 1997]. The second OL attack, which will be presented below, was also put forward in [van Dijk et al. 2010], and subsequently revisited in [Galbraith et al. 2016, Cheon and Stehlé 2015]. OLs essentially consist of finding vectors v orthogonal to q, and that impose some relation on r.

Given n samples $agcd$, $x_1, \cdots, x_n$, define the lattice L with the following basis matrix, where ρ is the length of r in bits and $R = 2^\rho$ is an upper bound on the error terms r_i:

$$B = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ 2^\rho & & & \\ & 2^\rho & & \\ & & \ddots & \\ & & & 2^\rho \end{pmatrix}.$$

From this basis, we have that $v \in L \Rightarrow \exists u$ such that $v = Bu = (\sum u_i x_i, 2^\rho u_1, \cdots, 2^\rho u_n)$. Heuristically, if we had a vector $w = (1, -\frac{r_1}{2^\rho}, \cdots, -\frac{r_n}{2^\rho})$, orthogonal to Bu, one could use it to obtain a vector $r = (r_1, \cdots, r_n)$, from which p is obtainable. Moreover, we have that

$$\begin{aligned} v \cdot w &= \sum u_i(x_i - r_i) \\ &= p \sum u_i q_i \\ &\equiv 0 \; mod \; p. \end{aligned}$$

From which it follows that, if $\|v\|$ is sufficiently small, the only multiple of p that satisfies this condition is 0 itself, and $v \perp w$.

More precisely, if $|v \cdot w| < p$, then $v \cdot w = 0$, and for $|v \cdot w| < p$ to hold it's sufficient that $\|v\| < \frac{p}{\|w\|}$. Then, the attack consists in running a lattice reduction algorithm in L in order to recover n short vectors v_i, which will constitute the rows of V, so as to find a w satisfying $Vw = 0$. According to [Xu et al. 2018], the amount of AGCD samples, n, needs to satisfy $n > \frac{\gamma - \rho}{\eta - \rho}$.

There are several variants of this attack, with slightly different base matrices, and which have essentially the same effectiveness in both theory and practice. In [Xu et al. 2018], an analysis of 3 of these variants is made, and an *optimized* OL is proposed that uses a lattice parameterized by α.

$$L(\alpha) := \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ \alpha & & & \\ & \alpha & & \\ & & \ddots & \\ & & & \alpha \end{pmatrix},$$

which is a generalization of the previous case, where one would have $\alpha = 2^\rho$. They find the value of α as $\frac{\sqrt{n}}{n-1}2^\rho$ to provide an optimized upper bound.

They also analyze another optimization, inspired by the rounding technique used in [Bi et al. 2014], which consists of using the lattice with the basis

$$L(\alpha) := \begin{pmatrix} \lfloor \frac{x_1}{\alpha} \rceil & \lfloor \frac{x_2}{\alpha} \rceil & \cdots & \lfloor \frac{x_n}{\alpha} \rceil \\ 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{pmatrix},$$

that reduces the maximum entries in the base matrix by approximately $\gamma - \rho$ bits, which can perform better when $\gamma - \rho$ is small. For this case, they find α optimal as $\alpha = \frac{\sqrt{n}}{(n-1)(\sqrt{n}+1)}2^\rho$. They also show that the time complexity of these attacks is equal when $\gamma \gg \rho$.

3. Concrete security

Concrete security is a practice-oriented approach [Bellare et al. 1997] that requires a more precise estimate of the computational complexity of adversarial tasks than polynomial equivalence would allow. Concrete security aims to quantify an upper bound for the probability of any adversary breaking the system under study.

More precisely, a security proof involves an upper bound for the adversary's advantage in breaking the system as a function of adversarial resources and the size of the problem. The resources available to the adversary generally include execution time and memory, as well as other system-specific resources, such as the number of plaintexts they can obtain or the number of queries they can make to any available oracle. By "size of the problem", it's generally meant the key size, but may also include the size of cipher blocks when these are defined in the system.

Along these lines, the literature was consulted to obtain the most effective attacks against AGCD-based schemes, and to estimate their parameters for a concrete security level. More specifically, it is desired to estimate the parameters γ, ρ, η to obtain a desired security level, λ. In lattice-based attacks, that entails in finding the value of δ_0 that forces the lattice reduction algorithm to spend at least 2^λ operations to get an adequate reduction.

In this section, we analyze the existing algorithms for the AGCD problem and how one can estimate concrete security levels from them.

3.1. Orthogonal lattice attacks

As described by Xu et al., to estimate whether (γ, ρ, η) achieves a security level λ by means of assessing OL attack, it is necessary to map the required δ_0 of an OL attack, determine the size of the BKZ block, and estimate the resulting processing time.

In [Xu et al. 2018], an upper bound is obtained for $\log\ \delta_0$, whose exact formula depends on the variant of the OL attack used[2]. For the "optimized OL attack" of [Xu et al. 2018], the condition given by the inequality given from theorem 2 therein, isolating $log\ \delta_0$, reads

$$\log\ \delta_0 < \frac{1}{n}\left[(\eta - \rho) - \frac{\gamma - \rho}{n} - \log\sqrt{n^2 + 2n}\right] \tag{4}$$

from where the condition for δ_0 is obtained.

It is thus necessary to determine the optimal quantity n of AGCD samples in order to estimate δ_0. The optimal quantity is given by Xu as $n = \frac{2(\gamma-\rho)}{\eta-\rho}$, which is found by maximizing the right hand side of inequality 4, ignoring the log term. Namely, one seeks to maximize the quantity $\frac{(\eta-\rho)}{n} - \frac{\gamma-\rho}{n^2}$ as a function of n, which can be found, using ordinary calculus, to be attained when $n = \frac{2(\gamma-\rho)}{\eta-\rho}$. That is a simplification and if the log term is taken into account, the expression given by the right-hand-side of inequality 4 must be maximized instead. The estimated optimal n differs in that case - for instance, if $\gamma = 680, \eta = 125$ and $\rho = 100$, the original estimate is 46, whereas the one obtained by numerically maximizing the whole formula is 52 - that difference, however, isn't enough to change the estimated minimum BKZ block-size β required to achieve the δ_0 so obtained, thus justifying the simplification adopted in [Xu et al. 2018].

Formula 4 allows one to find a δ_0. Once it has been obtained, it is necessary to calculate the required BKZ block size, β, from the δ_0. This can be done using Chen's formula (as done in the original *lattice-estimator*):

$$\delta_0 = \left(\frac{\beta}{2\pi e}(\pi e)^{1/\beta}\right)^{\frac{1}{2(\beta-1)}},$$

which allows obtaining β from δ_0 by Newton's method.

Finally, using the obtained n and β, the computational cost to solve the AGCD instance using the BKZ-β algorithm is estimated. If the SVP oracle is instantiated with *sieving*, the cost was estimated as $T = c \cdot n \cdot 2^{0.292\beta+16.4}$ clock cycles [Ding and Tao 2014, Laarhoven 2015], where c is a small constant [Xu et al. 2018], which took the value of 8 in [Albrecht 2017, Albrecht et al. 2017], but in [Albrecht et al. 2019] it was noted that, with the improvements used in G6K, BKZ-β would be about 4 times faster, making $c = 2$.

Furthermore, the exponent of 0.292, which reflects the asymptotic cost of the SVP oracle, can be updated to 0.2075, from the bgj1 regime that G6K explores [Zhao et al. 2024]. It is worth noting, however, that the authors of G6K themselves warn against such overconfident extrapolations in [Albrecht et al. 2019].

[2]Which, it is worth noting, are all asymptotically equivalent

There are other cost-models for the running of BKZ-β that could be reasonably chosen, and whose choice may affect the estimation of the running time. In *lattice-estimator*, 10 such cost models are available for use, namely ones presented in [Albrecht et al. 2020, Albrecht et al. 2021, Alkim et al. 2016, Becker et al. 2016, Chen and Nguyen 2011, Avanzi et al. 2019, MATZOV , Guo and Johansson 2021, Laarhoven et al. 2015, Chailloux and Loyer 2021], all of which are also available for estimating cost of attacking AGCD-based schemes (using the *lattice-estimator* structure).

For instance, if the SVP oracle is instantiated with *sieving*, the cost was estimated as $T_{OL} = c \cdot n \cdot 2^{0.292\beta+16.4}$ clock cycles [Ding and Tao 2014, Laarhoven 2015], where c is a small constant [Xu et al. 2018], which took the value of 8 in [Albrecht 2017, Albrecht et al. 2017]. However, in the so-called *core SVP cost model*, the cost is simply estimated as $T_{OL} = 2^{0.292\beta}$ [Albrecht et al. 2021].

4. Other algorithms for the AGCD problem

OLs are recognized as the main class of attacks against AGCD, meaning that it is expected to yield the best results even with reasonably large parameters. However, there are other attacks that may fare better for some specific parameter range.

Firstly, a trivial such attack is the exhaustive search on the parameter η, namely trying all the $2^{\eta-1}$ integers $\hat{p}$ in the interval $[2^{\eta-1}, 2^{\eta}]$, which will at some point give us $\hat{p} = p$. This attack requires around $T_p = 2^{\eta}$ operations, thus, warding against that attack requires setting $\eta > \lambda$.

Another such attack is also related to exhaustive search, but on the parameter ρ. The first idea is to take two AGCD samples $x_i = p \cdot q_i + r_i$, for $i \in \{1, 2\}$, and try all the $2^{2\rho}$ possible pairs $(\hat{r}_1, \hat{r}_2)$, computing $\gcd(x_1 - \hat{r}_1, x_2 - \hat{r}_2)$ for each of them. At some point, $(\hat{r}_1, \hat{r}_2) = (r_1, r_2)$, so that $x_i - \hat{r}_i = pq_i$ and the GCD will give us p (or a small multiple of it). This basic strategy is improved by other algorithms using time-memory tradeoffs and requiring many samples instead of two. In [Pereira 2020], this class of attacks is discussed in detail. At the end, the number of operations one needs to run this attack is given by

$$T_{\gcd} = \rho^2 \cdot \gamma \cdot \log(\gamma) \cdot 2^{\rho} \tag{5}$$

Those attacks are added to the estimator for completeness.

5. Estimator algorithm

The estimator algorithm is straightforward, ans its implementation is available at [Aquiles Lima and Lima Pereira 2026]. First it checks whether the given parameters are valid (that is, if $\eta < \rho$ and $\gamma < \eta$). Then it calculates the optimal n for the OL attack, and uses it to find an upper bound for $\log \delta_0$. It then checks whether $\log \delta_0 \leqslant 0$, in which case the OL attack by [Xu et al. 2018] can't be expected to work.

If the estimator finds that the OL attack may be applicable, it proceeds to calculate the BKZ block-size β needed to achieve the found δ_0. Then it applies the given BKZ cost model and estimates the security level using the resulting time.

The program also follows suit by calculating the expected cost of the GCD attack on ρ and of the "trivial" attack on η, and reports the estimated λ found, together with the

Table 1. Comparison of parameters, ciphertext expansion and ciphertext size of typical implementations of CKKS and the AGCD-based scheme by [Cheon and Stehlé 2015].

Mult. levels	Parameters	Ciphertext Expansion	Ciphertext size
$L = 1$	$(N, \log Q) = (2^{11}, 56)$	224	28.6 KB
	$(\rho, \eta, \gamma) = (121, 128, 1965)$	128	0.24 KB
$L = 2$	$(N, \log Q) = (2^{12}, 111)$	444	113 KB
	$(\rho, \eta, \gamma) = (114, 128, 7758)$	128	0.96 KB
$L = 4$	$(N, \log Q) = (2^{13}, 220)$	880	450 KB
	$(\rho, \eta, \gamma) = (100, 128, 30828)$	128	3.85 KB

expected time, in terms of clock cycles. In addition, if the estimated number of samples n is too small, HKZ may be applicable, and the program estimates its running time. The estimated security level λ is the minimum among all the tested attacks (that is, the minimum among the $\log_2$ of the estimated running time of the tested attacks).

5.1. Overall estimation

Putting it all together, given AGCD parameters (γ, η, ρ), our estimator computes T_{OL}, T_p, and T_{gcd}, then the concrete security level is $\lambda = \log(\min(T_{OL}, T_p, T_{\text{gcd}}))$.

6. Concrete estimation of existing AGCD-based schemes

We now turn to using that estimator in order to estimate concrete parameters for existing schemes. Some papers provide a concrete set of parameters, whereas some provide only asymptotics. In the first case, we evaluate the parameters given, estimate the security level λ and check whether it matches with the λ estimated by the author. In the second case, we first fix the security parameter λ, and then set the parameters γ, η and ρ based on the paper's recommendations.

At [Coron et al. 2012], some concrete parameters for the DGHV scheme are suggested. For their "large" instance, they suggest $(\gamma, \eta, \rho) = (19.53 \cdot 10^6, 2698, 71)$ to achieve 72 bits of security. However, for this set of parameters, our estimator gives us $\lambda = 42$, indicating that another set of parameters would be necessary for that λ.

In [Cheon and Stehlé 2015], an AGCD-based scheme for leveled homomorphic encryption is given, but a concrete set of parameters is not provided. The authors recommend that to evaluate circuits of multiplicative depth equal to L, one has to set $\eta = \rho + L \cdot \log \lambda$. Thus, we can fix the security level as $\lambda = 128$ and derive concrete parameters for some values of L by grid searching parameters: trying each parameter from some base quantity up to a somewhat big quantity. Here, we tried, for each level L, $\rho \in \{100, \cdots, 200\}$, $\eta = \rho + L \cdot \log_2(\lambda)$ and $\gamma \in \{\eta + 500, 1000 \cdot \eta\}$, and list in the table 1 the smallest parameters that achieved the desired $\lambda = 128$.

By using the CRT-AGCD, one can pack $s = \lfloor \gamma/\eta \rfloor$ messages in a single γ-bit ciphertext, which gives us a message expansion of $\gamma/s \approx \eta$.

Considering that RLWE-based FHE schemes such as BGV and CKKS are implemented with RNS representation [Bajard et al. 2017, Cheon et al. 2019, Lee et al. 2022],

where the ciphertext modulus is represented as a product of $L + 1$ small primes to evaluate circuits with multiplicative depth L, and also that each prime has typically around 50 bits [Halevi and Shoup 2014], then for a given ciphertext modulus Q with $\log Q$ bits, one has $L \approx \log Q/50$. For each polynomial degree N, we have a maximum value of $\log Q$ to maintain security. Each RLWE ciphertext has a maximum number of slots, which represents how many messages one can encrypt in each ciphertext. Considering CKKS, the number of slots is $N/2$ (which is much higher than for BGV), thus, for a ciphertext with $2N \log Q$ bits, one can encrypt $N/2$ messages, giving us a ciphertext expansion of $4 \log Q$.

The parameters for those schemes are then represented by N and $\log Q$. Following the parameter selection suggested by [Albrecht et al. 2018], for $\lambda = 128$, we have the parameters and ciphertext expansions described in Table 1.

Surprisingly, both the ciphertext size and the ciphertext expansion of the AGCD-based scheme [Cheon and Stehlé 2015] are actually better than the RLWE-based schemes, even considering the CKKS scheme.

At last, in [Pereira 2021] a new fully homomorphic encryption scheme based on the AGCD with fast bootstrapping is presented. The construction actually uses two schemes, one with very small ciphertexts and another called the accumulator. The small ciphertexts are then refreshed by the accumulator after each homomorphic operation, following the blueprint of TFHE [Chillotti et al. 2020]. For that base scheme, [Pereira 2021] uses $(\gamma, \eta, \rho) = (680, 105, 100)$ and claims 100 bits of security according to a somehow crude analysis of the OL attack. However, with our estimator, we obtain $\lambda = 85$ for those parameters, indicating a much lower security level. To achieve $\lambda = 100$, one has to increase γ from 680 to at least 800. However, notice that would increase the running time of their fast bootstrapping. In more detail, the bootstrapping uses a decomposition basis B, which, according to their Table 2, assumes the values $2^6, 2^7$, and 2^8. The γ-bit ciphertext of the base scheme is then decomposed into $L = \lceil \gamma / \log B \rceil$ words, and the bootstrapping iterates L times. Thus, increasing γ to 800 gives us the following 3 new values of L: 134, 115, and 100, which are about 1.17 times larger than the original values of L. Therefore, we can predict an impact of 1.17 times in the bootstrapping times. In particular, for their first parameter set, the running time increases to $0.94s$ to $1.1s$, thus, no longer being below the one second mark.

6.1. Conditions on the OL attack

With an estimator in hand it was noted that, using the parameters set at [Pereira 2021], namely $\gamma = 680, \eta = 105, \rho = 100$, the OL attack by Xu estimates a $\delta_0 < 1$, whose infeasibility indicates that the attack isn't expected to work. In [Xu et al. 2018], it's not observed that there are sets of parameters for which it's not expected for their optimized OL attack to work, so we develop it here in terms of $gap := \eta - \rho$ and $signal := \gamma - \rho$.

The attack is feasible only if $\exists n > 1$ such that the quantity estimated in inequality 4 is positive, that is

$$R(n) := \frac{\eta - \rho}{n} - \frac{\gamma - \rho}{n^2} - \frac{1}{2n} \log(n^2 + 2n) > 0. \tag{6}$$

Multiplying by n yields $(\eta - \rho) - \frac{\gamma - \rho}{n} - \frac{1}{2} \log(n^2 + 2n) > 0$. In terms of the gap,

we have $\eta - \rho > \frac{\gamma-\rho}{n} + \frac{1}{2}\log(n^2+2n)$. As the first term from the right-hand-side diverges, but the second doesn't, it's clear that for any fixed gap, $\exists n$ such that the log exceeds it, entailing that attack is always unfeasible for sufficiently large dimensions. The question then is whether $\exists n$ such that the attack works.

At the feasibility boundary one has from inequality 6

$$\eta - \rho = \frac{\gamma - \rho}{n} + \frac{1}{2}\log(n^2 + 2n). \tag{7}$$

Hence, for given γ, η and ρ, the attack is feasible if $\exists n > 0$ satisfying equation 7. For the parameters in [Pereira 2021], the greatest n satisfying the condition is found to be $n = -213$ and, since current technology doesn't allow for negative samples, it follows that the attack cannot be applied in that setting.

However, using the analysis of [Lima Pereira 2020], one has a different story for the same AGCD parameters. Namely, it yields the condition that $\log \delta_0 < \frac{\eta-\rho}{n} - \frac{\gamma}{t^2}$. Maximizing that bound, one finds $n = 272$, so the condition becomes $\log \delta_0 < \frac{5}{272} - \frac{680}{272^2} \approx 0.00919$, which is positive. It might be noted that other papers use slightly different δ_0 estimates for essentialy the same attack (OL), and our results indicate that one may want to pay a closer attention to those estimates, as they may lead to different security estimation.

7. Conclusion

We have analyzed in detail the algorithms to solve the AGCD problem and how they translate to the security of AGCD-based schemes. From this, we have extended the well-known lattice estimator to also support the AGCD problem. Finally, we used our tool to find concrete parameters for existing AGCD-based schemes, since many of them only present asymptotic parameters while others use parameters that turned out not achieving the claimed security level. We hope that our tool will be useful for anyone designing new schemes based on the AGCD problem.

References

Ajtai, M. (1998). The shortest vector problem in ℓ_2 is np-hard for randomized reductions. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC)*, pages 10–19. ACM.

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

Albrecht, M. R. (2017). On dual lattice attacks against small-secret lwe and parameter choices in helib and seal. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 103–129. Springer.

Albrecht, M. R., Bai, S., Fouque, P.-A., Kirchner, P., Stehlé, D., and Wen, W. (2020). Faster enumeration-based lattice reduction: root hermite factor time. In *Annual International Cryptology Conference*, pages 186–212. Springer.

Albrecht, M. R., Bai, S., Li, J., and Rowell, J. (2021). Lattice reduction with approximate enumeration oracles: practical algorithms and concrete performance. In *Annual International Cryptology Conference*, pages 732–759. Springer.

Albrecht, M. R., Ducas, L., Herold, G., Kirshanova, E., Postlethwaite, E. W., and Stevens, M. (2019). The general sieve kernel and new records in lattice reduction. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 717–746. Springer.

Albrecht, M. R., Göpfert, F., Virdia, F., and Wunderer, T. (2017). Revisiting the expected cost of solving usvp and applications to lwe. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 297–322. Springer.

Albrecht, M. R., Player, R., and Scott, S. (2015). On the concrete hardness of learning with errors. *Cryptology ePrint Archive*.

Alkim, E., Ducas, L., Pöppelmann, T., and Schwabe, P. (2016). Post-quantum key {Exchange—A} new hope. In *25th USENIX security symposium (USENIX Security 16)*, pages 327–343.

Aquiles Lima, and Lima Pereira, H. (2026). Agcd estimator. `https://github.com/ericlesaquiles/lattice-estimator/tree/agcd`.

Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Seiler, G., Stehlé, D., et al. (2019). Crystals-kyber algorithm specifications and supporting documentation. *NIST PQC Round*, 2(4):1–43.

Bajard, J.-C., Eynard, J., Hasan, M. A., and Zucca, V. (2017). A full rns variant of fv like somewhat homomorphic encryption schemes. In *Selected Areas in Cryptography (SAC 2016)*, volume 10532 of *Lecture Notes in Computer Science*, pages 423–442. Springer.

Becker, A., Ducas, L., Gama, N., and Laarhoven, T. (2016). New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 10–24. SIAM.

Bellare, M., Desai, A., Jokipii, E., and Rogaway, P. (1997). A concrete security treatment of symmetric encryption. In *Proceedings 38th annual symposium on foundations of computer science*, pages 394–403. IEEE.

Bi, J., Coron, J.-S., Faugère, J.-C., Nguyen, P. Q., Renault, G., and Zeitoun, R. (2014). Rounding and chaining lll: finding faster small roots of univariate polynomial congruences. In *International Workshop on Public Key Cryptography*, pages 185–202. Springer.

Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Seiler, G., and Stehle, D. (2018). Crystals - kyber: A cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367.

Brakerski, Z. and Vaikuntanathan, V. (2011). Efficient fully homomorphic encryption from (standard) lwe. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 97–106.

Chailloux, A. and Loyer, J. (2021). Lattice sieving via quantum random walks. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 63–91. Springer.

Chen, Y. and Nguyen, P. Q. (2011). Bkz 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 1–20. Springer.

Cheon, J. H., Coron, J.-S., Kim, J., Lee, M. S., Lepoint, T., Tibouchi, M., and Yun, A. (2013). Batch fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 315–335. Springer.

Cheon, J. H., Han, K., Kim, A., Kim, M., and Song, Y. (2019). A full rns variant of approximate homomorphic encryption. In *Selected Areas in Cryptography (SAC 2018)*, volume 11349 of *Lecture Notes in Computer Science*, pages 347–368. Springer.

Cheon, J. H. and Stehlé, D. (2015). Fully homomophic encryption over the integers revisited. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 513–536. Springer.

Chillotti, I., Gama, N., Georgieva, M., and Izabachène, M. (2020). Tfhe: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91.

Coron, J.-S., Lepoint, T., and Tibouchi, M. (2014). Scale-invariant fully homomorphic encryption over the integers. In *International Workshop on Public Key Cryptography*, pages 311–328. Springer.

Coron, J.-S., Naccache, D., and Tibouchi, M. (2012). Public key compression and modulus switching for fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 446–464. Springer.

Coron, J.-S. and Pereira, H. V. L. (2019). On kilian's randomization of multilinear map encodings. In Galbraith, S. D. and Moriai, S., editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 325–355, Cham. Springer International Publishing.

Ding, J., Kim, S., Takagi, T., and Wang, Y. (2018). Why 1.02? the root hermite factor of lll and stochastic sandpile models. *arXiv preprint arXiv:1804.03285*.

Ding, J. and Tao, C. (2014). A new algorithm for solving the approximate common divisor problem and cryptanalysis of the fhe based on gacd. *IACR Cryptol. ePrint Arch.*, 2014:42.

Feo, L. D. (2017). Mathematics of isogeny based cryptography.

Galbraith, S. D., Gebregiyorgis, S. W., and Murphy, S. (2016). Algorithms for the approximate common divisor problem. *LMS Journal of Computation and Mathematics*, 19(A):58–72.

Gama, N. and Nguyen, P. Q. (2008). Predicting lattice reduction. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 31–51. Springer.

Guo, Q. and Johansson, T. (2021). Faster dual lattice attacks for solving lwe with applications to crystals. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 33–62. Springer.

Halevi, S. and Shoup, V. (2014). Algorithms in helib. In *Advances in Cryptology – CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 554–571. Springer.

Hermite, C. (1850). Extraits de lettres de m. ch. hermite à m. jacobi sur différents objects de la théorie des nombres. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1850(40):261–278.

Johnson, D., Menezes, A., and Vanstone, S. (2001). The elliptic curve digital signature algorithm (ecdsa). *Int. J. Inf. Secur.*, 1(1):36–63.

Kim, J., Lee, M. S., Yun, A., and Cheon, J. H. (2013). Crt-based fully homomorphic encryption over the integers. *Cryptology ePrint Archive*.

Korkine, A. and Zolotareff, G. (1873). Sur les formes quadratiques. *Mathematische Annalen*, 6(3):366–389.

Korkine, A. and Zolotareff, G. (1877). Sur les formes quadratiques positives. *Mathematische Annalen*, 11(2):242–292.

Laarhoven, T. (2015). Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In *Annual Cryptology Conference*, pages 3–22. Springer.

Laarhoven, T., Mosca, M., and Van De Pol, J. (2015). Finding shortest lattice vectors faster using quantum search. *Designs, Codes and Cryptography*, 77(2):375–400.

Lee, E., Lee, J. W., Kim, Y. S., and No, J.-S. (2022). Optimization of homomorphic comparison algorithm on rns-ckks scheme. *IEEE Access*, 10:26163–26176.

Lima Pereira, H. V. (2020). Homomorphic encryption and multilinear maps based on the approximate-gcd problem.

MATZOV, I. Report on the security of lwe: improved dual lattice attack, 2022. *URL: https://zenodo. org/record/6412487*.

Nguyen, P. and Stern, J. (1997). Merkle-hellman revisited: a cryptanalysis of the qu-vanstone cryptosystem based on group factorizations. pages 198–212.

Peikert, C. (2016). A decade of lattice cryptography. *Found. Trends Theor. Comput. Sci.*, 10(4):283–424.

Pereira, H. V. L. (2020). Efficient agcd-based homomorphic encryption for matrix and vector arithmetic. In *International Conference on Applied Cryptography and Network Security*, pages 110–129. Springer.

Pereira, H. V. L. (2021). Bootstrapping fully homomorphic encryption over the integers in less than one second. In *IACR international conference on public-key cryptography*, pages 331–359. Springer.

van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. (2010). Fully homomorphic encryption over the integers. In *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer.

Xu, J., Sarkar, S., and Hu, L. (2018). Revisiting orthogonal lattice attacks on approximate common divisor problems and their applications. *Cryptology ePrint Archive*.

Zhao, Z. and Ding, J. (2023). Practical improvements on bkz algorithm. In *International Symposium on Cyber Security, Cryptology, and Machine Learning*, pages 273–284. Springer.

Zhao, Z., Ding, J., and Yang, B.-Y. (2024). Bgj15 revisited: Sieving with streamed memory access. *Cryptology ePrint Archive*.