A comprehensive analysis of Regev's quantum algorithm

Razvan Barbulescu, Mugurel Barcau, Vicentiu Pasol

A COMPREHENSIVE ANALYSIS OF REGEV'S QUANTUM ALGORITHM

RAZVAN BARBULESCU[1], MUGUREL BARCAU[2,3], AND VICENŢIU PAŞOL[2,3]

Abstract. Public key cryptography can be based on integer factorization and the discrete logarithm problem (DLP), applicable in multiplicative groups and elliptic curves. Regev's recent quantum algorithm was initially designed for the factorization and was later extended to the DLP in the multiplicative group. In this article, we further extend the algorithm to address the DLP for elliptic curves. Notably, based on celebrated conjectures in Number Theory, Regev's algorithm is asymptotically faster than Shor's algorithm for elliptic curves.

Our analysis covers all cases where Regev's algorithm can be applied. We examine the general framework of Regev's algorithm and offer a geometric description of its parameters. This preliminary analysis enables us to certify the success of the algorithm on a particular instance before running it.

In the case of integer factorization, we demonstrate that there exists an infinite family of RSA moduli for which the algorithm [Reg23] always fails (as opposed to a newer variant of Pilatte [Pil24]). In the case of discrete logarithms, when the parameters align with the Gaussian heuristics, we prove that Regev's algorithm succeeds. By noting that the algorithm naturally adapts to the multidimensional DLP, we proved that it succeeds for a certain range of parameters. We make a precise parameter choice for its implementation to specific NIST-listed elliptic curves. Notably, Bernstein's Curve25519 has small coefficients, which makes it vulnerable to Regev's algorithm.

1. Introduction

The two most used primitives in public key cryptography are integer factorization and discrete logarithm problem (DLP), which arises in two variants: multiplicative groups and elliptic curves. While these problems are not known to be directly linked, the methods employed to analyze them often reflect each other—for instance, Pollard's rho, the number field sieve (NFS), and Shor's quantum algorithm each have adaptations for both problems. However, this correspondence is not complete, as some algorithms do not successfully carry over to elliptic curves. Specifically, for identical group orders, solving the DLP on a classical computer is easier in multiplicative groups than on elliptic curves because index calculus algorithms do not effectively extend well to curves of small genus, in particular to elliptic curves.

In a recent article [Reg23], Regev introduced a quantum algorithm for integer factorization that requires fewer quantum gates than Shor's algorithm. Ekerå and Gärtner [EG23a] later expanded this algorithm to solve discrete logarithms in multiplicative groups, but its elliptic curve variant has remained unexplored. Our research extends this algorithm to elliptic curves, and by analyzing its complexity,

[1]Institut de mathematiques de Bordeaux, CNRS, Universite de Bordeaux, Bordeaux INP, Inria, France

[2]Institute of Mathematics of the Romanian Academy, Romania

[3]CertSIGN S.A., Romania

we demonstrate that its speedup with repect to Shor's algorithm goes to infinity with the size of the input. From the practical perspective, we identify elliptic curves from the NIST list which are vulnerable to our adaptation of Regev's algorithm.

In its original form, Regev's factorization algorithm was based on an assumption about the behavior of the quadratic residuosity of small prime numbers. In a very recent development, Pilatte [Pil24] demonstrated that, with a slight modification to Regev's algorithm, specifically by selecting random primes from a somewhat larger range.

From a cryptographic standpoint, an attacker will generally favor Regev's initial variant due to its complexity of $O(n^{3/2}\log n)$ as opposed to $O(n^{3/2}\log^3 n)$ for Pilatte's variant. Additionally, the hidden constants in Pilatte's variant are larger[1], making it less attractive for practical attacks. This complexity difference can significantly influence the efficiency of factorization attempts in cryptographic applications, reinforcing the preference for the original algorithm.

We identify infinitely many RSA moduli that Regev's variant fails to factor. While Pilatte's modified variant can asymptotically tackle these cases, the question remains open on cryptographic sizes.

We examine the general framework of Regev's algorithm and offer a geometric description of its parameters. This preliminary analysis enables us to certify the success of the algorithm on a particular instance before running it. Extensive experimental testing of Regev's heuristics (e.g., in [EG23b]) indicates that the algorithm is likely to be successful in practice for a significant number of cases and our certification procedure is applicable to a large proportion of these successful instances. This is crucial because when an implementation requires extensive research and development, it is important to eliminate mathematical errors early, allowing the focus to shift to detecting implementation issues.

Moreover, our general framework for Regev's algorithm has enabled us to identify another problem where it can be successfully applied. In 1993, Brands [Bra93] created the first cryptographic application based on an extension of DLP, called multidimensional DLP (see Definition 7, which can have small variations). Over time, the problem gained even greater significance in cryptography, as discussed in [GR09, Section 6]. While there are several variants of the problem, the state-of-the-art attacks remain the same. A natural question is whether the DLP solvers can be extended to multidimensional DLP with the same complexity. Contrary to Pollard's rho algorithm for the DLP, which had an extension to the multidimensional DLP by Galbraith and Ruprai [GR09], Shor's algorithm is not known to have an extension to this problem. We note that Regev's algorithm applies to multidimensional DLP and we find a range of the parameters where one can eliminate the heuristics.

The main idea behind Regev's algorithm can be summarized as follows. Recall that Shor's algorithm is based on the quantum Fourier transform (QFT) and computes in superposition elements of the form g^z where $g \in \mathbb{G}$ and z is an n-bit integer. Regev's algorithm uses $g_1, \ldots, g_d \in \mathbb{G}$ for a parameter $d \geq 2$ and computes in superposition elements of the form $\prod_{i=1}^{d} g_i^{z_i}$ where the z_i are n/d-bit integers; we call this product a multi-scalar product. Note that Regev's algorithm can be viewed as an adaptation of the Shor's algorithm for DLP to the multidimensional case. This is a non-trivial achievement because the hidden subgroup problem can

[1]Pilatte's bound d^{1000d} is larger than 2^n for RSA moduli of less than $0.94 \cdot 10^8$ bits.

be solved using Shor's algorithm in polynomial time only when the dimension d is independent of the size of the problem.

However, Regev's approach goes beyond merely extending Shor's algorithm: its quantum procedure is roughly d times faster for all problems where the algorithms applies successfully. To understand why, recall that on a classical, computer the multi-scalar product can be computed in $O(n/d)$ group operations (see Algorithm 2) compared to $O(n)$ operations for an n-bit scalar product. A key objective of this article is to rigorously demonstrate the speed-up achieved by Regev's algorithm.

Let us now list the problems addressed in this article where Regev's algorithm can be applied, specifying the group $\mathbb{G}$ and the possible values for $g_1, ..., g_d$.

- factorization of an integer N: $\mathbb{G} = \mathbb{Z}_N^*$ and the g_i's are the squares of the first d rational primes;
- DLP in the multiplicative group of a prime field $\mathbb{F}_p$: $\mathbb{G} = \mathbb{F}_p^*$ and the g_i's are the first d rational primes;
- DLP in the multiplicative group of a field $\mathbb{F}_{\ell^n}$ for a small ℓ: $\mathbb{G} = \mathbb{F}_{\ell^n}^*$ and the g_i's are the first d unitary irreducible polynomials;
- multidimensional DLP (see Definition 7) of given parameters $g_1, \ldots, g_d$ and $\mathbb{G}$: there is no choice for the g_i's as they are predetermined;
- DLP on a given elliptic curve E over $\mathbb{F}_p$: $\mathbb{G} = E(\mathbb{F}_p)$ and we show how to search for the g_i's in Section 3;
- DLP with pre-computations (see [EG23a]), i.e. same as DLP but $g_1, \ldots, g_{d-1}$ are given or computed in advance.

1.1. **Regev's algorithm in a nutshell.** Although it can be presented in an independent way, Regev's algorithm can also be seen within the general framework of the CHSP-solvers as described in [EHKS14, dBDF20]. Given a function that hides a lattice of periods $\mathcal{L}$, Regev's algorithm and the CHSP-solver aim to compute $\mathcal{L}$. The two algorithms are composed of the following main steps:

(1) The quantum procedure (Algorithm 1). The quantum Fourier transform (QFT) is used over the domain $[-D/2, D/2]^d$ with coefficients a Gaussian distribution of standard deviation R, producing approximations of elements from the dual lattice $\mathcal{L}^*$, which are randomly sampled from $\mathcal{L}^*/\mathbb{Z}^d$ with a probability that is uniform up to error roughly of order $1/R$;
(2) The post-processing. A classical algorithm is applied to the output of $d+4$ runs of the quantum procedure. This gives a linearly independent set of vectors from $\mathcal{L}$, which generates all vectors with lengths shorter than a specified bound T.

It is important to note that the key difference between the two algorithms is that Regev's algorithm computes only a certain vector in $\mathcal{L}$ which is sufficient for solving the desired problem. In contrast, the CHSP solver computes a full basis of the period lattice.
In a recent paper, Ragavan and Vaikuntanathan [RV23] proposed a different post-processing which can find the correct samples among a large number of samples with errors.

1.2. **Our contribution.** Our main results are as follows:

- Theorem 2.3 proves that Regev's algorithm [Reg23] fails to factor infinitely many RSA moduli; this result is unexpected because a natural heuristics

Algorithm 1 The quantum subroutine of Regev's algorithm

Input: an integer d

Output: a vector of $\mathcal{L}^*/\mathbb{Z}^d$ approximated $O(d+n/d)$ bits of precision, randomly drawn according to the distribution Q below , which is close to uniform.

Set $T = \exp(Cn/d)$ for an absolute constant C. Set $R = 2^{d+n/d}T$ and $D = 2^{\lceil \log_2 R \rceil}$.

Compute the quantum state

$$|\varphi_1\rangle = \sum_{z \in [-D/2, D/2]^d \bigcap \mathbb{Z}^d} \varrho_R(z)|x\rangle|0\rangle$$

then in a multi-scalar manner (Algorithm 2) the state

$$|\varphi_2\rangle = \sum_{z \in [-D/2, D/2]^d} \varrho_R(z)|x\rangle|f(x)\rangle.$$

Measure the register $|f(x)\rangle$ to obtain, for a value $e \in \mathbb{Z}^d$,

$$|\varphi_3\rangle = \sum_{z \in (e+L) \bigcap [-D/2, D/2]^d} \varrho_R(z)|z\rangle.$$

Compute the quantum Fourier transform of $|\psi_3\rangle$ to obtain

$$|\varphi_4\rangle = \sum_{y \in (\mathbb{Z}/D\mathbb{Z})^d} e^{-2i\epsilon \cdot y} \left(\sum_{\ell \in L \bigcap (\frac{1}{D}\mathbb{Z}/\mathbb{Z})^d} e^{-2i\pi \ell \cdot y} \varrho_R(e+\ell) \right) |y\rangle.$$

Measure $|\psi_4\rangle$ to obtain every $y \in \mathcal{L}^*/\mathbb{Z}^d$ with probability $Q(y) = (\det \mathcal{L})^{-1} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} Q_v(y)$, where

$$Q_v(y) = \frac{\varrho_{1/\sqrt{2}R}(v + y + \mathbb{Z}^d)}{\varrho_{1/\sqrt{2}R}(v - D^{-1}\mathbb{Z}^d)}$$

would imply that Regev's algorithm for integer factorization succeeds in all the cases;

- Theorem 2.5 states that if the lattice in Regev's algorithm does not contain vectors shorter than predicted by the Gaussian heuristic, the algorithm will succeed;
- Theorem 2.9 eliminates the heuristics in Regev's algorithm for the multidimensional DLP;
- Fact 3.4 provides a precise estimate of the cost of Regev's algorithm for specific curves on the NIST list; for instance, Bernstein's Curve25519 has small coefficients, which is not a random property of an elliptic curve of its security level but it is a useful property for an attack with Regev's algorithm;
- Theorem 3.6 shows that celebrated conjectures in Number Theory imply an asymptotic speed-up of Regev's algorithm for elliptic curves compared to Shor's algorithm; however the speed-up is much smaller than that achieved in the factorization case.

1.3. **Organization of the paper.** In Section 2.1, we outline the general framework of the Continuous Hidden Subgroup Problem (CHSP), defining the period

lattice and highlighting the presence of a "trivial" period sublattice in Regev's algorithm. We formalize the concept of identifying nontrivial periods, which, in our context, would address the cryptographic challenge. In Section 2.2, we introduce a critical concept utilized in Regev's algorithm—the requirement that the parameter T must exceed what we term the inter-lattice minimal distance, an extension of the successive minima concept. Further, in Section 2.3 we discuss scenarios where Regev's approach to the factorization problem can't succeed. The concept of "Regev-friendly" groups is then explained in Section 2.4 using period lattices, and we demonstrate the conditions under which Regev's algorithm is successful, including an estimation of its complexity. Additionally, we provide a practical criterion for verifying the efficacy of Regev's algorithm when pre-computations are involved. Section 2.5 analyzes the multidimensional DLP, a natural application for Regev's algorithm, where we affirm its effectiveness for specific parameter choices. We also assess the complexity of Regev's algorithm in this multidimensional context and include a comparative analysis with other algorithms which can be used to tackle the multidimensional DLP.

We then pass to the case of elliptic curves. In Section 3.1 one provides two estimates (Proposition 3.1 and Proposition 3.2), demonstrating that a randomly chosen elliptic curve is not suited for a straightforward application of Regev's algorithm due to the absence of "small" points. We then present in Section 3.2 a list of elliptic curves that are suited to Regev's algorithm. For each curve, we detail its rank and the height of the largest linear combination of the Mordell-Weil generators, indicating their susceptibility to Regev's approach. However, it's important to note that the aforementioned curves are not of cryptographic significance. Consequently, we devise in Section 3.3 a strategy to adapt Regev' algorithm to many NIST curves, utilizing quadratic twists to generate small points. Section 3.4 includes an analysis of the computational complexity of Regev's algorithm on elliptic curves. The superior efficiency over Shor's algorithm goes to infinity when the bit size of the elliptic curve goes to infinity.

2. General results on Regev's algorithm: definitions and proofs

2.1. **General set-up.** Let $f : \mathbb{Z}^d \to \mathbb{G}$ be a homomorphism of groups defined by

$$f(z_1, \dots, z_d) = \prod_{i=1}^{d} g_i^{z_i}, \tag{1}$$

where $S = \{g_1, \dots, g_d\}$ is a subset of $\mathbb{G}$. Then the *period lattice* of f is the set

$$\mathcal{L} = \mathcal{L}(\mathbb{G}, S, f) := \left\{ (z_1, \dots, z_d) \in \mathbb{Z}^d \;\middle|\; f(z_1, \dots, z_d) = 1_{\mathbb{G}} \right\}, \tag{2}$$

where $1_{\mathbb{G}}$ is the neutral element of $\mathbb{G}$.

In practical cryptographic contexts involving the CHSP, a "trivial" sublattice $\mathcal{L}_0$ is typically identified, defined by the "obvious" relations upheld by elements of S. Solving CHSP, i.e. finding a basis for the period lattice, finds in particular a 'nontrivial period,' which involves identifying a vector in $\mathcal{L} \setminus \mathcal{L}_0$. This often resolves the cryptographic challenge associated to the CHSP. This paper examines in particular two specific cryptographic challenges which fit in the framework of the CHSP: the factorization problem and the discrete logarithm problem. Regev's

algorithm represents a pioneering approach in this area, as it can identify a period in $\mathcal{L} \setminus \mathcal{L}_0$ without necessarily computing a basis for the period lattice $\mathcal{L}$. Note that the preceding discussion can be framed as the following general lattice problem:

Nontrivial Period Problem: Given $f : \mathbb{Z}^d \to X$ a function whose set of periods:

$$\mathcal{L} := \mathcal{L}(f) := \{z \in \mathbb{Z}^d \mid f(a+z) = f(a)\ \forall a \in \mathbb{Z}^d\}$$

is a full-rank sublattice of $\mathbb{Z}^d$ and a sublattice $\mathcal{L}_0$ of $\mathcal{L}$ (given possibly by an oracle), find any $z \in \mathcal{L} \setminus \mathcal{L}_0$.

A stronger statement of the Nontrivial Period Problem would ask to find the shortest vector in $\mathcal{L} \setminus \mathcal{L}_0$.

Given that we are focusing on two specific instances of the general setup described above, we will present them separately.

The case of integer factorization. In this scenario, N is an m-bit integer that we aim to factorize and $g_1, \ldots, g_d$ are some small $O(\log d)$-bit integers (for instance, let p_i be the i^{th}-prime) and $g_i = p_i^2$. Then, we consider the lattice

$$\mathcal{L} = \left\{ (z_1, \ldots, z_d) \in \mathbb{Z}^d \;\middle|\; \left(\prod_{i=1}^{d} p_i^{z_i} \right)^2 \equiv 1 \pmod{N} \right\} \tag{3}$$

and its sublattice

$$\mathcal{L}_0 := \left\{ (z_1, \ldots, z_d) \in \mathbb{Z}^d \;\middle|\; \prod_{i=1}^{d} p_i^{z_i} \equiv -1 \text{ or } 1 \pmod{N} \right\}. \tag{4}$$

If one puts $g_i = p_i^2$ then $\mathcal{L} = \{z \in \mathbb{Z}^d \mid \prod_{i=1}^{d} g_i^{z_i} \equiv 1 \pmod{N}\}$. Any element of $\mathcal{L}$ outside $\mathcal{L}_0$ allows to find a non-trivial factor of N by computing $\gcd(N, \prod_{i=1}^{d} g_i^{z_i} - 1)$. The oracle which describes $\mathcal{L}_0$ consists in computing the previous gcd: a vector $z \in \mathbb{Z}^d$ is outside $\mathcal{L}_0$ if and only if it gives a factor of N other than 1 and N.

The case of discrete logarithms. Here $\mathbb{G}$ is a commutative group, $\mathbb{H}$ is a subgroup of $\mathbb{G}$ of order r, where r is an n-bit prime. We denote by m the number of bits needed to represent an element of $\mathbb{G}$. If g is a generator of the subgroup $\mathbb{H}$, then the discrete logarithm problem asks to compute for any $x \in \mathbb{H}$ the smallest $e \in \mathbb{Z}_{\geq 0}$ such that $x = g^e$. The difficulty of computing discrete logarithms is known to depend on $n = 1 + \lfloor \log_2 r \rfloor$ and m. The most important examples are $\mathbb{G} = \mathbb{Z}_p^*$ and $\mathbb{G} = E(\mathbb{Z}_p)$ for an elliptic curve E defined over $\mathbb{Z}_p$, where p is a prime. In most cases, r is chosen to be as large as possible so that $n \approx \log_2 |\mathbb{G}|$.

Now, consider a set $S = \{g_1 = g, g_2, \ldots, g_{d-1}, g_d = x\}$ of elements of $\mathbb{H}$, where g is a generator, as above. If $f : \mathbb{Z}^d \to \mathbb{G}$ is the map $f(z_1, \ldots, z_d) = \prod_{i=1}^{d} g_i^{z_i}$, so that the *period lattice* of f is the set

$$\mathcal{L} = \mathcal{L}(\mathbb{G}, S) := \left\{ (z_1, \ldots, z_d) \in \mathbb{Z}^d \;\middle|\; \prod_{i=1}^{d} g_i^{z_i} = 1_{\mathbb{G}} \right\}, \tag{5}$$

where $1_{\mathbb{G}}$ is the neutral element of $\mathbb{G}$. Consider the sublattice

$$\mathcal{L}_0 = \{(z_1, \ldots, z_d) \in \mathcal{L} \mid z_d \equiv 0 \pmod{r}\}. \tag{6}$$

Given that g is a generator, for any vector $(z_1, \ldots, z_d)$ of $\mathcal{L}$, the following holds:

$$\sum_{i=1}^{d} z_i \log_g g_i \equiv 0 \pmod{r}.$$

For any $(z_1, ..., z_d) \in \mathcal{L} \setminus \mathcal{L}_0$ we have:

$$\log_g x \equiv -z_d^{-1}(\sum_{i=1}^{d-1} z_i \log_g g_i) \pmod{r}. \tag{7}$$

Consequently, the solution to the discrete logarithm problem is found. The case of the multiplicative group was treated in [EG23a] using $g_i = p_i$, the i-th rational prime. The oracle which describes $\mathcal{L}_0$ is as follows: a vector $z \in \mathbb{Z}^d$ is outside $\mathcal{L}_0$ if and only if one fails to compute $\log_g z$ using the equation above or equivalently if $z_d \equiv 0 \pmod{r}$.

The case of multidimensional DLP.. The setup is identical to the DLP except that d and $g_1, \ldots, g_d$ cannot be chosen and must be assumed small. The parameters n is $\log_2 N$ instead of $\log_2 r$ for a prime factor r of $|\mathbb{G}|$. The lattices $\mathcal{L}$ and $\mathcal{L}_0$ are identical to those of the DLP, i.e. given by Equations (5) and (6).

Final notations of the setup. Let us recall that the dual lattice of $\mathcal{L}$ is defined as

$$\mathcal{L}^* = \{y \in \mathbb{R}^d \mid \forall x \in \mathcal{L}, xy \in \mathbb{Z}\}.$$

Also, if B is a generating matrix of $\mathcal{L}$ then $(B^{-1})^T$ is a generating matrix of $\mathcal{L}^*$. Additionally, for a set of integers $\{a_2, \ldots, a_d\}$ we set[2]

$$\text{knapsack}(r, \{a_2, \ldots, a_d\}) = \begin{pmatrix} r & a_2 & \ldots & a_d \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & 1 \end{pmatrix}. \tag{8}$$

Note that, in the discrete logarithm case, $\mathcal{L}$ is generated by the columns of $\text{knapsack}(r, \{a_2, \ldots, a_d\})$, where $a_i = -\log_g g_i$.

The lattice $\mathcal{L}$ is the main object in several quantum algorithms. In Shor's algorithm for factoring the number N, the group used is $\mathbb{G} = \mathbb{Z}_N^*$ with $d = 1$, whereas for computing the discrete logarithm the group is $\mathbb{G} = \mathbb{Z}_p^*$ with $d = 2$. In quantum algorithms for computing unit and class groups, and more generally in the quantum algorithms for solving the CHSP [EHKS14, dBDF20, BS16], d increases with the input size n. Finally, in Regev's algorithm for factoring N one uses $\mathbb{G} = \mathbb{Z}_N^*$ and $d = \lceil \sqrt{n} \rceil$, whereas in its discrete logarithm variant [EG23a] the group $\mathbb{G} = \mathbb{Z}_p^*$ is employed with the same number $d = \lceil \sqrt{n} \rceil$. The parameter d is chosen to optimize the number of quantum gates in Regev's algorithm, while the set S is selected to enable efficient computation of any element of the form $\prod_{i=1}^{d} g_i^{\varepsilon_i}$, for all tuples $(\varepsilon_1, \ldots, \varepsilon_d) \in \{0, 1\}^d$.

[2]This type of lattice was also studied in the context of Merkle and Hellman's knapsack cryptosystem (see e.g. Equation (2.4) in [LO85]).

2.2. **On inter-lattice minimal distance.** In this subsection, we introduce the concept of *inter-lattice minimal distance*, which quantifies the Euclidean distance between a sublattice and its complement. This proves to be the appropriate measure for estimating the success of Regev's algorithm.

For $\mathcal{L} \subset \mathbb{Z}^d$ a lattice of rank d and $\mathcal{L}_0$ a sublattice of $\mathcal{L}$ (not necessarily of the same rank), we define:

$$\lambda_{\mathcal{L}_0}(\mathcal{L}) := \inf\{T > 0 | B(T) \cap \mathcal{L} \neq B(T) \cap \mathcal{L}_0\} = \inf\{\|v\| \mid v \in \mathcal{L} \setminus \mathcal{L}_0\} = d(\mathcal{L}_0, \mathcal{L} \setminus \mathcal{L}_0),$$

where $B(T)$ is the closed ball of radius T in $\mathbb{R}^d$, and $d(A, B)$ is the usual Euclidian distance between two subsets A and B of $\mathbb{R}^d$. By convention, if a T satisfying the above condition does not exist, we set $\lambda_{\mathcal{L}_0}(\mathcal{L}) = +\infty$. Since $\mathcal{L}$ is discrete with respect to the Euclidian topology in $\mathbb{R}^d$, both infima, when finite, are actually minima.

Definition 1. *Consider a lattice $\mathcal{L}$ in $\mathbb{Z}^n$ and a sublattice $\mathcal{L}_0 \subseteq \mathcal{L}$. The closure of $\mathcal{L}_0$ in $\mathcal{L}$ is defined as the sublattice $cl_{\mathcal{L}}(\mathcal{L}_0) := \mathbb{Q} < \mathcal{L}_0 > \cap \mathcal{L}$ of $\mathcal{L}$, where $\mathbb{Q} < \mathcal{L}_0 >$ is the $\mathbb{Q}-$vector space generated by $\mathcal{L}_0$ inside the ambient $\mathbb{Q}-$vector space. The sublattice $\mathcal{L}_0$ is called* closed *in $\mathcal{L}$ if $cl_{\mathcal{L}}(\mathcal{L}_0) = \mathcal{L}_0$.*

We mention the following easy remark:

Remark 1. *The sublattice $\mathcal{L}_0$ is closed in $\mathcal{L}$ if and only if for all $v \in \mathcal{L} \setminus \mathcal{L}_0$ the lattice spanned by $\mathcal{L}_0$ and v has rank $rk(\mathcal{L}_0) + 1$, i.e. v does not belong to the $\mathbb{Q}-$vector space generated by $\mathcal{L}_0$.*

As usual, we denote the successive minima of $\mathcal{L}$ by $\lambda_1(\mathcal{L}), \ldots, \lambda_d(\mathcal{L})$. The following result expresses successive minima in terms of inter-lattice minimal distance.

Proposition 2.1. *For any $i \in [1, d]$, $\lambda_i(\mathcal{L})$ is the supremum of $\lambda_{\mathcal{L}_0}(\mathcal{L})$, where $\mathcal{L}_0$ ranges over the sublattices of rank at most $i - 1$. Moreover, the supremum is a maximum, i.e. there exists $\mathcal{L}_0$ a sublattice of $\mathcal{L}$ of rank $i - 1$ such that $\lambda_i(\mathcal{L}) = \lambda_{\mathcal{L}_0}(\mathcal{L})$.*

Proof. Let $\mathcal{L}_0$ be a sublattice of rank at most $i - 1$. Since there exist i linearly independent vectors in $\mathcal{L}$ of length at most $\lambda_i(\mathcal{L})$, at least one of these vectors is not in $\mathcal{L}_0$. For such a vector v, it follows that $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \|v\| \leq \lambda_i(\mathcal{L})$.

Now, let $v_1, ..., v_i$ be linearly independent vectors in $\mathcal{L}$ such that $\|v_j\| = \lambda_j(\mathcal{L})$ for all $j \leq i$, and denote by $\mathcal{L}_0$ the closure in $\mathcal{L}$ of the lattice generated by $v_1, ..., v_{i-1}$. Choose a vector $v \in \mathcal{L} \setminus \mathcal{L}_0$ such that $\|v\| = \lambda_{\mathcal{L}_0}(\mathcal{L})$. Since $\mathcal{L}_0$ is closed in $\mathcal{L}$, by Remark 1, the set $v_1, ..., v_{i-1}, v$ is linearly independent, so that $\|v\| \geq \lambda_i(\mathcal{L})$. Thus, we have $\lambda_{\mathcal{L}_0}(\mathcal{L}) \geq \lambda_i(\mathcal{L})$. On the other hand, since $v_1, ..., v_i$ are linearly independent vectors, we get that $v_i \in \mathcal{L} \setminus \mathcal{L}_0$. In particular $\lambda_i(\mathcal{L}) = \|v_i\| \geq \lambda_{\mathcal{L}_0}(\mathcal{L})$, which proves that $\lambda_i(\mathcal{L}) = \lambda_{\mathcal{L}_0}(\mathcal{L})$. □

We now list some useful properties of the inter-lattice minimal distance.

Proposition 2.2. *Let $\mathcal{L}$ be a full-rank lattice in $\mathbb{Z}^d$, and $\mathcal{L}_0$ be a sublattice of $\mathcal{L}$. We have:*

(1) $\mathcal{L}_0 = \mathcal{L}$ if and only if $\lambda_{\mathcal{L}_0}(\mathcal{L}) = \infty$.
(2) If $\lambda_1(\mathcal{L}_0) > \lambda_1(\mathcal{L})$ then $\lambda_{\mathcal{L}_0}(\mathcal{L}) = \lambda_1(\mathcal{L})$.
(3) If $\mathcal{L}_0'$ is a sublattice of $\mathcal{L}$ with $\mathcal{L}_0 \subseteq \mathcal{L}_0' \subseteq \mathcal{L}$, then $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \lambda_{\mathcal{L}_0'}(\mathcal{L})$.
(4) Let $v \in \mathcal{L}_0^{\perp}$ and let $\mathcal{L}_0'$ be the sublattice generated by $\mathcal{L}_0$ and v. If $\|v\| \geq \lambda_{\mathcal{L}_0'}(\mathcal{L})$, then $\lambda_{\mathcal{L}_0}(\mathcal{L}) = \lambda_{\mathcal{L}_0'}(\mathcal{L})$.

(5) *If* $\lambda_{\mathcal{L}_0}(\mathcal{L}) > \lambda_d(\mathcal{L})$*, then* $\lambda_i(\mathcal{L}_0) = \lambda_i(\mathcal{L})$ *for all* $i \leq d$*.*
(6) *Let* $\mathcal{L}_0 \subseteq \mathcal{L}$ *be a sublattice such that* $[\mathcal{L} : \mathcal{L}_0] > d!$*, then* $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \lambda_d(\mathcal{L})$*.*

Proof. (1) It is a tautology.
(2) Since $B(\lambda_1(\mathcal{L})) \bigcap \mathcal{L}_0 = \emptyset$ while the same ball contains the shortest vector of $\mathcal{L}$, it follows that $\lambda_{\mathcal{L}_0}(\mathcal{L}) = \lambda_1(\mathcal{L})$.
(3) Indeed, if $w \in \mathcal{L} \setminus \mathcal{L}'_0$ is such that $\|w\| = \lambda_{\mathcal{L}'_0}(\mathcal{L})$, then clearly $w \notin \mathcal{L}_0$ so $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \|w\| = \lambda_{\mathcal{L}'_0}(\mathcal{L})$.
(4) Since $\mathcal{L}_0 \subseteq \mathcal{L}'_0$, the preceding result implies that $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \lambda_{\mathcal{L}'_0}(\mathcal{L})$. Let $w \in \mathcal{L} \setminus \mathcal{L}_0$ be such that $\|w\| = \lambda_{\mathcal{L}_0}(\mathcal{L})$. If $w \notin \mathcal{L}'_0$, then we have $\lambda_{\mathcal{L}'_0}(\mathcal{L}) \leq \|w\| = \lambda_{\mathcal{L}_0}(\mathcal{L})$, which proves our result. Otherwise, consider $w = w_0 + a \cdot v \in \mathcal{L}'_0$, where $w_0 \in \mathcal{L}_0$ and $a \in \mathbb{Z} \setminus \{0\}$. Then $\|w\|^2 = \|w_0\|^2 + a^2\|v\|^2 \geq \|v\|^2 \geq \lambda_{\mathcal{L}'_0}(\mathcal{L})^2$, which also proves the claim.
(5) Assume by contradiction that $\lambda_i(\mathcal{L}_0) > \lambda_i(\mathcal{L})$ for some i, and consider a linearly independent set of vectors $v_1, ..., v_i$ in $\mathcal{L}$ such that $\|v_1\| \leq ... \leq \|v_i\| = \lambda_i(\mathcal{L})$. Then there exists $j \leq i$ with $v_j \in \mathcal{L} \setminus \mathcal{L}_0$, so that $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \|v_j\| \leq \lambda_i(\mathcal{L}) \leq \lambda_d(\mathcal{L})$.
(6) Assuming, for the sake of contradiction, that $\lambda_{\mathcal{L}_0}(\mathcal{L}) > \lambda_d(\mathcal{L})$, we derive from the previous result that $\lambda_i(\mathcal{L}_0) = \lambda_i(\mathcal{L})$ for all $i \leq d$. By the second Minkowski theorem we have that

$$\frac{2^d}{d!} \operatorname{Vol}(\mathcal{L}) \leq \operatorname{Vol}(B(1)) \cdot \prod_i \lambda_i(\mathcal{L}) = \operatorname{Vol}(B(1)) \cdot \prod_i \lambda_i(\mathcal{L}_0) \leq 2^d \operatorname{Vol}(\mathcal{L}_0).$$

This leads to a contradiction by using the relation $\frac{\operatorname{Vol}(\mathcal{L})}{\operatorname{Vol}(\mathcal{L}_0)} = [\mathcal{L} : \mathcal{L}_0]$.

□

2.3. **A brief discussion about the factorization case.**

In this section, we demonstrate that a modification of Regev's initial algorithm, such as that proposed by Pilatte, is necessary by presenting a family of RSA moduli that violate Regev's assumption. Unfortunately, we currently lack a method to address these moduli within the cryptographic range while maintaining the same complexity. First, we introduce the concept of admissible integers, defined as those integers for which Regev's initial algorithm is proven to succeed and achieve the desired complexity (see [Reg23, Th 1.1]). The formal definition is as follows:

Definition 2. *Let* c *and* K *be positive constants. An* n*-bit integer* N *is admissible with respect to* c *and* K *if, for* $d = \lfloor \sqrt{n} \rfloor$*, there exist* $c \log_2 n$*-bit integers* $g_1, \ldots, g_d$ *such that the lattices* $\mathcal{L}$ *and* $\mathcal{L}_0$ *defined in Equations* (3) *and* (4) *satisfy* $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \exp(K\sqrt{n})$*.*

Regev's algorithm achieves the desired complexity when factoring integers N that are admissible with respect to two constants c and K. Note however that the constants don't play any role because they are hidden in the big-Oh of the complexity.

For every prime p, let n_p be the least non-quadratic residue modulo p. In a footnote on page 3 of [Reg23], Regev mentions that the proven upper bounds on n_p are $O((\log p)^2)$, which is insufficient for achieving a speed-up compared to Shor's algorithm

Let us elaborate this discussion. It is not clear whether the heuristics states that the set of non-admissible integers is either negligible or empty. In simple words

the heuristics is that the integers $g_1, \ldots, g_d$ which are taken to be small behave as randomly chosen integers in the interval $[1, N]$. In particular, it states that not all of $g_1, \ldots, g_d$ are squares.

More precisely the footnote is as follows: in a RSA moduli $N = PQ$ the values $\log_2 P$, $\log_2 Q$ and $\log_2 N$ are equal up to a multiplicative constant. Regev's algorithm uses $d = \lfloor \sqrt{\log_2 N} \rfloor \approx \sqrt{\log_2 P} \approx \sqrt{\log_2 Q}$ primes $g_1, \ldots, g_d$. It is necessary to prove that $n_P = \tilde{O}(\sqrt{\log_2 P})$. Since in the literature one has $n_p = O((\log_2 P)^2)$, we can't prove that $g_1, \ldots, g_d$ are not all squares.

We noted that the literature contains results to show that not all integers are admissible, i.e. Regev's algorithm doesn't succeed to factor some integers. Indeed, Salié [Sal49] and Fridlender [Fri49] independently proved that there are infinitely many primes P such that $n_P > C \log_2 P$ for a constant C. More recently, Lau and Wu [LW08] proved that such primes are numerous enough so that one can multiply two of them together to obtain RSA moduli, i.e. the primes have aproximatively the same bit size.

We widen the definition and call RSA modulus an integer of the form $N = PQ$ such that $Q^{1/2} \leq P \leq Q$. Indeed, such integers cannot be factored with the algorithms which take advantage of small factors, namely $p-1$, $p+1$ and ECM. Hence they can only be factored with algorithms of the same cost as the most restricted definition of RSA moduli: integers of the form $N = PQ$ where P and Q have the same bit size.

Theorem 2.3. *Infinitely many RSA moduli are not admissible.*

Proof. By a result of Lau and Wu (Proposition A.1 in the appendix), there exist a constant $c > 0$ and a sequence $Q_m \to \infty$ such that for every m, the number of primes P such that $\frac{1}{2} \log_2 Q_m \leq \log_2 P \leq \log_2 Q_m$, $P \equiv 3 \pmod 4$ and $n_P > c \log_2 P$ goes to infinity. In particular, for every sufficiently large m there exist two primes $P < Q$ in the interval $[Q_m^{1/2}, Q_m]$ satisfying

$$n_P > c \log_2 P \text{ and } n_Q > c \log_2 Q.$$

We define $N_m = PQ$ which is an RSA moduli because the quotient $\log_2 Q / \log_2 P \leq 2$.

We claim that for large enough m we have $\mathcal{L}_0 = \mathcal{L}$ for the lattices defined in Equations (3) and (4). After proving the claim, we apply point 1 of Proposition 2.2 to conclude that N is not admissible.

To prove the claim, we first note that $p_1, \ldots, p_d$ are squares in $\mathbb{Z}/N\mathbb{Z}^*$. Chebyshev proved that $p_d \leq Cd \log d$ for an explicit constant C. We have that $n := \log_2 N = \log_2 P + \log_2 Q < 3 \log_2 P$ and, since m is large enough,

$$p_d \leq Cd \log d \leq C\sqrt{n} \log n/2 < cn/3 < c \log_2 P.$$

By the construction of P and Q we have further that $p_d < n_P$ and $p_d < n_Q$, so the first d rational primes, i.e. $p_1, \ldots, p_d$, are squares in $\mathbb{Z}/P\mathbb{Z}^*$ and $\mathbb{Z}/Q\mathbb{Z}^*$.

Let us now prove that $\mathcal{L} \subset \mathcal{L}_0$. For this let $(z_1, \ldots, z_d)$ be such that $\prod_{i=1}^d (p_i^2)^{z_i} \equiv 1 \pmod N$. Since $P \equiv 3 \pmod 4$ (respectively $Q \equiv 3 \pmod 4$), the order of $\mathbb{Z}/P\mathbb{Z}^*$ (respectively $\mathbb{Z}/Q\mathbb{Z}^*$) is 2 times an odd number. Since $p_1, \ldots, p_d$ are squares, their orders are odd in $\mathbb{Z}/P\mathbb{Z}^*$ and $\mathbb{Z}/Q\mathbb{Z}^*$. Therefore, the order of $g = \prod_{i=1}^d p_i^{z_i}$ is odd in $\mathbb{Z}/P\mathbb{Z}^*$ and $\mathbb{Z}/Q\mathbb{Z}^*$, and also in $\mathbb{Z}/N\mathbb{Z}^*$. Since g has odd order in $\mathbb{Z}/N\mathbb{Z}^*$ and $g^2 \equiv 1 \pmod N$, we conclude that $g \equiv 1 \pmod N$, so $(z_1, \ldots, z_d) \in \mathcal{L}_0$. □

2.3.1. *Moduli that present challenges for Regev's algorithm.* Contrary to Shor's algorithm which applies to all RSA moduli PQ, the performance of Regev's algorithm depends on the size of the least non-quadratic residues n_P and n_Q. As a countermeasure to Regev's algorithm, this suggests to select RSA moduli PQ with large n_P and n_Q.

Nonetheless, two significant drawbacks arise. First, there is no known efficient method to find such primes; for instance, Table 1 in [MT19] provides data but lacks an algorithm for rapid computation. Second, Paszkiewicz [Pas09] proved that the proportion of primes P such that $n_P > p_k$ is $1/2^k$. When factoring RSA moduli of n bits, one must take $d = \sqrt{n}$, which means the proportion of suitable primes P is $1/2^{\sqrt{n}}$. This severely limits the number of cryptographic keys that can resist Regev's algorithm.

2.4. **Regev-friendly groups.** In this section, we introduce the concept of a Regev pseudo-basis, which is essential for understanding the structure and properties of Regev-friendly groups. Specifically, we define a Regev pseudo-basis with respect to a parameter T as follows:

Definition 3. *Let $\mathcal{L}$ be a full-rank lattice in $\mathbb{Z}^d$ and let $T > 0$ be a parameter. A Regev pseudo-basis with respect to T is a liniarly independent set $\{z_1, \ldots, z_k\} \subset \mathcal{L}$ for some $k \leq d$ such that:*

(1) $\| z_i \| \leq \sqrt{d}\, 2^{d/2}\, T$ for all $i \leq k$,
(2) every $z \in \mathcal{L}$ of norm less than T is a linear combination with integer coefficients of $z_1, \ldots, z_k$.

It is important to recall that there is an efficient classical algorithm that given a basis of a lattice $\mathcal{L}$ and a norm bound $T > 0$, outputs a Regev pseudo-basis of $\mathcal{L}$ with respect to T (see Claim 5.1 in [Reg23]). In particular, a Regev pseudo-basis exists for any $\mathcal{L}$ and any norm bound $T > 0$. Let us note that $\lambda_{k+1}(\mathcal{L}) \geq T$, where by convention $\lambda_{d+1}(\mathcal{L}) = +\infty$; however, this fact will not be utilized in the article. To see why, assume by contradiction that there exists a rank $k+1$ sublattice $\mathcal{L}'$ of $\mathcal{L}$ with a basis consisting of vectors shorter than T. Since, by definition, a Regev pseudo-basis $z_1, \ldots, z_k$ generates this basis of $\mathcal{L}'$, it must also generate $\mathcal{L}'$. This is a contradiction, as k vectors cannot span a lattice of rank $k+1$.

We will show that when the Regev pseudo-basis is a full basis, the algorithm benefits from increased speed and the elimination of heuristics. This motivates the following definition:

Definition 4. *Let $K > 0$ be a constant. We say that a full-rank lattice $\mathcal{L}$ of $\mathbb{Z}^d$ is K-balanced if it has a basis in which all vectors have length less than $T = \exp(Kn/d)$, where n is the binary size of* $\operatorname{Vol}\mathcal{L}$.

Since the lattices encountered in Regev's algorithm are knapsack lattices (see Equation (8)), we are interested in how frequently these lattices are $K-$balanced. To investigate this, we conducted an experiment using Sage, where we considered $n-$bit random integers r for $n = 50, 75, \ldots, 200$. We also chose d random residues $a_1, \ldots, a_d$ modulo r for $d = 20$, 30 and 40. Table 1 summarizes the percentage of trials in which the knapsack lattice $\mathcal{L} :=\text{knapsack}(r, \{a_1, \ldots, a_d\})$ was 1-balanced. One notices that when $n \gg d$ a large proportion of such knapsack lattices are 1-balanced. Since in Regev's algorithm $d \approx \sqrt{n}$, the experiments indicate that nearly all lattices employed in Regev's algorithm are 1-balanced.

d \ n	50	75	100	125	150	175	200
20	100%	100%	100%	100%	100%	100%	100%
30	19%	100%	100%	100%	100%	100%	100%
40	0%	0%	37%	98%	100%	100%	100%

TABLE 1. Proportion of randomly chosen knapsack lattices which are 1-balanced.

Lemma 2.4. *If a lattice $\mathcal{L}$ is K-balanced then any Regev pseudo-basis with respect to $T = \exp(Kn/d)$ is a basis.*

Proof. Let $z_1, \ldots, z_k$ be a Regev pseudo-basis with respect to $T = \exp(Kn/d)$. Since these vectors span all vectors of length less than T and any basis is composed of such vectors, $\{z_1, \ldots, z_d\}$ therefore form a basis. □

Determining whether a lattice $\mathcal{L}$ is $K-$balanced is generally challenging. However, the following result establishes a connection between this property and another property that is comparatively easier to verify.

Theorem 2.5. *Any full-rank lattice $\mathcal{L}$ in $\mathbb{Z}^d$ with volume $\mathrm{Vol}(\mathcal{L}) = r$ (where r is an n-bit integer) and $\lambda_1(\mathcal{L}) \geq r^{1/d}$ admits a basis $\{v_1, \ldots, v_d\}$ that satisfies*

$$\max_i \| v_i \| \leq 2^{\left((d-1)\log(d)+\frac{n}{d}\right)}.$$

Proof. Indeed, apply Mahler's Theorem on lattices (see [TV06, Theorem 3.34]) to show that it exists a basis $v_1, \ldots, v_d$ for $\mathcal{L}$ such that $|v_1| = \lambda_1$ and $v_i \leq \frac{i\lambda_i}{2}$ for all $2 \leq i \leq d$. In particular, this basis satisfies the bound $\max_i \|v_i\| \leq \frac{d\lambda_d}{2}$. We get by Minkowski Second Theorem that:

$$\max_i \|v_i\| \leq \frac{d\lambda_d}{2} \leq \frac{d}{2}\frac{\prod_i \lambda_i}{\lambda_1^{d-1}} \leq \frac{d2^{d-1}r}{r^{(d-1)/d}} = d2^{d-1}r^{1/d} = 2^{\left((d-1)\log(d)+\frac{n}{d}\right)}.$$

□

Corollary 2.6. *Consider a full-rank lattice $\mathcal{L}$ in $\mathbb{Z}^d$ with volume $\mathrm{Vol}(\mathcal{L}) = r$, where r is an n-bit integer, $\lambda_1(\mathcal{L}) \geq r^{1/d}$. If $d \leq n^{1/2-\delta}$ such that $\delta \geq \frac{\log\log(n)-K_0}{2\log(n)}$, where $K_0 = \log(2\log(e)-2) \simeq -0.1756$, the lattice $\mathcal{L}$ is $1-$ballanced.*

Proof. The condition $\delta \geq \frac{\log\log(n)-K_0}{2\log(n)}$ implies the inequality:

$$d\log(d) + \frac{n}{d} < \frac{n}{d}\log(e).$$

Since $d - 1 < d$, by applying Theorem 2.5, we get that the lattice $\mathcal{L}$ is indeed 1-ballanced. □

Remark 2. *In* [EG23a, Section 2.3] *one discusses a bad choice of the generators: $g_i = g^i$ for $i = 1, \ldots, d$. It is a natural example of a lattice $\mathcal{L}$ with many small vectors:*

$$(0, \ldots, j, \ldots, -i, \ldots, 0), \quad \forall\, i < j$$

where the j value is in the i-th coordinate and the $-i$ value in the j-th coordinate. The authors conclude that $\lambda_n(\mathcal{L})$ is too large and makes Regev's algorithm slow.

In short, they had the intuition that the existence of small vectors makes Regev's algorithm slow. Theorem 2.5 shows that the converse is also true: the absence of small vectors in $\mathcal{L}$ makes Regev's algorithm succeed and have a better complexity than Shor.

We are now prepared to introduce Regev-friendly groups.

Definition 5. *A group $\mathbb{G}$ is K-Regev-friendly with respect to a function $d = d(n)$ if there exists a generating set $S = \{g_1, \dots, g_d\}$ such that the following conditions are satisfied:*

(1) The lattice $\mathcal{L} = \mathcal{L}(\mathbb{G}, S)$ defined in Equation (5) is K-balanced.

(2) One can compute in classical time $o(n)$ the product $\prod_{i=1}^{d} g_i^{\varepsilon_i}$ for any tuple $(\varepsilon_1, \dots, \varepsilon_d) \in \{0,1\}^d$.

For instance, when $\mathbb{G} = \mathbb{Z}/N\mathbb{Z}^*$ and S consists of the smallest d primes for a parameter $d \leq \sqrt{n}$,, then the second condition of the definition above holds. Let $p_1, \dots, p_d$ be these primes. The discussion after Equation (5) in [Reg23] shows that one can compute in time $\tilde{O}(d)$ any product $\prod_{i=1}^{d} p_i^{\varepsilon_i}$ with $\varepsilon_i = 0$ or 1.

2.4.1. *Regev's algorithm is fast in Regev-friendly groups.* We recall the main result about the complexity and correctness of Regev's algorithm. We reproduce here the proofs from the literature in order to distinguish the parameters m and n, as well as letting d have any value less than $\sqrt{n}$.

Proposition 2.7 (Th 1.1 in [Reg23], Lemma 5 in [EG23a])**.** *Let $\mathbb{G}$ be a commutative group of cardinality r, and let n be the bit size of r. Assume that the elements of $\mathbb{G}$ can be represented using $O(m)$-bits, and the group operation can be done in classical $M(m)$-bit operations. We consider $u_1, \dots, u_k$ elements in $\mathbb{G}$, where $k = O(1)$. Let $d \leq \sqrt{n}$ be a parameter, and consider $g_1, ..., g_{d-k}$ elements of $\mathbb{G}$ such that we are given an algorithm, called tab, that computes $\mathrm{tab}(\varepsilon_1, \dots, \varepsilon_{d-k}) = \prod_{i=1}^{d-k} g_i^{\varepsilon_i}$ for any $(\varepsilon_1, \dots, \varepsilon_{d-k}) \in \{0,1\}^{d-k}$ in $\tilde{O}(d)$ classical operations. Then we have:*

(1) Each run of Regev's algorithm (see Algorithm 1) uses $(d + \frac{n}{d})M(m)$ quantum gates.

(2) Assuming that $\mathbb{G}$ is K-Regev-friendly for a constant $K > 0$, Regev's algorithm succeeds in finding the discrete logarithms in $\mathbb{G}$ with probability $\geq 1/4$. The algorithm uses $d + 4$ independent runs of the quantum subroutine (Algorithm 1), followed by a classical post-processing step that runs in polynomial time in m.

(3) Let $G = (\mathbb{Z}/N\mathbb{Z})^*$ with N a composite integer, and consider the lattices $\mathcal{L}$ and $\mathcal{L}_0$ as defined in equations (3) and (4). Assuming that $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq \exp(K\sqrt{n})$ for a constant $K > 0$ (i.e. N is admissible), then Regev's algorithm succeeds in factoring N with probability $\geq 1/4$. The algorithm uses $d+4$ independent runs of the quantum subroutine (Algorithm 1), followed by a classical post-processing step that runs in polynomial time in $m = \log_2 N$.

Proof. 1. We recall the parameters used in the Regev's algorithm as follows: $T = \exp(K\sqrt{n})$, $R = 2^{n/d}T$ and $D = 2^{\lceil \log_2 R \rceil}$. Note that

$$O(\log_2 R) = O(\log_2 T) = O(\log_2 D) = O(d + \frac{n}{d}) = O(\frac{n}{d}).$$

The computation of $|\psi_1\rangle$ can be performed with a $1/\operatorname{poly}(n)$ approximation in $d(\log D + \operatorname{poly}(d))$ quantum gates (see [Reg23, page 4]). The cost of computing $|\psi_2\rangle$

is that of a multi-scalar product (Algorithm 2) when the products with exponents 0 and 1 are evaluated on the fly. Note that the lines 3, 4 and 5 are repeated n/d times, and the dominant steps are 3 and 5 which have a cost of $O(M(m))$, because they use a constant number of operations in $\mathbb{G}$. By contrast, in line 4 one uses arithmetic in $\mathbb{Z}$ and uses

$$\tilde{O}(d) \leq \tilde{O}(\sqrt{n}) \leq \tilde{O}(\sqrt{m}) = o(M(m)),$$

so it is negligible. To summarize, the computation of $|\psi_2\rangle$ uses $O(\frac{n}{d}M(m))$ quantum gates. The measurement done to obtain $|\psi_3\rangle$ doesn't use quantum gates.

Finally, the quantum Fourier transform to obtain $|\psi_4\rangle$ is done with an approximation of $\epsilon := 1/\operatorname{poly}(d)$ using Coppersmith's method (see [Reg23, page 5]). It has a cost

$$\begin{aligned} O\left(d \log D \cdot (\log\log D + \log d)\right) &= O\left(d \cdot \tfrac{n}{d} \cdot (\log \tfrac{n}{d} + \log d)\right) \\ &= O(n \log n) \leq M(n) \leq M(m). \end{aligned}$$

Adding all costs, the complexity is dominated by the evaluation of $|\psi_2\rangle$, which uses $(\frac{n}{d}M(m))$ gates.

2. The definition of K-balanced lattices corresponds to the property required in Assumption 1 of [EG23a]. By [EG23a, Lemma 5] one obtains a basis of $\mathcal{L}$ (see also Lemma 2.4). Since $\operatorname{Vol}\mathcal{L} \neq \operatorname{Vol}\mathcal{L}_0$, we get that $\mathcal{L} \neq \mathcal{L}_0$. In particular, one of the elements of this basis is in $\mathcal{L} \setminus \mathcal{L}_0$, and, as discussed after Equation (6), such a vector enables the computation of the desired discrete logarithm.

3. From [Reg23, Appendix A] we know that each run of Regev's algorithm outputs a vector $w \in \mathbb{R}^d$ of the form $w = v + \epsilon$, where v is randomly frawn from $\mathcal{L}^*/\mathbb{Z}^d$ with uniform probability and ϵ is less than $\delta := \frac{\sqrt{d}}{\sqrt{2}R}$ with probability $1 - 1/\operatorname{poly}(d)$. Regev's algorithm is run $d+4$ times, and with probability $\geq 1/4$, a generating set $w_1, \ldots, w_{d+4}$ is obtained. This allows to write a matrix as in [Reg23, Lemma 4.4], whose columns are a basis of lattice $\mathcal{L}'$ in the same lemma. By [Reg23, Claim 5.1] there is an efficient classical algorithm which outputs a Regev-pseudobasis of $\mathcal{L}'$ with respect to $\sqrt{2}T$: a list of vectors $z_1, \ldots, z_k$ of norm $\sqrt{2k}2^{k/2}T$ which generate all vectors of norm less than $\sqrt{2}T$.

By the assumption on N, $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq T$, so there exists a vector $t \in \mathcal{L} \setminus \mathcal{L}_0$ such that $\| t \| \leq T$. By [Reg23, Lemma 4.4] there exists a vector $z \in \mathcal{L}'$ such that t are the first d coordinates of z and $\| z \| \leq (1 + (d+4)\delta^4)^{1/2})T < \sqrt{2}T$. Hence z is generated by $z_1, \ldots, z_k$ so, for one of the indices $i \in [1, k]$, the first d coordinates of z_i form a vectors which is in $\mathcal{L} \setminus \mathcal{L}_0$.
Finally, one obtains a vector in $\mathcal{L} \setminus \mathcal{L}_0$, and the discussion after Equation (4) shows that the algorithm succeeds in factoring N. □

Remark 3. *Note the importance of the parameter $T = \lambda_{\mathcal{L}_0}(\mathcal{L})$. Indeed, the cost of the algorithm depends on $\log_2 D$ whereas the correction of the algorithm requires $O(\log_2 D) = O(\log_2 T)$. Also note that in [Reg23] one takes $T = 2^d r^{1/d}$ which is of the form $\exp(K\sqrt{n})$ for a constant K.*

2.4.2. *Guaranties that Regev's algorithm succeeds on a specific instance of discrete logarithm problem.* In the context of Regev's algorithm with pre-computations, as proposed in [EG23a], we give a method to certify that Regev's algorithm succeeds in finding discrete logarithms.

Recall that $\mathbb{G}$ is a cyclic group with a given generator g. In the pre-computation stage, the following quantities $a_i = -\log_g g_i$ for $i = 2, \ldots, d-1$ are computed (for instance, using Shor's algorithm). The challenge in our notations will be $g_d = x$, while $g_1 = g$, and the problem is to compute $a_d = -\log_g(g_d)$ in short time. Let us assume that we are in the case $d = n^\alpha$, for some $\alpha < 1$.

It is clear that, the following two matrices:

$$\mathcal{L}: \begin{pmatrix} r & a_2 & \cdots & a_d \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{pmatrix} \qquad \mathcal{L}_0: \begin{pmatrix} r & a_2 & \cdots & 0 \\ & 1 & & \\ & & \ddots & \\ & & & r \end{pmatrix} \tag{9}$$

generate $\mathcal{L}$ and $\mathcal{L}_0$, respectively.

Note now that $\mathcal{L}_0$ doesn't depend on the challenge x, so one can compute a basis during the pre-computations. Since $d = n^\alpha$ and SVP can be solved in exponential time with respect to d (see e.g. [HS07]), one can find the shortest vector in $\mathcal{L}_0$ in sub-exponential time with respect to n. On the other hand, Regev's algorithm guaranties to find at least one vector, call it z, with norm at most T, for the chosen parameter T. Thus, if $\lambda_1(\mathcal{L}_0) > T$, then z is guaranteed to be outside $\mathcal{L}_0$.

It is important to note that, according to Gaussian heuristics, the expected first minimum of the lattice $\mathcal{L}_0$ is given by $\lambda_1(\mathcal{L}_0) \approx \sqrt{\frac{d-1}{2\pi e}}\, r^{1/(d-1)}$ (notice that the shortest vector in $\mathcal{L}_0$ has the same length as the shortest vector in the $d-1$-dimensional full-rank lattice generated by the first $d-1$-coordinates of the first $d-1$ vectors in the considered basis of $\mathcal{L}_0$), while for $\mathcal{L}$ it holds that $\lambda_1(\mathcal{L}) \approx \sqrt{\frac{d}{2\pi e}} r^{1/d}$. Hence, heuristically one can hope that the lattice $\mathcal{L}_0$, which is known in advance, allows to prove that the algorithm will succeed. This discussion leads to the following:

Definition 6. *If $\lambda_1(\mathcal{L}_0) \geq \exp(Kn/d)$ we say that $\mathcal{L}$ is K-Regev-certified.*

Given the above conventions, we have:

Lemma 2.8. *If $d < 0.41\sqrt{n}$ and $\mathcal{L}$ is 1-Regev-certified then Regev's algorithm with respect to the parameter $T = \exp(n/d)$ is correct.*

Proof. Regev's algorithm computes a Regev pseudo-basis with respect to T. By the LLL theorem $\lambda_1(\mathcal{L}) \leq 2^{\frac{d-1}{4}} r^{1/d}$. Then

$$2^{\frac{d-1}{4}} r^{1/d} \leq \exp((0.3 + \log_e 2)n/d) \leq \exp(n/d) = T.$$

Therefore Regev's algorithm with respect to T outputs at least a nonzero vector z with norm $|z| < T$. Since $\lambda_1(\mathcal{L}_0) > T$, we conclude that such z is not in $\mathcal{L}_0$. Finally, by Equation (7) we solve the discrete logarithm. □

Note that in practice it is fast to certify a lattice.

Experiment 1. *Let $n = 10000$, $d = 40 < 0.41\sqrt{n}$ and r the next prime after 2^n. Let $a_1 = 1415\cdots940$ be the first n decimal digits of π after the dot, a_2 the next n digits and so on until a_d. Using SageMath one finds in 6.56 seconds the shortest vector of $\mathcal{L}_0$:=knapsack($r, \{a_1, \ldots, a_{d-1}\}$). Hence, solving SVP in dimension d is fast for cryptographic sizes.*

We take $K = 1$ so that $T = \exp(n/d) < 1.8 \cdot 10^{75}$. One has $\lambda_1(\mathcal{L}_0) > 3.0 \cdot 10^{75}$ so $\lambda_1(\mathcal{L}_0) > T$. Hence $\mathcal{L}$ is K-Regev-certified. By Lemma 2.8, Regev's algorithm will succeed on this instance of DLP.

One can ask whether it is likely for a lattice to be 1−Regev-certified. We also conducted the following:

Experiment 2. *We fix $d = 40$ and take random integers r of n bits for $n = 50, 75, \ldots, 150$. We draw d random integers $a_1, \ldots, a_d$ and found with SageMath the percentage of knapsack lattices knapsack($r, \{a_1, \ldots, a_d\}$) which are 1-balanced respectively 1-Regev-certified.*

n	50	75	100	125	150	175
1−balanced	0	0	0	37	98	100
Regev−certified	100	100	100	100	100	100

One notes that Regev's algorithm succeeds and can be certified in advance that it succeeds in a much larger proportion of cases than when the lattice is 1-balanced.

2.5. Eliminating the heuristics in a multi-dimensional DLP variant of Regev.

We start by recalling the definition of the multi-dimensional DLP:

Definition 7 (multidimensional DLP -see e.g. [GR09])**.** *Let $\mathbb{G}$ be a commutative group and let $P_1, \ldots, P_d \in \mathbb{G}$ and $N_1, \ldots, N_d \in \mathbb{N}$ be given. The multidimensional discrete logarithm problem is to find (if they exist) the integers $n_i \in [-N_i, N_i]$, $1 \leq i \leq d$ such that*

$$[n_1]P_1 + \cdots [n_d]P_d = 0_{\mathbb{G}} \text{ and } n_d \neq 0.$$

The size of the problem is the integer $N = \prod_{i=1}^{d}(2N_i + 1)$.

The state of the art attacks on the multidimensional DLP express the security as a function of the size N. We can take N slightly smaller than $|G|$ so that:

the classical cost of the problem is divided by a factor between one and m (i.e. less than $\log_2 m$ bits of security) whereas it makes possible to prove that Regev's algorithm succeeds and has a smaller cost than Shor.

More precisely, we tackle the multidimensional DLP in the following setting:

- $\mathbb{G} = \mathbb{F}_{p^e}^*$ where p is a prime and $e = p^2$ such that $\mathbb{G}$ has a subgroup of prime order r such that $\log_2 r \sim \log_2 |\mathbb{G}|$; we write $\mathbb{F}_{p^e} = \mathbb{F}_p(y)$ for a $y \in \mathbb{F}_{p^e}$ and we call φ the minimal polynomial of y;
- $N = \lfloor 2^{m - \log_2 m} \rfloor$ where $m = \log_2 |\mathbb{G}|$;
- d is set to a value equivalent to $e/2 = p^2/2$, which will be made precise later.
- $g_1 = y$, g_d is a random element of $\mathbb{F}_p(y)$ and $g_2, \ldots, g_{d-1}$ are unitary quadratic irreducible polynomials over $\mathbb{F}_p$, evaluated in y;
- $N_1 = \ldots = N_d = (N^{1/d} - 1)/2 < 2^{m/d}/m$.

Note that

$$N_1 + \cdots + N_d \leq \frac{d}{m} 2^{m/d} \leq \frac{1}{\log p} 2^{m/d} < \frac{1}{2} 2^{m/d} < p^2/2. \tag{10}$$

On a classical computer the cost of an exponential, sub-exponential or quasi-polynomial algorithm for DLP in $\mathbb{G}$ has the same cost up to an exponent $1 + o(1)$

as a DLP in a group of size N. Hence we compare the cost of a multidimensional DLP with Regev for a parameter N to the DLP with Shor in the group $\mathbb{G}$.

We prove now an upper bound on $\lambda_{\mathcal{L}_0}(\mathcal{L})$ for the lattices $\mathcal{L}_0$ and $\mathcal{L}$ of Equations (6) and (5). In addition, the bound is small enough so that Regev is asymptotically faster than Shor. Then we recall complexity of the best classical algorithms on the same problem, including heuristic algorithms. We will conclude that Regev is both proven and the fastest.

Theorem 2.9. *The multidimensional DLP, as applied in the aforementioned setting, succeeds and has complexity $O(p^2 M(m))$.*

Proof. We set $d = (p^2 - p + 4)/2$, which is equivalent to $e/2$. We note that $d-2$ is the number of irreducible unitary quadratic polynomials in $\mathbb{F}_p[X]$, evaluated in y:

$$i = 2, \ldots, d-1, \quad g_i = y^2 + \alpha_i y + \beta_i.$$

We claim that $\lambda_{\mathcal{L}_0}(\mathcal{L}) < e/2$. For this we prove that $\lambda_1(\mathcal{L}) < e/2$ and $\lambda_1(\mathcal{L}_0) \geq e/2$. Here, we make the clarification that we shall work with the L^1 norm instead of the L^2 norm.

Assume by contradiction that $\lambda_1(\mathcal{L}_0) < e/2$. Let $z \in \mathbb{Z}^d \backslash \{0\}$ be such that $z_d = 0$ and $\| z \|_1 < e/2$. This is equivalent to

$$y^{z_1} \prod_{i=2}^{d-1} (y^2 + \alpha_i y + \beta_i)^{z_i} = 1.$$

We set $\nu(X) := \text{numerator}\left(X^{z_1} \prod_{i=2}^{d-1} (X^2 + \alpha_i X + \beta_i)^{z_i} - 1\right)$ in the polynomial ring $\mathbb{F}_p[X]$. We assumed that $\nu(y) = 0$ or equivalently that $\varphi(X)$ is a multiple of $\varphi(X)$, the modulus of $\mathbb{F}_{p^e}$. If $\nu(X) = 0$ then

$$X^{\max(z_1,0)} \prod_{\substack{2 \leq i \leq d-1 \\ z_i > 0}} (X^2 + \alpha_i X + \beta_i)^{z_i} = X^{\max(-z_1,0)} \prod_{\substack{2 \leq i \leq d-1 \\ z_i < 0}} (X^2 + \alpha_i X + \beta_i)^{|z_i|},$$

which is impossible because the polynomials $g_i(X)$ are irreducible and $z \neq 0$. Also $\deg \nu(X) \leq 2\| z \|_1 < e = \deg \varphi(X)$ so $\nu(X)$ is not a multiple of $\varphi(X)$, which is a contradiction.

Now, by the definition of the multidimensional DLP there exists $z = (z_1, \ldots, z_d)$ such that $\| z \|_1 \leq N_1 + \cdots + N_d$. By Equation (10) we obtain $\| z \|_1 < p^2/2$. We combine with the result above that $\lambda_1(\mathcal{L}_0) \geq e/2$ and obtain

$$\lambda_1(\mathcal{L}) < p^2/2 \leq \lambda_1(\mathcal{L}_0),$$

so $\lambda_{\mathcal{L}_0}(\mathcal{L}) \leq p^2/2$. In other words, there exist a vector $z \in \mathcal{L} \setminus \mathcal{L}_0$ such that $\|z\|_1 < p^2/2$. Since $\|z\|_2 \leq \|z\|_1$, we also get that $\|z\|_2 < p^2/2$. Consequently, we can run Regev's algorithm with $T = p^2/2 = e/2$ and it will succeeds because $T \geq \lambda_{\mathcal{L}_0}(\mathcal{L})$. By Proposition 2.7 we obtain that its complexity is

$$O((d + \log_2 T) M(m)) = O((p^2 + 2\log_2 p) M(m)) = O(p^2 M(m)).$$

□

Let us now compare Regev to the other algorithms. The state-of-the-art algorithms for the multidimensional DLP are adaptations of the DLP algorithms. In a generic group an on elliptic curves the best algorithm is Pollard's rho, whose complexity is $O(\sqrt{N})$, where N is the cardinality of the group. It was extended to

algorithm	reference	classical complexity	quantum complexity
Shor	[Sho94]	none	$O(p^2 \log pM(m))$
Pollard's rho/Galbraith & Ruprai	[GR09]	$O(p^{p^2/2})$	Grover speedup
quasi-polynomial	[BGJT14]	$p^{O(\log p)}$	no speedup
Regev	Theorem 2.9	none	$O(p^2 M(m))$

TABLE 2. Comparison of Regev for multidimensional DLP to other algorithms. When multidimensional variants exist we idicate the name of the algorithms, otherwise we write the DLP algorithm and indicate its complexity.

arbitrary dimensions $d \geq 2$ by Galbraith and Ruprai [GR09], and it has complexity $O(\sqrt{N})$, where N is the parameter in the definition of the multidimensional DLP. To our knowledge, the other DLP algorithms don't have multidimensional versions, so we use the DLP complexities for a comparison. Note that the complexity of Shor's algorithm is $O(\log_2 rM(m)) = O(mM(m)) = O(p^2 \log pM(m))$, which is asymptotically slower than Regev. Any classical algorithm can be applied on a quantum one and Grover obtains a speedup by a constant in the exponent, so Pollard's rho remains exponential in n. Finally, the quasi-polynomial algorithm [BGJT14] has complexity $p^{O(\log e)} = p^{O(\log_2 p)}$. This complexity corresponds to classical computers. It can be emulated on a quantum computer but it is not known to have a speed-up. We summarize the various complexities in Table 2.

3. Extending Regev to elliptic curves

The aim of this section is to investigate the discrete logarithm problem on elliptic curves. A key aspect of this investigation is the number of bits needed to represent a point of an elliptic curve over a prime field and what it means to be "small". For this we recall the naive height of a point of an elliptic curve over the rationals and also over a prime field. We start with the height of a nonzero rational u/v which is defined by $h(u/v) = \max(\log|u|, \log|v|)$, where log denotes the logarithm in base e. The naive (or Weil) height of a rational point (x, y) of an elliptic curve is

$$h(x, y) = \max(h(x), h(y)). \tag{11}$$

Clearly, the naive height of a point is between one fourth and one half of the bit size representation of the point.

Given a prime p and an integer $a \in [1, p-1]$, the rational reconstruction of a modulo p is the rational number u/v, where $u, v \in \mathbb{Z}$, with the smallest height such that $u \equiv va \pmod{p}$. This can be computed in polynomial time in the bit size of p by reducing a two-dimensional lattice (see [Wan81] or more recent books of computer arithmetics). The height of an element of $\mathbb{F}_p$ is the height of its corresponding rational reconstruction.

Recall that the discrete logarithm problem on elliptic curves over finite fields involves two $\mathbb{F}_p$ points P and Q on an elliptic curve E defined over $\mathbb{F}_p$. The goal is to find the positive integer a such that $Q = [a]P$, which we denote as $\log_P Q = a$. In Regev's algorithm one uses auxiliary points $P_2, \ldots, P_{d-1}$ for a parameter d and

sets $P_1 = P$ and $P_d = Q$. The algorithm is efficient if the points can be represented using a small number of bits, and if the cost to evaluate all linear combinations $\sum_{i=1}^{d} [\varepsilon_i] P_i$ with $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_d) \in \{0,1\}^d$ remains computationally feasible (see Section 3.4). By analogy with the multiplicative case, we seek $d-2$ points $P_i = (x_i, y_i)$ for $i = 2, \ldots, d-1$, whose coordinates can be expressed using $O(\sqrt{n})$ bits, where $n = \lfloor \log_2 p \rfloor$ (here n is approximately the same size as $\log_2 |E(\mathbb{F}_p)|$). This is significantly fewer than $m = 2n$ bits required to represent arbitrary points on the curve. Since Regev's algorithm offers a speed-up by a factor of d compared to Shor's algorithm (Prop. 2.7), we aim to make d as large as possible.

Since we are interested in elliptic curves that possess at least two small points, we shall restrict to the case where the coefficients of the elliptic curve are bounded by a polynomial in $\log_2 n$ (see Proposition 3.2 bellow). From a cryptographic perspective, this restriction is not a hindrance, as most elliptic curves chosen for cryptographic implementation already meet this condition.

Notice that, in this case, the $\tilde{O}$ complexity of the algorithm is the same as in the multiplicative case. Indeed, the addition law of the elliptic curve is defined by rational functions and requires one inversion along with a constant number of multiplications (less than 10) in the coefficient field. The complexity of adding two points $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ is that of multiplying two of the coordinates, e.g. x_P and y_Q, or a product of coordinates by a coefficient of E. Since the cost of the multiplication is quasi-linear, the complexity of adding two points of height $O(\sqrt{n})$ is $\tilde{O}(\sqrt{n})$, which is negligible compared to n. The rest of the algorithm and analysis is identical to the multiplicative case (Proposition 2.7).

The remaining obstacle is to find points with coefficients that can be written on $O(\sqrt{n})$ bits. Bellow, we shall prove two negative results (Proposition 3.1 and Proposition 3.2 stating that such points are very few on random elliptic curves.

In one direction we show that some classical conjectures in number theory imply that there exist elliptic curves with $d = \log_2 n$. In practice we give a list of elliptic curves of high rank which are good examples to be tackled in early simulations or implementations of Regev's algorithm. In Theorem 3.6 we prove that the speed-up of Regev with respect to Shor is $\log_2 n$. This is less than the speed-up in the multiplicative case, but it also tends to infinity with n.

In the second direction we use Goldfeld's conjecture to prove that one can apply Regev's algorithm to all elliptic curves with coefficients of size polynomial in $\log_2 n$. We don't make a complexity analysis in this case but we apply it to some of the curves in the NIST curves.

3.1. **Obstacle to a direct extension of Regev to elliptic curves.** The main result of this section asserts that only an exponentially small fraction of elliptic curves defined over a prime field possess points that can be represented using $O(\sqrt{n})$ bits. Such points will henceforth be referred to as *small points*. More precisely, we provide an upper bound on the probability that an elliptic curve

$$E_{a,b} : y^2 z = x^3 + axz^2 + bz^3$$

has small points. In the next proposition we sample random elliptic curves by randomly selecting the pair (a, b), rather than considering isomorphism classes of elliptic curves, as this would introduce unnecessary technicalities without impacting the conclusion of our negative result.

Proposition 3.1. *Let p be an odd prime, and let α be such that $0 < \alpha < 1/3$. If integers a and b are drawn randomly with uniform probability from $((-p/2, p/2) \bigcap \mathbb{Z})^2$, then the expected value of the number of points with height less than $\alpha\, n$ is bounded as follows:*

$$\mathbb{E}\left(\#\left\{ (x,y,z) \in \left((-p^\alpha, p^\alpha) \bigcap \mathbb{Z}\right)^3 \right\} \;\middle|\; \begin{array}{l} a, b \text{ random in } \mathbb{F}_p^2 \\ \text{uniform} \\ 4a^3 + 27b^2 \neq 0 \end{array} \right) \leq \frac{8p^{3\alpha}}{p-2}.$$

In particular, the probability that a random elliptic curve has points of height less than $\alpha\, n$ is less than $\frac{8p^{3\alpha}}{p-2}$.

Proof. Let $X_0 := \{(a,b) \in ((-p/2, p/2) \bigcap \mathbb{Z})^2 \mid 4a^3 + 27b^2 \neq 0\}$ and $X_1 := \{(a,b) \in ((-p/2, p/2) \bigcap \mathbb{Z})^2 \mid 4a^3 + 27b^2 = 0\}$. For each $(a,b) \in ((-p/2, p/2) \bigcap \mathbb{Z})^2$, let

$$R_{a,b} := \left\{ (x,y,z) \in \left((-p^\alpha, p^\alpha) \bigcap \mathbb{Z}\right)^3 \;\middle|\; (x:y:z) \in E_{a,b}(\mathbb{F}_p) \setminus \{\infty\} \right\}$$

be the set of points of height less than $\alpha\, n$ on the curve $y^2 z = x^3 + axz^2 + bz^3$, and set $r_{a,b} := |R_{a,b}|$. Then, the expected value will be $(\sum_{(a,b)\in X_0} r_{a,b})/|X_0|$.

Since for each value of a there exist at most 2 values of b such that $(a,b) \in X_1$, we have $|X_1| \leq 2p$. Therefore, $|X_0| \geq p^2 - 2p$. On the other hand, we can upper-bound the numerator $\sum_{(a,b)\in X_0} r_{a,b}$ by summing over all pairs $(a,b) \in X_0 \cup X_1$, as follows: $\sum_{(a,b)\in X_0 \cup X_1} r_{a,b} \leq 8p^{1+3\alpha}$.
Indeed, since there is exactly one value of b for each quadruple (x,y,z,a), the total number of quintuples is at most $8p^{3\alpha} \times p$. Hence, we obtain: $\frac{\sum_{(a,b)\in X_0} r_{a,b}}{|X_0|} \leq \frac{8p^{1+3\alpha}}{p^2-2p} = \frac{8p^{3\alpha}}{p-2}$. Finally, since $r_{a,b}$ is a natural number, we get $\frac{\#\{(a,b)\in X_0 | r_{a,b} \neq 0\}}{|X_0|} \leq \frac{\sum_{(a,b)\in X_0} r_{a,b}}{|X_0|} \leq \frac{8p^{3\alpha}}{p-2}$. □

Remark 4. *Even though a triple (x,y,z) may, in fact, correspond to the same point $(x:y:z)$ on the elliptic curve, the conclusion remains correct.*

Observe that by choosing $\alpha = O(\frac{1}{\sqrt{n}})$ in Proposition 3.1, the points of height less than $\alpha\, n$ are those that can be represented using $O(\sqrt{n})$ bits. Therefore, the probability that a random elliptic curve has small points is less than $\frac{1}{2^{n-O(\sqrt{n})}}$.

Proposition 3.2. *Let $c > 0$ be a constant, p a prime, and n its bit size. Let $(a,b) \in \mathbb{F}_p$ such that $4a^3 + 27b^2 \neq 0$. If $E_{a,b}$ has two affine points P_1, P_2 with $P_1 \neq \pm P_2$ for which $\max(h(P_1), h(P_2)) \leq c\sqrt{n}$, then a and b have heights less than $3c\sqrt{n} + 2$ and $4c\sqrt{n} + 3$, respectively.*

Proof. Let $P_1(x_1 : y_1 : 1)$ and $P_2(x_2 : y_2 : 1)$ be two affine points as described in the statement. Since $P_1 \neq \pm P_2$, we get that $x_1 \neq x_2$. Then we have

$$\begin{aligned} a &= \tfrac{(y_1^2 - x_1^3) - (y_2^2 - x_2^3)}{x_1 - x_2} \\ b &= y_1^2 - x_1^3 - a x_1. \end{aligned}$$

The first equation bounds the height of a and then the second bounds the height of b. □

We conclude that only curves with small coefficients can have two or more small points. Such curves are even less frequent than the probability given in Proposition

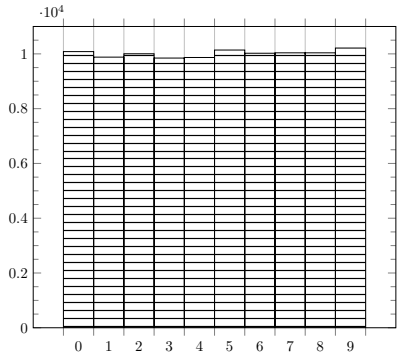

FIGURE 1. Distribution of $\sqrt{x^3+ax+b}$ when $x = 1, 2, \ldots, X$ for a bound X (see Experiment 3).

5, which shows that most curves possessing a small point cannot have two such points. In practice, for Regev's algorithm, more than one point is required to achieve observable efficiency, so in the search for Regev-friendly curves, we are compelled to focus on curves with small coefficients. Fortunately, this is the case for all the curves proposed on the NIST list.

To reinforce the conclusion of the above results, we considered an experiment in which we search for affine points $(x, y, 1)$ that can be represented using $O(\sqrt{n})$ bits. To do this, one enumerates all possible values of the x-coordinate and retains randomly one of the two possible values of $y := \sqrt{x^3+ax+b}$. The results of this experiment show that the values of y are uniformly distributed.

Experiment 3. *We fix a random elliptic curve:* $y^2 = x^3 + ax + b$ *with* $a = 3$ *and* $b = 11$. *We also fix a randomly drawn* 50*-bit prime:* $p = 234094748715283$. *We enumerate* $x = 1, 2, \ldots, X = 10^5$*, take a random value among the two square roots of* $x^3 + ax + b$ *and place them in one of the intervals* $[1, p/10]$, $[p/10, 2p/10]$, $\ldots$, $[9p/10, p]$. *We summarize the distribution in Figure 1. Repeating the process on a sample of 100 primes of* 50 *bits, the maximum value of the statistical distance to uniform distribution was* 0.00648.

3.2. **Some examples of Regev-friendly curves.** In this section we search for instances of the discrete logarithm for elliptic curves where Regev's algorithm requires few quantum resources. Due to the lack of large quantum computers to run the algorithm, we apply the definition of Regev-friendliness to choose the elliptic curves on which the algorithm performs most efficiently. More specifically, we seek elliptic curves along with d points on them, aiming for d to be as large as possible, such that the points of the form $\sum_{i=1}^{d} \varepsilon_i P_i$ can be represented using a small number of bits for all tuples $(\varepsilon_1, \ldots, \varepsilon_d) \in \{0, 1\}^d$.

In what follows, we will examine two scenarios: elliptic curves over large characteristic prime fields and those over small characteristic finite fields.

3.2.1. *Elliptic curves over prime fields.* Our strategy is to use elliptic curves $E_{[a_1,a_2,a_3,a_4,a_6]} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ defined over $\mathbb{Q}$ with small coefficients which have many points with small coordinates. Then, we find a prime p large enough such that $E(\mathbb{F}_p)$ provides strong security against classical computer

rank	curve	$(\varepsilon_1, \ldots, \varepsilon_r)$	$h(\sum_{i=1}^r \varepsilon_i P_i)$
1	$[0, 0, 0, 1, 1]$	(1)	1
2	$[0, 1, 1, -2, 0]$	$(1, 1)$	3
3	$[0, 6, 5, 5, -6]$	$(0, 1, 0)$	5
4	$[1, -1, 6, -82, 280]$	$(0, 1, 1, 1)$	14
5	$[0, 0, 1, -79, 342]$	$(0, 1, 0, 1, 0)$	18
5	$[1, 0, 0, -22, 219]$	$(1, 1, 0, 0, 1)$	25
6	$[1, 1, 0, -2582, 48720]$	$(0, 1, 1, 1, 0, 1)$	42
6	$[0, 0, 1, -7077, 235516]$	$(1, 1, 0, 1, 1, 1)$	45
6	$[1, -1, 0, -16249, 799549]$	$(1, 1, 1, 0, 0, 1)$	45
7	$[1, 0, 1, -14733, 694232]$	$(1, 1, 1, 1, 0, 1, 1)$	75
8	$[1, -1, 0, -71899, 5522449]$	$(1, 1, 1, 1, 1, 0, 1, 0)$	76
9	$[0, 0, 1, -3835819, 2889890730]$	$(1, 1, 1, 1, 0, 1, 1, 0, 0)$	95
10	$[1, -1, 0, -1536664, 648294124]$	$(0, 0, 0, 1, 1, 0, 0, 1, 1, 1)$	110
11	$[0, 0, 1, -16359067, 26274178986]$	$(0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1)$	147
12	$[0, 0, 1, -6349808647, 193146346911036]$	$(1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1)$	202

TABLE 3. A short list of curves of small discriminant and high rank. We list the bit size $h(P)$ where P is the point of the form $\sum_{i=1}^r [\varepsilon_i] P_i$ of the highest bit size in the set $(\varepsilon_1, \ldots, \varepsilon_r) \in \{0, 1\}^r$. Here h is as defined in Equation (11).

attacks on the discrete logarithm problem, while still allowing Regev's algorithm to be implemented with a relatively small number of quantum gates. Therefore, these curves can be classified as insecure for the discrete logarithm problem.

In Table 3 we list some elliptic curves of small height and large rank. We end the list at rank 12 because the height of the points is over 200 bits but we include in the additional files curves of rank up to 29. All the curves are taken from the literature, the ones up to rank 11 are due to Elkies, the one of rank 12 to Mestre and the others to a list of authors listed in [Duj24].

3.2.2. *Small characteristic curves.* The idea of using elliptic curves defined over $\mathbb{Q}$ is well-suited for generating weak elliptic curves defined over $\mathbb{F}_p$. By analogy, to produce elliptic curves over $\mathbb{F}_{p^n}$ for a small prime p, we use elliptic curves defined over $\mathbb{F}_p(t)$. We summarize in Appendix B examples of elliptic curves suited for Regev in small characteristic.

3.3. **A list of NIST curves that are Regev-friendly.** None of the examples of Regev-friendly curves in Section 3.2 are used in cryptography, in particular they are not part of the curves recommended by the NIST [Nat23], referred below as the NIST list. Recall that being Regev-friendly essentially means having small coefficients and many points with small coordinates. Many of the curves in the NIST curve have very small coefficients. This is not a coincidence because this allows to speed up the crypto-systems without introducing any known attacks on a classical computer. The main issue will be the second required property: the presence of many small points.

Recall also that researchers have previously discussed the scenario where the standards can be prone to attacks that are only known to the emitting authority, e.g. [BCC+15],[FPRE15]. We investigate which curves in the NIST list use fewer quantum resources than other curves of the same key size.
Main strategy: We lift a curve E defined over a finite field $\mathbb{F}_p$ for a prime p (resp. $\mathbb{F}_{2^n}$ for an integer n) to the curve $\overline{E}$ defined over $\mathbb{Q}$ with coefficients in $(-p/2, p/2]$

equation	name	standard	applications
$y^2 + x^3 + 7$	secp256k1	[Nat23]	Bitcoin[3], Etherium[4]
$y^2 = x^3 + 486662x^2 + x$	Curve25519	[Nat23]	OpenSSH[5]
$x^2 + y^2 = 1 + (21665/121666)x^2y^2$	Ed25519	identical to Curve25519	
$y^2 = x^3 + 4$	BLS12-381	[Cer10]	BLS signatures[6]
$y^3 = x^3 + 3$	BN-254	no longer recommended	
$y^2 + xy = x^3 + 1$	Koblitz curves	[Nat23]	hardware implementation[7]

TABLE 4. Examples of some of the most used elliptic curves in cryptography.

(resp. over $\mathbb{F}_2(t)$ with coefficients polynomials of degree at most $n/2$). Then we search for a twist $\overline{E}_\delta$ of $\overline{E}$ by a $\delta \in \mathbb{Q}$ (resp. $\delta \in \mathbb{F}_2(t)$) so that:

- δ is a square in $\mathbb{F}_p$ (resp. $\mathbb{F}_{2^n}$)
- $\overline{E}_\delta$ has large rank.

The rationale for adopting this strategy is based on the following straightforward yet important lemma.

Lemma 3.3. *A curve E is Regev-friendly if there exists a square δ in its base field so that the twist of E by δ is Regev-friendly.*

Proof. A twist by a square is an isomorphism $\phi : E \to E_\delta$, so that if P and Q are two points of E then $\log_P Q = \log_{\phi(P)} \phi(Q)$. □

We tried our strategy on all the curves of the NIST list and we summarize below the short list where we succeeded.

Fact 3.4. *The following curves in the NIST list are Regev-friendly:*

(1) The Montgomery curves, $By^2 = x^3+Ax^2+x$, the recommended curves have $B = 1$ and small A. Some examples are: Curve25519, Curve383187, M-221, M-383, M-511. The twisted Edwards curves, $ax^2+ay^2 = 1+dx^2y^2$, the recommended curves have $a = 1$ and small d. They are different equations for the Montgomery curves and can be treated together when evaluating their security. Some examples are: Curve1174 (bi-rationally equivalent to Curve25519), Curve41417, E-222, E-382, Ed448-Goldilocks and E-521.

(2) The elliptic curves of the form $y^2 = x^3 + ax + b$ with a and b small. They include the curve sec256k1.

(3) The elliptic curves in the pairing-based cryptography which have small equations, in particular the BN and the BLS12-381 curves. Note that for the BLS12 curve, the equation is independent on the bit size of p. See paragraphs secp256k1 and BLS12-381 at the end of this list.

(4) The Koblitz curves have equation $y^2 + xy = x^3 + 1$ and are seen as defined over $\mathbb{F}_{2^n}$ where n is the bit size, e.g. the Koblitz curve K-233 has $n = 233$.

Before giving the justification, note that the Regev-friendly curves are among the most used ones in cryptography (Table 4).

justification: **1.** *1. The Montgomery and twisted Edwards curves. Bernstein's curve*

$$E = Curve25519: \quad y^2 = x^3 + 486662x^2 + x.$$

has the B-twist with $B = 9282$ of rank 4:

$E_B : By^2 = x^3 + 486662x^2 + x$
$P_1 = (17793594 : 17793594 : 2650183726878506)$
$P_2 = (434841177498 : -3148775574127 : 99509057901162)$
$P_3 = (245681894294428907406 : -1779134065187933378821 : 23892963464408987193336)$
$P_4 = (2634676586421829950 : -19151161818923118455 : 6489881247255884478)$
$\max\{h(\sum_{i=1}^{3} \varepsilon_i P_i)) : \varepsilon \in \{0,1\}^3\} = 162.9.$

2. The curves with small Weierstrass equations include secp256k1 whose equation is $E : y^2 = x^3 + 4$. Its twist $E_\delta : y^2 = x^3 + 7\delta^3$ for $\delta = 942$ has rank 3:

$E_\delta/\mathbb{Q} : y^2 = x^3 + 7 \cdot 942^3$
$P_1 = (-41919/25 : 4214979/125 : 1)$
$P_2 = (1570 : 98596 : 1)$
$P_3 = (1794 : 107820 : 1)$
$\max\{h(\sum_{i=1}^{3} \varepsilon_i P_i)) : \varepsilon \in \{0,1\}^3\} = 41.8$

3. Many of the pairing-friendly curves have very small coefficients. For the curves BN and BLS 12 we have twists of rank 3:

- *E=BLS12-38 with $\delta = 2453$1*
  $E_\delta/\mathbb{Q} : y^2 = x^3 + 4 \cdot 2453^3$
  $P_1 = (-375453727/131044 : 8940748216367/47437928 : 1)$
  $P_2 = (31889/4 : 6017209/8 : 1)$
  $P_3 = (9225733/9 : 28022142313/27 : 1)$
  $\max\{h(\sum_{i=1}^{3} \varepsilon_i P_i)) : \varepsilon \in \{0,1\}^3\} = 88.7$
- *E=BN with $\delta = 298$*
  $E_\delta/\mathbb{Q} : y^2 = x^3 + 3 \cdot 298^3$
  $P_1 = (-2343770/6241 : -2534554964/493039 : 1)$
  $P_2 = (-263 : 7823 : 1)$
  $P_3 = (745 : 22201 : 1)$
  $\max\{h(\sum_{i=1}^{3} \varepsilon_i P_i)) : \varepsilon \in \{0,1\}^3\} = 38.6$

4. The Koblitz curves, in particular K-233, have the equation $\overline{E}/\mathbb{F}_2(t) : y^2 + xy = x^3 + 1$. We set $x, y \in \mathbb{F}_2[t]$ (e.g. $y = t$ and $x = 1$) and use the twist $\overline{E}_\delta : y^2 + xy = x^3 + \delta x^2 + 1$ with $\delta = (y^2 + xy + x^3 + 1)/x^2$.

Using Magma we obtained that the analytic rank of all the twists associated to polynomials x and y of degree less than 12 is always 1. In conclusion we use $E_\delta/\mathbb{F}_2(t) : y^2 + xy = (t^2 + t)x^2 + 1$ and $P_1 = (1 : t : 1)$.

Note however that our strategy has its limits. We couldn't find any argument that the following curves are Regev-friendly:

a) the curves in short Weierstrass form $y^2 = x^3 + ax + b$ with $a = -3$ and b an arbitrary element of $\mathbb{F}_p$. Examples include: P-224, P-256, P-384, brainpoolP256t1, brainpoolP384t1 and FRP256v1. Note that P-n are denoted W-n by some authors.

b) binary curves other than Koblitz's curves; they have equation $y^2 + xy = x^3 + x^2 + b$ with large degree $b \in \mathbb{F}_2[t]$ representing an element of $\mathbb{F}_{2^n}$. Examples include: B-233, B-283, B-409, B-571.

3.3.1. *More on quadratic twists.* The strategy used in the beginning of this section can be applied to any elliptic curve $E/\mathbb{F}_p$. One searches for squares $\delta \in \mathbb{F}_p$ such that the δ-twist of E has a lift of high rank. We refer to it as a strategy rather than an algorithm because we do not know whether such twists exist or how to find them systematically if they do. Hence, we pose the following problem.

Problem 1. *Given an elliptic curve E with rational coefficients, find a twist of high rank.*

A classical conjecture (Conjecture C.3) states than there exist elliptic curves with arbitrarily large rank. One can ask whether the same is true when restricted to the twists of a given elliptic curve. The following example of an elliptic curve which has been studied in the literature supports the heuristics that twists of large rank exist and are numerous enough to be found by enumerating possible values of δ.

Example 1. *Watkins et al.* [WDE+14] *tackle the problem for $E : y^2 = x^3 - x$, which is not one of the NIST curves. For this curve Goldfeld's conjecture is proven (see* [Kri20]*), and they obtained experimentally that there are many twists of large rank. More experiments about the densities of high rank twists are in* [RS01].

A second argument that twists of high rank exist is that the average value of the rank is the same among the twists of a curve as in the set of all elliptic curves (see the following conjecture and [KS98]).

Conjecture 3.5 (Goldfeld's Conjecture [BH12])**.** *Let $E/\mathbb{Q}$ (resp. $E/\mathbb{F}_q(t)$ for some prime power q with $(q, 6) = 1$ be a fixed elliptic curve. For an elliptic curve $E : y^2 = f(x)$ we denote by E_δ the twist $\delta y^2 = f(x)$. We have*

$$\lim_{D\to\infty} \frac{\sum_{\delta<D} r(E_\delta)}{\#\{\delta \mid |\delta| < D\}} = \frac{1}{2},$$

where $r(E_\delta)$ is the analytic rank, i.e. the order of vanishing at $s = 1$ of the L-function of the quadratic twist $E_\delta/\mathbb{Q}$ (resp. $E_\delta/\mathbb{F}_q(t)$).

We remark that Goldfeld's Conjecture is implied by the moment conjectures as suggested by Katz-Sarnak philosophy [KS98]. Those conjectures have been recently proved over the function field $\mathbb{F}_q(t)$ in the easier case of Dirichlet characters in [BDPW24], and their method is believed to adapt also to twists of elliptic curves.

3.4. **Asymptotic complexity.** In order to conclude that Regev's algorithm has an elliptic curve version, we prove, under the conjectures of Appendix C, that Regev's algorithm is faster than Shor's algorithm by a factor which tends to infinity.

Theorem 3.6. *Let $E/\mathbb{F}_q$ be an elliptic curve. Assume Lang's conjecture, Hall's conjecture, and the rank conjecture (Conjectures C.1, C.2 and C.3). Regev's algorithm is implemented on E with $d - 2 = \tau$ points $P_2, \ldots, P_{d-1}$ of canonical height as in Conjecture C.1. The computation of the linear combinations $\sum \epsilon_i P_i$ with $\epsilon_i \in \{0, 1\}$ is treated as part of pre-computation. Assume also that the lattice $\mathcal{L}$ defined as in Equation (5) is K-balanced for a parameter $K > 0$. Let α and c be as in Conjecture C.3 and let $\varepsilon > 0$ be a constant which can be taken arbitrarily*

small. Then Regev's algorithm with parameter K uses $(\log n)^{2/(4\alpha+1)-\epsilon}$ times less quantum gates than Shor.

Proof. As before, r is the cardinality of the group of discrete logarithm and n is its binary size. In the following $\epsilon > 0$ is an arbitrary constant. We set

$$N := \exp\left(\frac{2}{4\alpha+1}(\log n \log\log n)^{2/(4\alpha+1)-\epsilon}\right) \text{ and } d-2 = \tau := \min(1,c)\left(\frac{\log N}{\log\log N}\right)^{\alpha},$$

which implies $\frac{\log N}{\log\log N} \leq (\log n)^{2/(4\alpha+1)-\epsilon}$.

Since $\hat{h}$ is a positive quadratic form (see e.g. [Zim76]), we can view the Neron-Tate pairing $\langle , \rangle$ as an inner product on the curve modulo the torsion subgroup. Hence, the triangle inequality holds:

$$\begin{aligned}
\sqrt{\hat{h}(\textstyle\sum_{i=2}^{1+\tau}[\varepsilon_i]P_i)} &\leq \textstyle\sum_{i=2}^{1+\tau}\sqrt{\hat{h}(P_i)} \leq \tau\sqrt{\hat{h}(P_\tau)} \\
&\leq \exp\tfrac{1}{2}\left(2\log\tau + \tfrac{1}{12}\log H(E) + \epsilon(N)\log\log N + \sqrt{\tfrac{\log N}{\log\log N}}\tfrac{\tau^2-\tau}{2}\right) \\
&\ll \exp(\tfrac{1}{2}(\tfrac{\log N}{\log\log N})^{1/2}\tau^2) \\
&\ll \exp(\tfrac{c^2}{4}(\log n)^{1-\varepsilon}(\log\log n)^{O(1)}) \ll n^{1/4+\epsilon'},
\end{aligned}$$

for a constant $\varepsilon' < 1/4$. In the last three lines we used Conjectures C.1, C.2 and C.3. By [Zim76, Section 2] we have that for all P, $|h(P)-\hat{h}(P)| = O(\max(\log|a|, \log|b|))$. Observe that the right hand side is $O(\log H(E))$ which is bounded from above by Hall's conjecture by $O((\log N)(\log\log N)^2) = o(n)$. Hence, we obtained for all $P = \sum_{i=1}^{\tau+1}[\varepsilon_i]P_i$,

$$h(P) \leq \hat{h}(P) + o(n) = o(n).$$

We have shown that $\mathbb{G}$ is K-Regev-friendly. By Proposition 2.7, Regev's algorithm with parameter K is correct for $E(\mathbb{F}_p)$ and uses $O((d+\frac{n}{d})M(n))$ gates. This represents a reduction by a factor d with respect to Shor. Finally,

$$d = 2+\min(1,c)\left(\frac{\log N}{\log\log N}\right)^{\alpha} = (\log n)^{2\alpha/(4\alpha+1)+o(1)} = \begin{cases} (\log n)^{1/3+o(1)}, & \text{if } \alpha = 1/2 \\ (\log n)^{2/5+o(1)}, & \text{if } \alpha = 1. \end{cases}$$

□

References

[BCC+15] Daniel J Bernstein, Tung Chou, Chitchanok Chuengsatiansup, Andreas Hülsing, Eran Lambooij, Tanja Lange, Ruben Niederhagen, and Christine Van Vredendaal. How to manipulate curve standards: A white paper for the black hat http://bada55. cr. yp. to. In *International Conference on Research in Security Standardisation*, pages 109–139. Springer, 2015.

[BDPW24] Jonas Bergström, Adrian Diaconu, Dan Petersen, and Craig Westerland. Hyperelliptic curves, the scanning map, and moments of families of quadratic l-functions, 2024.

[BGJT14] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–16. Springer, 2014.

[BH12] Salman Baig and Chris Hall. Experimental data for goldfeld's conjecture over function fields. *Experimental Mathematics*, 21(4):362–374, 2012.

[Bra93] Stefan A Brands. An efficient off-line electronic cash system based on the representation problem. 1993.

[BS16] Jean-François Biasse and Fang Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 893–902. SIAM, 2016.

[Cer10] Certicom research. Standards for efficient cryptography sec 2: Recommended elliptic curve domain parameters, 2010. Available online at http://www.secg.org/sec2-v2.pdf.

[Dav97] Sinnou David. Points de petite hauteur sur les courbes elliptiques. *journal of number theory*, 64(1):104–129, 1997.

[dBDF20] Koen de Boer, Léo Ducas, and Serge Fehr. On the quantum complexity of the continuous hidden subgroup problem. In *Advances in Cryptology – EUROCRYPT 2020*, volume 12106 of *Lecture notes in computer science*, pages 341–370. Springer, 2020.

[Duj24] Andrej Dujella. History of elliptic curves rank records, 2024. Available online at https://web.math.pmf.unizg.hr/~duje/tors/rankhist.html.

[EG23a] Martin Ekerå and Joel Gärtner. Extending Regev's factoring algorithm to compute discrete logarithms, 2023. arXiv preprint arXiv:2311.05545.

[EG23b] Martin Ekerå and Joel Gärtner. Simulating regev's quantum factoring algorithm, 2023. https://github.com/ekera/regevnum.

[EHKS14] Kirsten Eisenträger, Sean Hallgren, Alexei Kitaev, and Fang Song. A quantum algorithm for computing the unit group of an arbitrary degree number field. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 293–302, 2014.

[Elk94] Noam D Elkies. Mordell-Weil lattices in characteristic 2: I. construction and first properties. *International Mathematics Research Notices*, 1994(8):343–361, 1994.

[EW04] Noam D Elkies and Mark Watkins. Elliptic curves of large rank and small conductor. In *International Algorithmic Number Theory Symposium–ANTS VI*, pages 42–56. Springer, 2004.

[FPRE15] Jean-Pierre Flori, Jérôme Plût, Jean-René Reinhard, and Martin Ekerå. Diversity and transparency for ECC, 2015. Cryptology ePrint Archive.

[Fri49] VR Fridlender. On the least n-th power non-residue. In *Dokl. Akad. Nauk SSSR*, volume 66, pages 351–352, 1949.

[GR09] Steven Galbraith and Raminder S Ruprai. An improvement to the Gaudry-Schost algorithm for multidimensional discrete logarithm problems. In *Cryptography and Coding: 12th IMA International Conference, Cryptography and Coding 2009, Cirencester, UK, December 15-17, 2009. Proceedings 12*, pages 368–382. Springer, 2009.

[HS07] Guillaume Hanrot and Damien Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In *Annual international cryptology conference*, pages 170–186. Springer, 2007.

[Kri20] Daniel Kriz. Supersingular main conjectures, sylvester's conjecture and goldfeld's conjecture. *arXiv preprint arXiv:2002.04767*, 2020.

[KS98] Nicholas M. Katz and Peter Sarnak. Random matrices, frobenius eigenvalues, and monodromy. 1998.

[Lan83] Serge Lang. Conjectured diophantine estimates on elliptic curves. *Arithmetic and Geometry: Papers Dedicated to IR Shafarevich on the Occasion of His Sixtieth Birthday Volume I Arithmetic*, pages 155–171, 1983.

[LO85] Jeffrey C Lagarias and Andrew M Odlyzko. Solving low-density subset sum problems. *Journal of the ACM*, 32(1):229–246, 1985.

[LW08] Yuk-Kam Lau and Jie Wu. On the least quadratic non-residue. *International Journal of Number Theory*, 4(03):423–435, 2008.

[MT19] Kevin McGown and Enrique Trevino. The least quadratic non-residue. *preprint*, 2019.

[Nas16] Bartosz Naskrkecki. Distribution of Mordell–Weil ranks of families of elliptic curves. *arXiv preprint arXiv:1609.04731*, 2016.

[Nat23] National Institute of Science and Thechnology, Computer security resource center. Special publication NIST 800-186: Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters. https://csrc.nist.gov/pubs/sp/800/186/final, February 2023.

[Pas09] Andrzej Paszkiewicz. On the average of the least quadratic non-residue modulo a prime number. *Electronics and Telecommunications Quarterly*, 55(4):639–647, 2009.

[Pil24] Cédric Pilatte. Unconditional correctness of recent quantum algorithms for factoring and computing discrete logarithms. *arXiv preprint arXiv:2404.16450*, 2024.

[Reg23] Oded Regev. An efficient quantum factoring algorithm, 2023. arXiv preprint arXiv:2308.06572.

[RS01] Karl Rubin and Alice Silverberg. Rank frequencies for quadratic twists of elliptic curves. *Experimental Mathematics*, 10(4):559–569, 2001.

[RV23] Seyoon Ragavan and Vinod Vaikuntanathan. Optimizing space in Regev's factoring algorithm, 2023. arXiv preprint arXiv:2310.00899.

[Sal49] Hans Salié. Über den kleinsten positiven quadratischen nichtrest nach einer primzahl. *Mathematische Nachrichten*, 3(1):7–8, 1949.

[Sho94] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science–SFCS'94*, pages 124–134. Ieee, 1994.

[TV06] Terence Tao and Van H. Vu. *Additive combinatorics*, volume 105. Cambridge University Press, 2006.

[Ulm14] Douglas Ulmer. Explicit points on the Legendre curve. *Journal of Number Theory*, 136:165–194, 2014.

[Wan81] Paul S Wang. A p-adic algorithm for univariate partial fractions. In *Proceedings of the fourth ACM symposium on Symbolic and algebraic computation*, pages 212–217, 1981.

[WDE+14] Mark Watkins, Stephen Donnelly, Noam D Elkies, Tom Fisher, Andrew Granville, and Nicholas F Rogers. Ranks of quadratic twists of elliptic curves. *Publications mathematiques de Besancon Algebre et theorie des nombres*, 2:63–98, 2014.

[Zim76] Günter Horst Zimmer. On the difference of the Weil height and the Néron-Tate height. *Mathematische Zeitschrift*, 147:35–51, 1976.

Appendix A. Extending the theorem of Lau and Wu to the primes $\equiv 3 \pmod 4$

For $a = 1$ or 3 we define

$$P_y^{(a)} = \{p : p \equiv a \pmod 4, \text{ and } \chi_p(q) = 1 \text{ for all } q \le y\}.$$

Proposition A.1 (minor extension of Th 1.2 in [LW08])**.** *Let $\delta > 0$ be a constant and let $a = 1$ or 3. There exists a sequence $Q_m \to \infty$ such that*

$$\sum_{\substack{x^{1/2} < p < x \log x \\ p \in P_y^{(a)}}} 1 \gg_m Q_m e^{-c \log Q_m / \log\log Q_m}.$$

$y^2+xy+y=x^3+(t^2+t+1)x^2+(t^3+t^2+t)x+(t^2+t)$	$[(t+1:t:1),(0:t:1)]$
$y^2+xy+y=x^3+x^2+(t^2+t+1)x$	$[(t+1:1:1),(0:1:1)]$

TABLE 5. Examples of elliptic curves over $\mathbb{F}_2(t)$ of rank 2

$(x,y)\in\mathbb{F}_{31}(t)\times\mathbb{F}_{31}(t)$
$(19t^2+2t+6, 15t^3+3t^2+10t+17)$
$(7t^2+16t+6, 23t^3+6t^2+18t+17)$
$(14t^2+4t+6, 27t^3+12t^2+20t+17)$
$(28t^2+t+6, 29t^3+24t^2+5t+17)$
$(2t^2+10t+30, 15t^3+3t^2+10t+17)$

TABLE 6. Elliptic curve $E: y^2=x^3+t^5+11$ of rank 5 for discrete logarithm in $\mathbb{F}_{31^n}$. Among the points $\sum_{i=1}^5[\varepsilon_i]P_i$ with $\varepsilon_i\in\{0,1\}$ all the maximum degree in the form $(x:y:z)$ with $x,y,z\in\mathbb{F}_{31}[t]$ is 9.

Proof. By a verbatim copy of the proofs of Equation (5.5) in [LW08], we have for $a=1$ or 3,

$$\sum_{\substack{x^{1/2}<p<x\log x\\ p\in P_y^{(a)}}} 1 \gg \frac{1}{(\log x)2^{\pi(y)+2}}\sum_{m\in P_y^{(a)}}(S_x(m)+(-1)^aS_x(4m))+O(\frac{x^{1/2}}{\log x})).$$

where S_x is an expression such that

$$S_x(\ell m)=\delta_{\ell m,1}x-O(x^{1-\epsilon}y^2)+O(x^{1/2}\log x),$$

for a constant $\epsilon>0$ which depends on the potential exceptional zeros of the $L(s,\chi)$ and $\delta_{j,1=1}$ if $j=1$ and 0 otherwise. We conclude that for $a=1$ and $a=3$, $P_y^{(a)}$ admits the same lower bound, which is the one of the main theorem of [LW08]. □

APPENDIX B. EXAMPLES OF ELLIPTIC CURVES WITH MANY SMALL POINTS

Table 5 makes a list of elliptic curves whose generators were computed with Magma.

Elliptic curves in characteristic 31. See Table 6 for curves over fields of characteristic 31.

Elliptic curves in characteristic 7 and 11. See Table 7 for curves obtained with Ulmer's work [Ulm14, Example 4.6(3))].

Elliptic curves over $\mathbb{F}_{2^5}(t)$ of rank 30. In [Elk94, proof og Prop 3], Elkies gives explicit generators $P=(x_P,y_P)$ of the Mordell-Weil group of $E/\mathbb{F}_{2^5}(t): y^2+y=x^3+t^{33}$ by setting

$$x_P=a^{-1}t^{12}+a^2t^9+a^5t^6+a^8t^3+c \tag{12}$$

$E/\mathbb{F}_7(t):\ y^2+(2t^4+2t^2)xy=x^3+(5t^6+6t^4+1)x^2+t^8x$
$P_1=(2t^6+4t^5+6t^4+4t^3+2t^2:4t^7+6t^6+t^5+6t^4+4t^3+2t^2:1)$
$P_2=(2t^6+3t^4+2t^2:2t^6+5t^2:1)$
$P_3=(2t^6+3t^5+6t^4+3t^3+2t^2:3t^7+6t^6+6t^5+6t^4+3t^3+2t^2:1)$
$P_1,P_2,P_3\in E(\mathbb{F}_7(t))$
$\max\{\deg(\sum_{i=1}^3\varepsilon_iP_i):\varepsilon\in\{0,1\}^3\}=12$
$E/\mathbb{F}_{11}(t):y^2+(9t^6+8t^4)xy=x^3+(8t^{10}+6t^8+1)x^2+t^{12}x$
$P_1=(3t^{10}+2t^9+8t^7+8t^6+8t^5+2t^3+3t^2:$
$\quad 8t^{13}+3t^{11}+t^{10}+3t^9+3t^8+8t^7+2t^6+8t^5+t^4+9t^3+8t^2:1)$
$P_1=(3t^{10}+4t^9+5t^8+7t^7+6t^6+7t^5+5t^4+4t^3+3t^2:$
$\quad 5t^{13}+3t^{12}+5t^{11}+3t^{10}+9t^9+5t^8+2t^7+4t^6+8t^5+4t^4+4t^3+3t^2:1)$
$P_3=(3t^{10}+2t^8+10t^6+2t^4+3t^2:10t^{12}+6t^{10}+6t^8+7t^6+10t^4+8t^2:1)$
$P_4=(3t^{10}+7t^9+5t^8+4t^7+6t^6+4t^5+5t^4+7t^3+3t^2:$
$\quad 6t^{13}+3t^{12}+6t^{11}+3t^{10}+2t^9+5t^8+9t^7+4t^6+3t^5+4t^4+7t^3+3t^2:1)$
$P_5=(3t^{10}+9t^9+3t^7+8t^6+3t^5+9t^3+3t^2:$
$\quad 3t^{13}+8t^{11}+t^{10}+8t^9+3t^8+3t^7+2t^6+3t^5+t^4+2t^3+8t^2:1)$
$\max\{\deg(\sum_{i=1}^5\varepsilon_iP_i):\varepsilon\in\{0,1\}^5\}=28$

TABLE 7. Examples of curves adapted for Regev in $\mathbb{F}_{p^n}$ for $p=7$ and 11.

and computing y_P using the equation of E. For example, if $\mathbb{F}_{2^5}=\mathbb{F}_2(\omega)$

$$P_1=((\omega^4+\omega)t^{12}+\omega^2t^9+(\omega^2+1)t^6+(\omega^3+\omega^2+1)t^3+\omega^4+\omega^3:$$
$$(\omega^4+\omega^3+\omega^2+1)t^{18}+(\omega^4+\omega^3+1)t^{12}+\omega t^9+(\omega^2+1)t^6+(\omega^2+\omega)t^3+\omega^3+1:1).$$

By direct computations one checks that

$$\max\left\{h\left(\sum_{i=1}^{20}[\varepsilon_i]P_i\right):\varepsilon_i\in\{0,1\}^{20}\,\forall i\right\}=414.$$

$$y^2=x^3+2(t^8+2t^4+1)x-4t^2(t^8-6t^4+1).$$

Naskrkecki's rank 4 elliptic curves over $\mathbb{F}_p(t)$ with $p\equiv 5,7,13,15\pmod{16}$. Naskrkecki [Nas16] gives examples of curves in characteristic 5, 7 and 11, which we list in Table 8.

APPENDIX C. CLASSICAL CONJECTURES THAT SUGGEST THE EXISTENCE OF ASYMPTOTIC FAMILIES OF REGEV-FRIENDLY CURVES

In this appendix we recall a list of conjectures about the existence of infinite families with arbitrarily many small points. Recall that the naive and the canonical heights (see the beginning of Section 3) are equal up to an additive constant, as described in [Zim76, Section 2].

Conjecture C.1 (Lang's conjecture on the heights of points, Conj 3 in [Lan83])**.** *Let E be an elliptic curve defined over $\mathbb{Q}$ or rank τ. Call $H(E)=\max(|a|^3,|b|^2)$*

$E/\mathbb{F}_5(t) : y^2 = x(x-(t^2-1)^2)(x-4t^2)$
$w^2+4w+2=0$
$P_1 = ((3w+3)t^3+(4w+4)t^2+(3w+3)t : (w+1)t^5+(w+3)t^4+(w+2)t^3+(w+3)t^2+(w+1)t : 1)$
$P_2 = (2t^2+t+2 : 2t^4+2t^2+3t+3 : 1)$
$P_3 = ((w+2)t^4+(4w+4)t^3+(w+2)t^2+3wt : (w+3)t^6+4wt^5+(3w+3)t^4+(3w+3)t^3+(2w+2)t^2+4w$
$P_4 = (t^4+4t^2 : t^5+4t^3 : 1)$
$\max\{\deg(\sum_{i=1}^4[\varepsilon_i]P_i) : \varepsilon \in \{0,1\}^4\} = 7$
$E/\mathbb{F}_7(t) : y^2 = x(x-(t^2-1)^2)(x-4t^2)$
$w^2-3=0$
$P_1 = (t^3+5t^2+t : (6w+4)t^5+(w+3)t^4+(w+3)t^2+(6w+4)t : 1)$
$P_2 = (2t^2+3t+2 : 2t^4+6t^2+t+5 : 1)$
$P_3 = (t^2+t : (5w+1)t^4+(4w+5)t^3+(5w+1)t^2+(6w+4)t : 1)$
$P_4 = (1 : 2t^3+3t : 1)$
$\max\{\deg(\sum_{i=1}^4[\varepsilon_i]P_i) : \varepsilon \in \{0,1\}^4\} = 7$

TABLE 8. Naskrkecki's rank 4 elliptic curves over $\mathbb{F}_p(t)$ with $p \equiv 5, 7, 13, 15 \pmod{16}$

the height of E and let $\hat{h}$ be the Néron-Tate height. There exists a basis $P_1, \ldots, P_\tau$ of $E(\mathbb{Q})$ divided by the $\mathbb{Q}$-torsion such that $\hat{h}(P_1) \leq \cdots \leq \hat{h}(P_\tau)$ and for one has

$$\hat{h}(P_1) \leq H(E)^{1/(12\tau)} N^{\epsilon(N)/\tau} \log(N) c^{(r-1)/2},$$
$$\hat{h}(P_\tau) \leq H(E)^{1/12} N^{\epsilon(N)} \log(N) c^{\tau(\tau-1)/2},$$

with c an universal constant[8] *and $\epsilon(N)$ is of the form $\epsilon(N) = c' \cdot (\log(N)\log\log(N))^{-1/2}$. Here $N = N_E$ is the conductor of E.*

Note that the statement of the conjecture cannot be significantly improved, see e.g. [Dav97].
Since the previous conjecture makes use of $H(E)$ it is worth mentioning Hall's conjecture.

Conjecture C.2 (Hall's conjecture, page 160 of [Lan83])**.** *$H(E) \ll \Delta(E)^{6+\epsilon}$. In particular, E has a minimal model with $\log H(E) \ll \log N(\log\log N)^2$.*

Finally, we have a conjecture about the size of τ.

Conjecture C.3 ([EW04] summary of the heuristic discussion in Sec 5)**.** *The maximum rank among the elliptic curves of conductor less than N is equal to $c(\frac{\log N}{\log\log N})^\alpha$ for two constants $c > 0$ and $\alpha = 1$ or $1/2$.*

APPENDIX D. A MULTI-TWIST VARIANT OF FAST-EXPONENTIATION

For completeness we reproduce here the multi-scalar variant of fast-exponentiation.

Algorithm 3 is a minor modification of Algorithm 2. We write it here for a precise statement of the parameters.

[8] We don't use the exact value of c which can be taken equal to $c = 2/\sqrt{3}$ and similarly for c'.

Algorithm 2 Fast multi-scalar exponentiation

Input:
- a commutative group $\mathbb{G}$ with elements represented on $O(m)$ bits having an n-bit prime order r
- a parameter $d \leq n$
- a parameter $T > 1$
- a constant k and $u_1, \ldots, u_k \in \mathbb{G}$ or an algorithm, called tab, to compute them on the fly in negligible complexity
- $g_1, \ldots, g_{d-k} \in \mathbb{G}$ which have coordinates represented on at most $O(d)$ bits
- $z_i = \sum_{j=0}^{\lceil \log_2 T \rceil - 1} z_{i,j} 2^j$ for $i = 1, 2, \ldots, d$ integers of $\lceil \log_2 T \rceil$ bits

Output: $\prod_{i=1}^{d} g_i^{z_i}$

result $= 1_{\mathbb{G}}$
for $j = 0, 1, \ldots, \lceil \log_2 T \rceil$ **do**
  result $=$ result $*$ result
  result $=$ result $*$ tab$(z_{1,j}, \ldots, z_{d-k,j})$
  result $=$ result $* u_1^{z_{1,j}} * \ldots * u_k^{z_{k,j}}$
end for
return result

Algorithm 3 A multi-twist variant of fast multi-scalar exponentiation

Input:
- an elliptic curve E defined over a finite field $\mathbb{F}_q$ where q has m bits
- a parameter $d \leq n$
- a parameter $T > 1$
- an integer k and a list of points $U_1, \ldots, U_k$ on E
- a list of integers $d_1 = 1, d_2, \ldots, d_t$ for a parameter $t \geq 0$
- and, for each $1 \leq i \leq t$ a list of rational points $P_{i,1} = (x_{i,1}, y_{i,1}), \ldots, P_{i,r_i} = (x_{i,r_i}, y_{i,r_i})$ on the d_i-twist of E, such that $r_1 + \cdots + r_t = d - k$
- $(g_1, \ldots, g_{d-k}) = \big((x_{1,1}, \sqrt{d_1} y_{1,1}), \ldots, (x_{1,r_1}, \sqrt{d_1} y_{1,r_1}), (x_{2,1}, \sqrt{d_2} y_{2,1}), \ldots, (x_{t,r_t}, \sqrt{d_t} y_{t,r_t})\big)$
- a list of $\lceil \log_2 T \rceil$-bits integers $(z_1, \ldots, z_d) = (z_{1,1}, \ldots, z_{1,r_1}, z_{2,1}, \ldots, z_{t,r_t})$. We denote $z_{\mu,\nu,j}$ the j-th bit of $z_{\mu,\nu}$.
- a flag "look-up table" = true/false to decide whether the sums of points with coefficients 0 and 1 are stored in a look-up table or are recomputed on the fly whenever needed.

Output: $\sum_{i=1}^{d-k} [z_i] g_i + \sum_{\mu=1}^{k} [z_{d-k+\mu}] U_\mu$

result $= \mathcal{O}_E$
for $j = 0, 1, \ldots, \lceil \log_2 T \rceil$ **do**
  result $=$ result $+$ result
  for $\mu = 1, \ldots, t$ **do**
    $Q_\mu = (X_\mu, Y_\mu) = \sum_{\nu=1}^{r_\mu} [z_{\mu,\nu,j}] P_{\mu,\nu}$ ▷ Computations done in the d_μ-twist of E
    $Q'_\mu = (X_\mu, \sqrt{\mu} Y_\mu)$ ▷ Q'_μ belongs to E
    result $=$ result $+ Q'_\mu$
  end for
  result $=$ result $+ \sum_{\nu=1}^{k} [z_{d-k+\nu,j}] U_\nu$
end for
return result