Designs, Codes and Cryptography (2020) 88:2313–2335
https://doi.org/10.1007/s10623-020-00783-2

On post-processing in the quantum algorithm for computing short discrete logarithms

Martin Ekerå[1,2]

Received: 31 March 2020 / Revised: 24 June 2020 / Accepted: 29 June 2020 / Published online: 6 August 2020

Abstract
We revisit the quantum algorithm for computing short discrete logarithms that was recently introduced by Ekerå and Håstad. By carefully analyzing the probability distribution induced by the algorithm, we show its success probability to be higher than previously reported. Inspired by our improved understanding of the distribution, we propose an improved post-processing algorithm that is considerably more efficient, enables better tradeoffs to be achieved, and requires fewer runs, than the original post-processing algorithm. To prove these claims, we construct a classical simulator for the quantum algorithm by sampling the probability distribution it induces for given logarithms. This simulator is in itself a key contribution. We use it to demonstrate that Ekerå–Håstad achieves an advantage over Shor, not only in each individual run, but also overall, when targeting cryptographically relevant instances of RSA and Diffie–Hellman with short exponents.

Keywords Discrete logarithms · Factoring · RSA · Shor's algorithms

Mathematics Subject Classification 68Q12 · 81P68 · 94A60

1 Introduction

As in [4,5], let $\mathbb{G}$ under $\odot$ be a group of order r generated by g, and let

$$x = [d]\, g = \underbrace{g \odot g \odot \cdots \odot g}_{d \text{ times}}.$$

Given g and x the discrete logarithm problem (DLP) is to compute $d = \log_g x$. In the general DLP $0 \leq d < r$, whereas d is smaller than r by some order of magnitude in the

Communicated by D. Stebila.

✉ Martin Ekerå
ekera@kth.se

1 KTH Royal Institute of Technology, 100 44 Stockholm, Sweden

2 Swedish NCSA, Swedish Armed Forces, 107 85 Stockholm, Sweden

short DLP. The order r may be assumed to be known or unknown. Both cases are cryptographically relevant.

1.1 Earlier works

In 2016 Ekerå [4] introduced a modified version of Shor's algorithm [16,17] for computing discrete logarithms that is more efficient than Shor's original algorithm when the logarithm is short. This work was originally motivated by the use of short discrete logarithms in instantiations of cryptographic schemes based on the computational intractability of the DLP in finite fields. A concrete example is the use of short exponents in the Diffie–Hellman key exchange protocol when instantiated with safe-prime groups.

This work was subsequently generalized by Ekerå and Håstad [5] so as to enable tradeoffs between the number of times that the algorithm needs to be executed, and the requirements it imposes on the quantum computer. These ideas parallel earlier ideas by Seifert [15] for making tradeoffs in Shor's order finding algorithm; the quantum part of Shor's general factoring algorithm.

Ekerå and Håstad furthermore explained how the RSA integer factoring problem may be expressed as a short DLP. This gives rise to a new algorithm for factoring RSA integers that does not rely directly on order-finding, and that imposes less requirements on the quantum computer in each run compared to Shor's or Seifert's general factoring algorithms.

1.2 Background on quantum cost metrics

The hard part when implementing Shor's algorithms and their various derivatives in practice is to exponentiate group elements in superposition. The algorithms of Seifert, Ekerå and Ekerå–Håstad reduce the requirements on the quantum computer compared to Shor by reducing the exponent length in each run.

In virtually all practical implementations, an exponent length reduction translates into a reduction in the number of controlled group operations that need to be evaluated quantumly.[1] This in turn translates into a reduction in the number of logical qubit operations, the circuit depth, the runtime and the required coherence time. The number of logical qubits required is typically not reduced, however. This is because control qubits can be recycled [10,11] in implementations.

When accounting for the need for quantum error correction, the number of physical qubits required may nevertheless be reduced as a result of reducing the exponent length, and it is physical rather than logical qubit counts that matter in practice. To understand why physical qubits may be saved, note that quantum error correcting codes, such as the surface code (see [6] for a good overview), use an array of physical qubits to construct each logical qubit. In simple terms; the fewer the number of operations that need be performed with respect to a logical qubit, for some fixed maximum tolerable probability of uncorrectable errors arising, the fewer the number of physical qubits needed to construct said logical qubit.

Quantum spacetime volume, the product of the runtime and the number of physical qubits required, is sometimes used as a complexity metric in the literature. For the reasons explained above, a reduction in the exponent length typically gives rise to a reduction in the spacetime volume in each run, and in some cases overall.

[1] Square-and-multiply (or double-and-add) exponentiation is typically employed, with classically precomputed powers (or multiples) of the group element to exponentiate.

1.3 Our contributions

To compute an m bit short discrete logarithm d, the quantum algorithm of Ekerå and Håstad exponentiates group elements in superposition to $m + 2m/s$ bit exponents for $s \geq 1$ an integer. Given a set of s good outputs, the classical post-processing algorithm in [5] recovers d by enumerating vectors in an $s + 1$-dimensional lattice. The probability of observing a good output in a run is lower bounded by $1/8$ in [5], so we expect to find s good outputs after at most about $8s$ runs. As good outputs cannot be efficiently recognized, d is recovered in [5] by exhaustively post-processing all subsets of s outputs from a set of about $8s$ outputs.

We refer to s as the tradeoff factor, as it controls the tradeoff between the number of runs required and the exponent length in each run. To minimize the exponent length we would like to select s large. However, the fact that the post-processing complexity grows exponentially in s limits the achievable tradeoff. Furthermore, the fact that we expect to have to perform about $8s$ runs implies that an overall reduction in the number of group operations performed quantumly may not be achieved, even if the number of operations in each run is reduced.

In this work, we analyze and capture the probability distribution induced by the quantum algorithm of Ekerå and Håstad. This analysis allows us to tightly estimate the probability of observing a good output, and to show that it is significantly higher than the lower bound of $1/8$ in [5] guarantees.

More importantly, the fact that we can capture the probability distribution induced by the quantum algorithm of Ekerå and Håstad for known d enables us to classically simulate the quantum algorithm by sampling the distribution. This simulator is in itself a key contribution. Inspired by our improved understanding of the distribution, we design an improved and considerably more efficient classical post-processing algorithm. We then use the simulator to heuristically estimate the number of runs required to solve for d for different tradeoff factors, and to practically verify the heuristic estimates by post-processing simulated outputs.

With our new post-processing and analysis, Ekerå–Håstad's algorithm achieves an advantage over Shor's algorithms, not only in each individual run, but also overall, when targeting cryptographically relevant instances of RSA and Diffie–Hellman with short exponents. When making tradeoffs, Ekerå–Håstad's algorithm with our new post-processing furthermore outperforms Seifert's algorithm in practice when factoring RSA integers. To the best of our knowledge, this makes Ekerå–Håstad's algorithm the currently most efficient polynomial time quantum algorithm for breaking the RSA cryptosystem [12], and DLP-based cryptosystems such as Diffie–Hellman [3] when instantiated with safe-prime groups with short exponents, as in TLS [7], IKE [8] and standards such as NIST SP 800-56A [2].

For example, compared to Shor's original algorithms, Ekerå–Håstad can achieve a reduction in the number of group operations that are evaluated quantumly in each run, of a factor 6.1 or 14.2 for FF-DH-2048 in safe-prime groups with 224 bit exponents, and of a factor 1.35 or 3.6 for RSA-2048, depending on whether tradeoffs are made. When not making tradeoffs, a single run of Ekerå–Håstad in general suffices. For other options and technical details, please see Appendix A.

Both RSA and Diffie–Hellman are widely deployed cryptosystems. It is important to understand the complexity of attacking these systems quantumly to project the remaining lifetime of confidential information protected by these systems, and to inform migration strategies. This paper helps further this understanding.

1.4 Overview

The remainder of this paper is structured as follows:

We recall the quantum algorithm of Ekerå and Håstad in Sect. 2 and we analyze the probability distribution it induces in Sects. 3–4. In particular, we derive a closed form expression for the probability of observing a given output.

In Sect. 5 we use the closed form expression to generate a high resolution histogram for the probability distribution, and describe how it may be sampled to classically simulate the quantum algorithm, for known d. In Sect. 6 we introduce our improved post-processing algorithm. We furthermore use the simulator to heuristically estimate and experimentally verify the number of runs required to solve for d. We summarize and conclude the paper in Sect. 7. Concrete cost estimates for RSA and Diffie–Hellman are given in Appendix A.

1.5 Notation

Before proceeding, we introduce notation used throughout this paper:

- $u \bmod n$ denotes u reduced modulo n constrained to $0 \leq u \bmod n < n$.
- $\{u\}_n$ denotes u reduced modulo n constrained to $-n/2 \leq \{u\}_n < n/2$.
- $\lceil u \rceil$, $\lfloor u \rfloor$ and $\lceil u \rfloor$ denotes u rounded up, down and to the closest integer.
- $|a + ib| = \sqrt{a^2 + b^2}$ where $a, b \in \mathbb{R}$ denotes the Euclidean norm of $a + ib$.
- $|\,\mathbf{u}\,|$ denotes the Euclidean norm of the vector $\mathbf{u} = (u_0, \ldots, u_{n-1}) \in \mathbb{R}^n$.
- $u \lll v$ is used to denote that u is less than v by some order of magnitude.
- $u \sim v$ is used to denote that u and v are approximately of similar size.

2 The quantum algorithm

Recall that given a generator g of a finite cyclic group of order r and $x = [d]g$, where d is such that $0 < d < 2^m \lll r$, the quantum algorithm of Ekerå and Håstad in [5] computes the logarithm $d = \log_g x$ by inducing the system

$$\frac{1}{2^{2\ell+m}} \sum_{a,\,j=0}^{2^{\ell+m}-1} \sum_{b,k=0}^{2^{\ell}-1} \exp\left[\frac{2\pi i}{2^{\ell+m}}\,(aj + 2^m bk)\right] |\, j, k, [e = a - bd]g\,\rangle \tag{1}$$

where $\ell \sim m/s$ is an integer for $s \geq 1$ the tradeoff factor. Observing (1) yields a pair (j, k) for j and k integers on the intervals $0 \leq j < 2^{\ell+m}$ and $0 \leq k < 2^{\ell}$, respectively, and some group element $y = [e]g \in \mathbb{G}$.

3 The probability distribution

The quantum system in (1) collapses to $y = [e]g$ and (j, k) with probability

$$\frac{1}{2^{2(2\ell+m)}} \left| \sum_{a} \sum_{b} \exp\left[\frac{2\pi i}{2^{\ell+m}}\,(aj + 2^m bk)\right] \right|^2$$

where the sums are over all a and b such that $e \equiv a - bd \pmod r$. Assuming that $r \geq 2^{\ell+m} + (2^\ell - 1)d$, this simplifies to $e = a - bd$. Summing over all e, we obtain the probability P of observing the pair (j, k) over all e as

$$P = \frac{1}{2^{2(2\ell+m)}} \sum_e \left| \sum_{b=b_0(e)}^{b_1(e)-1} \exp\left[\frac{2\pi i}{2^{\ell+m}} ((e+bd)j + 2^m bk) \right] \right|^2$$

$$= \frac{1}{2^{2(2\ell+m)}} \sum_e \left| \sum_{b=b_0(e)}^{b_1(e)-1} \exp\left[\frac{2\pi i}{2^{\ell+m}} b\{dj + 2^m k\}_{2^{\ell+m}} \right] \right|^2$$

where we have introduced the functions $b_0(e)$ and $b_1(e)$ that determine the length of the summation interval in b. For more information, see Sect. 3.1.

The above analysis implies that the probability P of observing a given pair (j, k) is determined by its argument α or, equivalently, by its angle θ, where

$$\alpha(j,k) = \{dj + 2^m k\}_{2^{\ell+m}} \quad \text{and} \quad \theta(\alpha) = \frac{2\pi\alpha}{2^{\ell+m}}$$

and the probability

$$P(\theta) = \frac{1}{2^{2(2\ell+m)}} \sum_e \left| \sum_{b=b_0(e)}^{b_1(e)-1} \mathrm{e}^{i\theta b} \right|^2 = \frac{1}{2^{2(2\ell+m)}} \sum_e \underbrace{\left| \sum_{b=0}^{\#b(e)-1} \mathrm{e}^{i\theta b} \right|^2}_{\zeta(\theta, \#b(e))}$$

where $\#b(e) = b_1(e) - b_0(e)$. If $\theta = 0$, then

$$\zeta(0, \#b(e)) = \left| \sum_{b=0}^{\#b(e)-1} \mathrm{e}^{i\theta b} \right|^2 = \left| \sum_{b=0}^{\#b(e)-1} 1 \right|^2 = (\#b(e))^2.$$

Otherwise, if $\theta \neq 0$ then

$$\zeta(\theta, \#b(e)) = \left| \sum_{b=0}^{\#b(e)-1} \mathrm{e}^{i\theta b} \right|^2 = \left| \frac{\mathrm{e}^{i\theta\,\#b(e)} - 1}{\mathrm{e}^{i\theta} - 1} \right|^2 = \frac{1 - \cos(\theta\,\#b(e))}{1 - \cos\theta}.$$

3.1 The summation interval for a given *e*

Recall that $e = a - bd$ where $0 < d < 2^m$, $0 \leq a < 2^{\ell+m}$ and $0 \leq b < 2^\ell$. This implies that e is an integer on the interval

$$-(2^\ell - 1)d \leq e = a - bd < 2^{\ell+m}. \tag{2}$$

Divide the interval for e into three regions, and denote these regions A, B and C, respectively. Define the middle region B to be the region in e where $\#b(e) = 2^\ell$ or equivalently $0 = b_0(e) \leq b < b_1(e) = 2^\ell$. Then by (2) region B spans the interval $0 \leq e < 2^{\ell+m} - (2^\ell - 1)d$. This is easy to see, as for all b on the interval $0 \leq b < 2^\ell$ there must exist an a on $0 \leq a < 2^{\ell+m}$ such that $e = a - bd$.

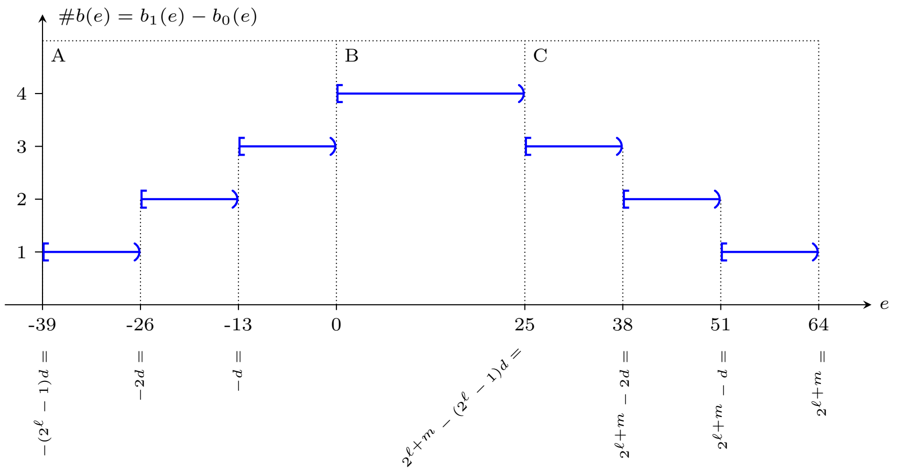

Fig. 1 The interval length $\#b(e) = b_1(e) - b_0(e)$ for $m = 4$, $\ell = 2$ and $d = 13$. Similar staircase-like plots arise irrespective of how the parameters are selected

It follows that region A to the left of B spans $-(2^\ell - 1)d \le e < 0$ and that region C to the right of B spans $2^{\ell+m} - (2^\ell - 1)d \le e < 2^{\ell+m}$. Regions A and C are hence both of length $(2^\ell - 1)d$. Regions A and C are furthermore both divided into $2^\ell - 1$ plateaus of length d, as there is, for each multiple of d that is subtracted from e in these regions, one fewer values of b for which there exists an a such that $e = a - bd$. The situation that arises is depicted in Fig. 1.

3.2 The probability of observing (*j*, *k*) summed over all *e*

We may now sum $\zeta(\theta, \#b(e))$ over all e to express

$$P(\theta) = \frac{1}{2^{2(2\ell+m)}} \sum_{e=-(2^\ell-1)d}^{2^{\ell+m}-1} \zeta(\theta, \#b(e))$$

on closed form by first using that if the sum is split into partial sums over the regions A, B and C, respectively, it follows from the previous two sections that the sums over regions A and C must yield the same contribution.

Furthermore, region B is rectangular, and each plateau in regions A and C is rectangular. Closed form expressions for the contribution from these rectangular regions may be trivially derived. This implies that

$$2^{2(2\ell+m)} P(\theta) = \underbrace{\sum_{e=-(2^\ell-1)d}^{-1} \zeta(\theta, \#b(e))}_{\text{region A}} + \underbrace{\sum_{e=0}^{2^{\ell+m}-(2^\ell-1)d-1} \zeta(\theta, \#b(e))}_{\text{region B}}$$
$$+ \underbrace{\sum_{e=2^{\ell+m}-(2^\ell-1)d}^{2^{\ell+m}-1} \zeta(\theta, \#b(e))}_{\text{region C}}$$

$$= \underbrace{(2^{\ell+m} - (2^\ell - 1)d) \cdot \zeta(\theta, 2^\ell)}_{\text{region B}} + \underbrace{2d \cdot \sum_{\#b=1}^{2^\ell-1} \zeta(\theta, \#b)}_{\text{regions A and C}}.$$

If $\theta = 0$ then

$$\sum_{\#b=1}^{2^\ell-1} \zeta(0, \#b) = \sum_{\#b=1}^{2^\ell-1} (\#b)^2 = \frac{2^\ell}{6}(2^\ell - 1)(2^{\ell+1} - 1)$$

so the probability of observing the pair (j, k) over all e is

$$P(0) = \frac{1}{2^{2(2\ell+m)}} \cdot \left((2^{\ell+m} - (2^\ell - 1)d) \cdot 2^{2\ell} + \frac{2^\ell d}{3}(2^\ell - 1)(2^{\ell+1} - 1) \right).$$

Otherwise, if $\theta \neq 0$ then

$$\sum_{\#b=1}^{2^\ell-1} \zeta(\theta, \#b) = \sum_{\#b=1}^{2^\ell-1} \frac{1 - \cos(\theta \cdot \#b)}{1 - \cos\theta}$$
$$= \frac{1}{1 - \cos\theta} \left[(2^\ell - 1) - \frac{1}{2} \left(\frac{\cos((2^\ell - 1)\theta) - \cos 2^\ell\theta}{1 - \cos\theta} - 1 \right) \right]$$

so the probability of observing the pair (j, k) over all e is

$$P(\theta) = \frac{1}{2^{2(2\ell+m)}} \cdot \frac{1}{1 - \cos\theta} \cdot \Bigg((2^{\ell+m} - (2^\ell - 1)d) \cdot (1 - \cos 2^\ell\theta)$$
$$+ 2d \cdot \left[(2^\ell - 1) - \frac{1}{2} \left(\frac{\cos((2^\ell - 1)\theta) - \cos 2^\ell\theta}{1 - \cos\theta} - 1 \right) \right] \Bigg).$$

3.3 The distribution of pairs (j, k) with argument α

We now proceed to analyze the distribution and multiplicity of angles θ, or equivalently of arguments α, to complete our analysis.

Definition 1 Let κ denote the greatest integer such that 2^κ divides d.

Definition 2 An argument α is said to be admissible if there exists $(j, k) \in \mathbb{Z}^2$ where $0 \leq j < 2^{\ell+m}$ and $0 \leq k < 2^\ell$ such that $\alpha = \{dj + 2^m k\}_{2^{\ell+m}}$.

Claim 1 *All admissible arguments $\alpha = \{dj + 2^m k\}_{2^{\ell+m}}$ are multiples of 2^κ.*

Proof As $2^\kappa \mid d < 2^m$ and the modulus is a power of two the claim follows. □

Lemma 1 *There are $2^{\ell+m-\kappa}$ admissible α that each occurs with multiplicity $2^{\ell+\kappa}$.*

Proof Let $d = 2^\kappa d'$. Fix some α' on $0 \leq \alpha' < 2^{\ell+m-\kappa}$. For any k with $0 \leq k < 2^\ell$ the equivalence $\alpha' \equiv d'j' + 2^{m-\kappa}k \pmod{2^{\ell+m-\kappa}}$ has a unique solution in j' with $0 \leq j' < 2^{\ell+m-\kappa}$. For any such j', there are 2^κ values of j on $0 \leq j < 2^{\ell+m}$ with $j \equiv j' \pmod{2^{l+m-\kappa}}$. Hence there are $2^{\ell+\kappa}$ pairs (j, k) with $\alpha = \{dj + 2^m k\}_{2^{\ell+m}}$. By Claim 1, these are the only admissible α, and so the lemma follows. □

3.4 The probability of observing a pair (j, k) with argument α

It follows from Lemma 1 that the probability $\Phi(\theta(\alpha))$ of observing one of the pairs (j, k) with argument α is $\Phi(\alpha) = N(\alpha) \cdot P(\theta(\alpha))$ where the multiplicity

$$N(\alpha) = \begin{cases} 2^{\ell+\kappa} & \text{if } \alpha \text{ is admissible} \\ 0 & \text{otherwise.} \end{cases}$$

4 Understanding the probability distribution

In this section, we introduce the notion of t-good pairs, prove upper bounds on the probability of obtaining a t-good pair, and show how the pairs are distributed as a function of t to build an understanding for the distribution.

Definition 3 A pair (j, k) is said to be t-good, for t an integer, if

$$\begin{cases} 2^{t-1} \leq |\,\alpha(j,k)\,| < 2^t & \text{when } 0 < t < \ell + m \\ \alpha(j,k) = 0 & \text{when } t = 0. \end{cases}$$

Definition 4 Let $\rho(t)$ denote the probability of observing a t-good pair.

Lemma 2 *For $0 < t < \ell + m$, the probability $\rho(t) \leq 2^{1-|m-t|}$.*

Proof For $0 < t < \ell + m$, the probability

$$\rho(t) = \sum_{\substack{(j,k):\\ 2^{t-1} \leq |\,\alpha(j,k)\,| < 2^t}} \Phi(\alpha(j,k)) = \frac{1}{2^{2(2\ell+m)}} \sum_{2^{t-1} \leq |\,\alpha\,| < 2^t} N(\alpha) \sum_e \zeta(\theta(\alpha), \#b(e)) \quad (3)$$

where the first sum is over all pairs (j, k) such that $0 \leq j < 2^{\ell+m}$, $0 \leq k < 2^\ell$ and $2^{t-1} \leq |\,\alpha(j,\ k)\,| < 2^t$. The second sum is over at most $2^{t-\kappa}$ admissible arguments α that each occurs with multiplicity $N(\alpha) = 2^{\ell+\kappa}$. The third sum is over at most $2^{\ell+m+1}$ values of e. It therefore follows from (3) that

$$\rho(t) \leq \underbrace{2^{-2(2\ell+m)} \cdot 2^{t-\kappa} \cdot 2^{\ell+\kappa} \cdot 2^{\ell+m+1}}_{= 2^{1-2l-m+t}} \max_{\substack{e,\,\alpha:\\ 2^{t-1} \leq |\,\alpha\,| < 2^t}} \zeta(\theta(\alpha), \#b(e)). \quad (4)$$

As $\zeta(\theta, \#b(e)) \leq 2^{2\ell}$, it follows immediately from (4) that

$$\rho(t) \leq 2^{1-2l-m+t} \cdot 2^{2\ell} = 2^{1-(m-t)}. \quad (5)$$

Furthermore, as $1 - \cos(\theta \cdot \#b(e)) \leq 2$, and as $1 - \cos\theta \geq 2(\theta/\pi)^2$ for $|\,\theta\,| \leq \pi$,

$$\zeta(\theta, \#b(e)) = \frac{1 - \cos(\theta \cdot \#b(e))}{1 - \cos\theta} \leq \frac{\pi^2}{\theta^2}. \quad (6)$$

As $|\,\theta(\alpha)\,| = 2\pi\,|\,\alpha\,|/2^{\ell+m} \geq 2^t\pi/2^{\ell+m}$, it follows from (4) and (6) that

$$\rho(t) \leq 2^{1-2l-m+t} \cdot \frac{\pi^2}{\theta^2} \leq 2^{1-2l-m+t} \cdot \frac{\pi^2}{(2^t\pi/2^{\ell+m})^2} = 2^{1+(m-t)} \quad (7)$$

and so, by combining (5) and (7), the lemma follows. □

Corollary 1 *The probability of observing a t-good (j, k) for t such that $|\,m - t\,| \leq \Delta$, and Δ some positive integer, is lower-bounded by $1 - 2^{2-\Delta} - 2^{1-m+\kappa}$.*

Proof The probability of observing a t-good (j, k) for $t > m + \Delta$ is

$$\sum_{t=\Delta+m+1}^{\ell+m-1} \rho(t) \le \sum_{t=\Delta+m+1}^{\ell+m-1} 2^{1+m-t} = \sum_{t=\Delta+1}^{\ell-1} 2^{1-t} \le \sum_{t=\Delta+1}^{\infty} 2^{1-t} = 2^{1-\Delta},$$

while the probability of observing a t-good (j, k) for $0 < t < m - \Delta$ is

$$\sum_{t=1}^{m-\Delta-1} \rho(t) \le \sum_{t=1}^{m-\Delta-1} 2^{1+t-m} \le \sum_{t=-\infty}^{m-\Delta-1} 2^{1+t-m} = \sum_{t=-\infty}^{-\Delta-1} 2^{1+t} = 2^{1-\Delta},$$

where we have used Lemma 2, and that the sum is a geometric series. Finally, the probability of observing a t-good (j, k) such that $t = 0$ is

$$\rho(0) = \frac{1}{2^{2(2\ell+m)}} N(0) \sum_{e} \zeta(0, \#b(e)) \le \frac{2^{\ell+\kappa} \cdot 2^{\ell+m+1} \cdot 2^{2\ell}}{2^{2(2\ell+m)}} = 2^{1-m+\kappa},$$

where we have used that $N(0) = 2^{\ell+\kappa}$, that the sum is over at most $2^{\ell+m+1}$ values of e, and that $\zeta(0, \#b(e)) \le 2^{2\ell}$, as in the proof of Lemma 2. Hence, the probability of observing a t-good (j, k) such that $|\, m - t \,| \le \Delta$ is $1 - 2 \cdot 2^{1-\Delta} - 2^{1-m+\kappa}$, where we have used that $\rho(t)$ sums to one, and so the corollary follows. $\square$

Recall that $\rho(t)$ is a probability distribution, and hence normalized, whilst Lemma 2 states that $\rho(t)$ is exponentially suppressed as $|\, m - t \,|$ grows large for $0 < t < \ell + m$. The probability mass is hence concentrated on the t-good pairs for $t \sim m$, except for artificially large κ as $t = 0$ then attracts a non-negligible fraction of the probability mass. This is formally stated in Corollary 1.

This implies that each pair (j, k) observed yields $\sim \ell$ bits of information on d, as we expect $|\, \alpha(j, k) = \{dj + 2^m k\}_{2^{\ell+m}} \,| \sim 2^m$, whilst for random (j, k) we expect $|\, \alpha(j, k) \,| \sim 2^{\ell+m}$. The situation that arises for different d is visualized in Fig. 2.

In the region denoted A in Fig. 2, d is odd and hence κ is zero. As d decreases from $2^m - 1$ to $2^{m-1} + 1$, the probability mass is redistributed towards slightly smaller values of t. This reflects the fact that it becomes easier to solve for d as d decreases in size in relation to 2^m. The redistributive effect weakens if d is further decreased: The histogram in region B where $d = 2^{m-5} - 1$ is roughly representative of the distribution that arises when d is smaller than 2^m by a few orders of magnitude. Further decreasing d has no significant effect.

In the region denoted C in Fig. 2, d is selected to be divisible by a great power of two to illustrate the redistribution that occurs when κ is close to maximal. As the admissible arguments are multiples of 2^κ, it follows that $\rho(t) = 0$ for $0 < t < \kappa$, so the corresponding histogram entries are forced to zero and the probability mass redistributed. This effect is only visible for artificially large κ. It does not arise in cryptographic applications where d may be presumed to be random.

Recall that [5] introduces the notion of a "good" pair (j, k), which in our vocabulary is a pair (j, k) such that $|\, \alpha(j, k) \,| \le 2^{m-2}$, and that it establishes a lower bound of $1/8$ on the probability of observing a good pair. The quantum algorithm is therefore run $8s$ times and all subsets of s outputs post-processed.

Compared to the original analysis in [5], our new analysis is much more precise: It indicates that the probability of observing a good pair is significantly higher that the $1/8$ bound guarantees. Empirically, it is at least $3/10$ for random d. This immediately gives rise to an efficiency gain when exhausting subsets.

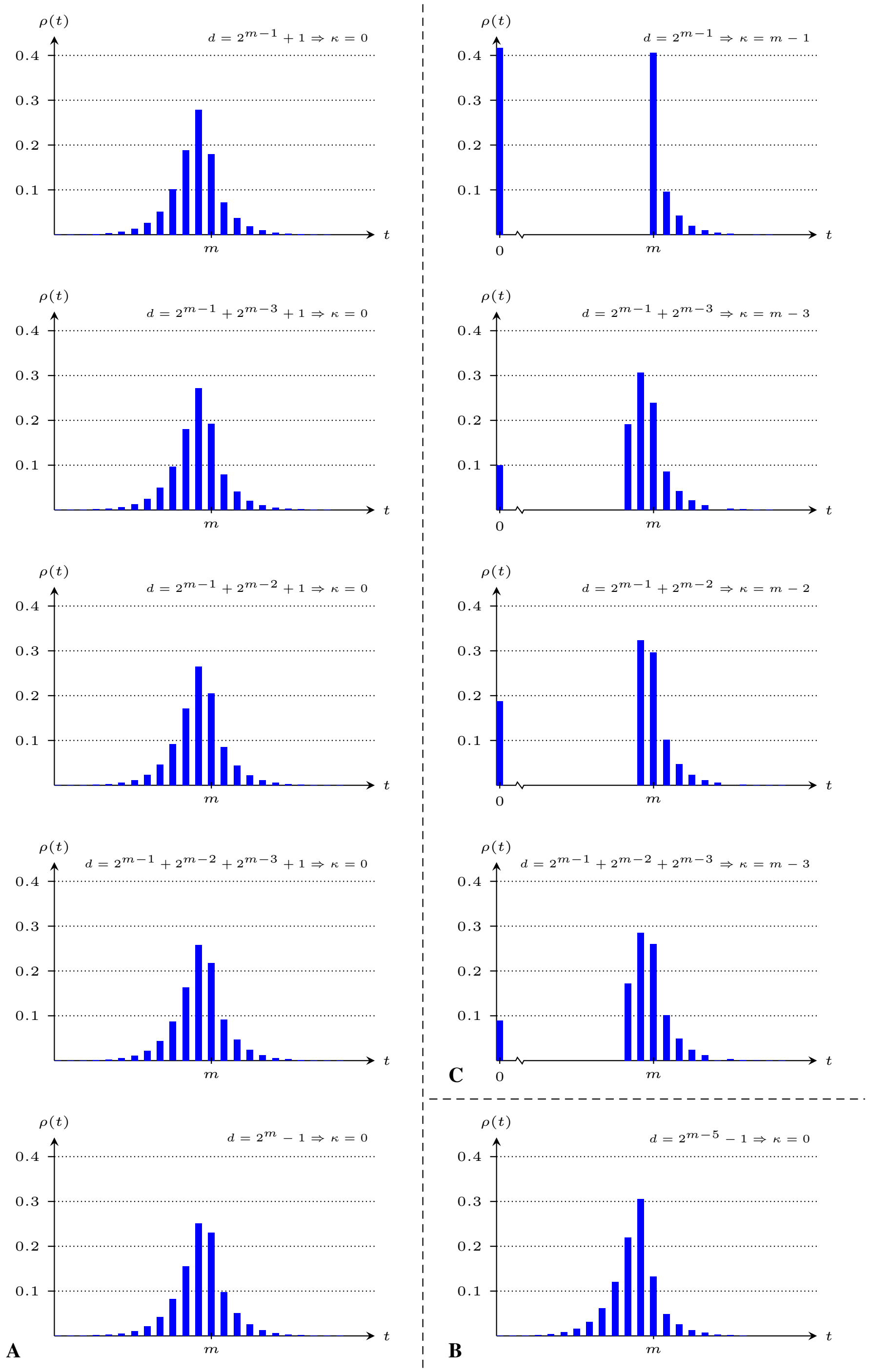

Fig. 2 Histograms for $\rho(t)$ for various d. The histograms were computed for $m = \ell = 256$. Virtually identical histograms arise for any choice of sufficiently large m and ℓ

More importantly, as is stated formally in Corollary 1, our new analysis shows that essentially all observed pairs (j, k) are "somewhat good", in that they yield $\sim \ell$ bits of information on d. This fact enables us to altogether forego exhausting subsets, giving rise to the improved post-processing algorithm in Sect. 6.

5 Simulating the quantum algorithm

In this section, we combine results from the previous sections to construct a high-resolution histogram for the probability distribution for a given d. We furthermore describe how the histogram may be sampled to simulate the quantum algorithm.

5.1 Constructing the histogram

To construct the high-resolution histogram, we divide the argument axis into regions and subregions and integrate $\Phi(\alpha)$ numerically in each subregion.

First, we subdivide the negative and positive sides of the argument axis into $30+\mu$ regions where $\mu = \min(\ell-2, 11)$. Each region thus formed may be uniquely identified by an integer η by requiring that for all α in the region

$$2^{|\eta|} \leq |\,\alpha\,| \leq 2^{|\eta|+1} \quad \text{and} \quad \operatorname{sgn}(\alpha) = \operatorname{sgn}(\eta)$$

where $m-30 \leq |\,\eta\,| < m+\mu-1$. Then, we subdivide each region into subregions identified by an integer ξ by requiring that for all α in the subregion

$$2^{|\eta|+\xi/2^{\nu}} \leq |\,\alpha\,| \leq 2^{|\eta|+(\xi+1)/2^{\nu}}$$

for ξ an integer on $0 \leq \xi < 2^{\nu}$ and ν a resolution parameter. We fix $\nu = 11$ to obtain a high degree of accuracy in the tail of the histogram.

For each subregion, we compute the probability mass contained within the subregion by applying Simpson's rule, followed by Richardson extrapolation to cancel the linear error term. Simpson's rule is hence applied $2^{\nu}(1+2)$ times in each region. Each application requires $\Phi(\alpha)$ to be evaluated in up to three points (the two endpoints and the midpoint). When evaluating $\Phi(\alpha)$, we divide the result by 2^{κ} to account for the density of admissible pairs.

Note that as the distribution is symmetric around zero, we need only compute one side in practice. Note furthermore that the above approach to constructing the histogram is only valid provided κ is is not artificially large in relation to m, as we must otherwise account for which α are admissible. This is not a concern in cryptographic applications as d may then be presumed random.

5.2 Sampling the distribution

To sample an argument α from the histogram for the distribution, we first sample a subregion and then sample α uniformly at random from the set of all admissible arguments in this subregion. To sample the subregion, we first order all subregions in the histogram by probability, and compute the cumulative probability up to and including each subregion in the resulting ordered sequence. Then, we sample a pivot uniformly at random from $[0, 1) \subset \mathbb{R}$, and return the first subregion in the ordered sequence for which the cumulative probability is greater than or equal to the pivot. The sampling operation fails if the pivot is greater than the total cumulative probability.

To sample a pair (j, k) from the distribution, we first sample an admissible argument α as described above, and then sample (j, k) uniformly at random from the set of all integer pairs (j, k) yielding α using Lemma 3 below.

Lemma 3 *The set of integer pairs (j, k) on $0 \leq j < 2^{\ell+m}$ and $0 \leq k < 2^{\ell}$ that yield the admissible argument α is given by*

$$j = \left(\frac{\alpha - 2^m k}{2^\kappa} \left(\frac{d}{2^\kappa} \right)^{-1} + 2^{\ell+m-\kappa}\, t \right) \mod 2^{\ell+m}$$

as t runs through all integers on $0 \leq t < 2^\kappa$ and k through all integers on $0 \leq k < 2^\ell$.

Proof As $\alpha \equiv dj + 2^m k \mod 2^{\ell+m}$, it follows by solving for j as in Lemma 1 that

$$j = ((\alpha - 2^m k) d^{-1} + 2^{\ell+m-\kappa}\, t) \mod 2^{\ell+m},$$

as k and t run through $0 \leq k < 2^\ell$ and $0 \leq t < 2^\kappa$, enumerates the $2^{\ell+\kappa}$ distinct pairs (j, k) that yield the admissible argument α. By dividing both $\alpha - 2^m k$ and d by 2^κ, so as to show that the required inverse exists, the lemma follows. □

More specifically, we sample integers t, k uniformly at random from $0 \leq t < 2^\kappa$ and $0 \leq k < 2^\ell$, and then compute j from α and t, k as in Lemma 3.

6 Our improved post-processing algorithm

We are now ready to introduce our improved post-processing algorithm.

In analogy with the post-processing algorithm in [5], it recovers d from a set $\{(j_1, k_1), \ldots, (j_n, k_n)\}$ of n pairs produced by executing the quantum algorithm n times. However, where the original post-processing sets $n = s$ and requires subsets of s pairs from a larger set of about $8s$ pairs to be exhaustively solved for d, we allow n to assume larger values and simultaneously solve all n pairs for d.

Note that we may admit larger n because of the analysis in Sect. 4 which shows that even if a pair is not good by the definition in [5], then it is close to being good with overwhelming probability, in the sense that all pairs (j_i, k_i) have associated arguments $\alpha_i = \alpha(j_i, k_i) = \{dj_i + 2^m k_i\}_{2^{\ell+m}}$ of size $|\alpha_i| \sim 2^m$. This implies that virtually all pairs (j_i, k_i) yield $\sim \ell$ bits of information on d.

In analogy with the original post-processing algorithm in [5], the set of n pairs is used to form a vector $\mathbf{v} = (\{-2^m k_1\}_{2^{\ell+m}}, \ldots, \{-2^m k_n\}_{2^{\ell+m}}, 0) \in \mathbb{Z}^D$ and a D-dimensional integer lattice L with basis matrix

$$\begin{bmatrix} j_1 & j_2 & \cdots & j_n & 1 \\ 2^{\ell+m} & 0 & \cdots & 0 & 0 \\ 0 & 2^{\ell+m} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 2^{\ell+m} & 0 \end{bmatrix}$$

where $D = n + 1$. For some constants $m_1, \ldots, m_n \in \mathbb{Z}$, the vector

$$\mathbf{u} = (\{dj_1\}_{2^{\ell+m}} + m_1 2^{\ell+m}, \ \ldots, \ \{dj_n\}_{2^{\ell+m}} + m_n 2^{\ell+m}, \ d) \in L$$

is such that the distance

$$\begin{aligned} R = |\,\mathbf{u} - \mathbf{v}\,| &= \sqrt{\sum_{i=1}^{n} \left(\{dj_i\}_{2^{\ell+m}} + m_i 2^{\ell+m} - \{-2^m k_i\}_{2^{\ell+m}}\right)^2 + d^2} \\ &= \sqrt{\sum_{i=1}^{n} \underbrace{\{dj_i + 2^m k_i\}_{2^{\ell+m}}^2}_{\alpha_i^2} + d^2} = \sqrt{\sum_{i=1}^{n} \alpha_i^2 + d^2}\,. \end{aligned}$$

This implies that $\mathbf{u}$ and hence d may be found by enumerating all vectors in L within a D-dimensional hypersphere of radius R centered on $\mathbf{v}$. The volume of such a hypersphere is

$$V_D(R) = \frac{\pi^{D/2}}{\Gamma(\frac{D}{2} + 1)} R^D. \tag{8}$$

For comparison, the fundamental parallelepiped in L contains a single lattice vector and is of volume $\det L = 2^{(\ell+m)n}$. Heuristically, we therefore expect the hypersphere to contain approximately $v = V_D(R) \,/\, \det L$ lattice vectors. The exact number depends on placement of the hypersphere in $\mathbb{Z}^D$ and on the shape of the fundamental parallelepiped in L.

Assuming v to be sufficiently small for it to be computationally feasible to enumerate all lattice vectors in the hypersphere in practice, the above algorithm may be used to recover d. As the volume quotient v decreases in n, the number of vectors that need to be enumerated may be reduced by running the quantum algorithm more times and including the resulting pairs in L. However, there are limits to the number of pairs that may be included in L, as a reduced basis must be computed to enumerate L, and the complexity of computing such a basis grows fairly rapidly in the dimension of L.

In what follows, we show how to heuristically estimate the minimum number of runs n required to solve a specific known problem instance for d with minimum success probability q, for a given tradeoff factor s, and for a given bound on the number of vectors v that we at most accept to enumerate in L.

6.1 Estimating the minimum *n* required to solve for *d*

The radius R depends on the pairs (j_i, k_i) via the arguments α_i. For fixed n and fixed probability q, we may estimate the minimum radius $\widetilde{R}$ such that

$$\Pr\left[\ R = \sqrt{\sum_{i=1}^{n} \alpha_i^2 + d^2} \le \widetilde{R}\ \right] \ge q$$

by sampling α_i from the probability distribution as in Sect. 6.2. Then

$$\Pr\left[\ v = \frac{V_D(R)}{\det L} \le \frac{V_D(\widetilde{R})}{2^{(\ell+m)n}}\ \right] \ge q, \tag{9}$$

providing a heuristic bound on the number of lattice vectors v that at most have to be enumerated that holds with probability at least q.

Given an upper limit on the number of lattice vectors that we accept to enumerate, equation (9) may be used as an heuristic to estimate the minimum value of n such that v is below this limit with probability at least q.

To compute the estimate in practice, we use the heuristic to compute an upper bound on v for $n = 1, 2, \ldots$ and return the minimum n for which the bound is below the limit on the number of vectors that we accept to enumerate.

As the volume quotient v decreases by approximately a constant factor for every increment in n, the minimum n may be found efficiently via interpolation once the heuristic bound on v has been computed for a few values of n.

6.2 Estimating $\widetilde{R}$

To estimate $\widetilde{R}$ for m, s and n, explicitly known d, and a given target success probability q, we sample N sets of n arguments $\{\alpha_1, \ldots, \alpha_n\}$ from the probability distribution as described in Sect. 5.2. For each set, we compute R, sort the resulting list of values in ascending order, and select the value at index $\lceil (N-1)\, q \rfloor$ to arrive at our estimate for $\widetilde{R}$. Note that the constant N controls the accuracy of the estimate. For N sufficiently large in relation to q, and to the variance of the arguments, we expect to obtain sufficiently good estimates. Note furthermore that the sampling of arguments may fail. This occurs when the argument being sampled is not in the regions of the argument axis covered by the histogram.

If the sampling of at least one argument in a set fails, we err on the side of caution and over-estimate $\widetilde{R}$ by letting $R = \infty$ for the set. The entries for the failed sets will then all be sorted to the end of the list. If the value of $\widetilde{R}$ selected from the sorted list is ∞, no estimate is produced.

6.3 Results and analysis

To illustrate the efficiency of our new post-processing algorithm, we heuristically estimate n as a function of m and various s for the hard case $d = 2^m - 1$, and verify the estimates in practical simulations, in this section:

To this end, we let $\ell = \lceil m/s \rceil$ and fix $q = 0.99$. We fix $N = 10^6$ when estimating $\widetilde{R}$, and record the smallest n for which the volume quotient $v < 2$. Note that the latter requirement ensures that the lattice need not be enumerated. If $v < 2$ it heuristically suffices to reduce a single basis of dimension $D = n + 1$ and to then apply Babai's nearest plane algorithm [1] to recover $\mathbf{u}$ and hence d from $\mathbf{v}$. We verify the estimate by sampling $M = 10^3$ sets of n pairs $\{(j_1, k_1), \ldots, (j_n, k_n)\}$ and solving each set for d with our improved post-processing algorithm. If d is thus recovered, the verification succeeds, otherwise it fails. We record the smallest n such that at most $M(1-q) = 10$ verifications fail.

Table 1 was produced by executing the procedure described above. As may be seen in the table, the heuristically estimated values of n are in general verified by the simulations, except when v is close to two, in which case the values may differ slightly. Note that for large tradeoff factors s in relation to m, increasing or decreasing n typically has a small effect on v. This may lead to slight instabilities in the estimates, as v may be close to two for several values of n. Note furthermore that the difference in the sample sizes N and M may of course also give rise to slight discrepancies when verifying the estimates.

Compared to the post-processing algorithm originally proposed in [5], that required $8s$ runs to be performed and all subsets of s outputs from the resulting $8s$ outputs to be exhaustively solved for d, the new post-processing algorithm is considerably more efficient and achieves

Table 1 The estimated (A) and simulated (B) number of pairs n required for $\mathbf{u}$ to be the closest vector to $\mathbf{v}$ in L allowing d to be recovered via Babai's algorithm. If A = B, we only print A, otherwise we print B/A. Dash indicates no information

	Logarithm size m						
s	128	256	512	1024	2048	4096	8192
1	2	2	2	2	2	2	2
2	3	3	3	3	3	3	3
3	4	4	4	4	4	4	4
4	6	5	5	5	5	5	5
5	7/8	6	6	6	6	6	6
6	10	8	7	7	7	7	7
7	13	10/9	8	8	8	8	8
8	18	11	10	9	9	9	9
10	–/32	15	12	11	11	11	11
20	–	–/71	32/31	24	22	21	21
30	–	–	–/60	41/40	35	32	31
40	–	–	–	–/62	50/48	45/44	42
50	–	–	–	–	–/65	58/57	54/53
60	–	–	–	–	–	–/69	65
70	–	–	–	–	–	–	–/76

considerably better tradeoffs. It furthermore requires considerably fewer quantum algorithm runs. Asymptotically, the number of runs required n tends to $s + 1$ as m tends to infinity for fixed s, when we require a solution to be found without enumerating the lattice, as may be seen in Table 1. If we accept to enumerate a limited number of vectors in the lattice, this may potentially be slightly improved. In particular, a single run then suffices for $s = 1$.

To reduce the lattice bases, we used the LLL [9] and BKZ [13,14] algorithms, as implemented in fpLLL 5.2, with default parameters and for BKZ a block size of $\min(10, n + 1)$ for all combinations of m, s and n. For these parameter choices, a basis in general takes at most minutes to reduce and solve for d in a single thread, except when using BKZ for the largest combinations of m and s.

6.4 Further improvements

Above, we conservatively fixed a high minimum success probability $q = 0.99$ and considered the hard case $d = 2^m - 1$. Furthermore, we required that mapping $\mathbf{v}$ to the closest vector in L should yield $\mathbf{u}$ without enumerating L.

In practice, some of these choices may be relaxed: Instead of requiring $\mathbf{u}$ to be the closest vector to $\mathbf{v}$ in L, we may enumerate all vectors in a hypersphere of limited radius centered on $\mathbf{v}$. In cryptographic applications, the logarithm d may in general be assumed to be randomly selected. If not, the logarithm may be randomized; solve $x \odot [c]\, g$ for $d + c$ with respect to the basis g for some random c.

7 Summary and conclusion

We have introduced a new efficient post-processing algorithm that is practical for greater tradeoff factors, and requires considerably fewer quantum algorithm runs, than the original post-processing algorithm in [5]. To develop the new post-processing algorithm, and to estimate the number of runs it requires, we have analyzed the probability distribution induced by the quantum algorithm, and developed a method for simulating it when d is known.

With our new analysis and post-processing, Ekerå–Håstad's algorithm achieves an advantage over Shor's algorithms, not only in each individual run, but also overall, when targeting cryptographically relevant instances of RSA and Diffie–Hellman with short exponents. When making tradeoffs, Ekerå-Håstad's algorithm with our new post-processing furthermore outperforms Seifert's algorithm when factoring RSA integers. To the best of our knowledge, this makes Ekerå–Håstad's algorithm the currently most efficient polynomial time quantum algorithm for breaking the RSA cryptosystem [12], and DLP-based cryptosystems such as Diffie–Hellman [3] when instantiated with safe-prime groups with short exponents, as in TLS [7], IKE [8] and standards such as NIST SP 800-56A [2]. The reader is referred to Appendix A for a detailed analysis supporting these claims.

Acknowledgements Open access funding was provided by KTH, the Royal Institute of Technology. I am grateful to Johan Håstad for valuable comments and advice, to Lennart Brynielsson, and other colleagues, for their comments on early versions of this manuscript, and to the anonymous reviewers, for their constructive criticism. Funding and support for this work was provided by the Swedish NCSA that is a part of the Swedish Armed Forces. Computations were performed at PDC at KTH. Access was provided by SNIC.

Appendix A: Quantifying the advantage

In this appendix, we analyze the advantage of Ekerå–Håstad with our improved post-processing when targeting RSA or Diffie–Hellman in safe-prime groups with short exponents.

A.1 Diffie–Hellman

The portions of the NIST SP 800-56A recommendation [2] that pertain to Diffie–Hellman key exchange in $\mathbb{G} \subset \mathbb{F}_p^*$ were recently revised.

The previous version mandated the use of randomly selected Schnorr groups and supported moduli p of length up to 2048 bits. The new version supports moduli of length up to 8192 bits. For lengths exceeding 2048 bits, it mandates the use of a fixed set of safe-prime groups, originally developed for TLS and IKE [7,8].

Migrating from Schnorr groups to safe-prime groups with full length exponents would result in a significant performance penalty. NIST therefore permits the use of short exponents; the exponent length m must be on $2z \leq m \leq \lceil \log_2 r \rceil$, where z is the strength level of the group and $r = (p-1)/2$ is the group order.

Table 2 Estimated complexity of computing short discrete logarithms in the FIPS 800-56A safe-prime groups originally developed for TLS and IKE

$\lceil \log_2 p \rceil$	Strength level z	m	s	n	v	In each run		Overall	
						#Ops	Adv	#Ops	Adv
2048	112	224	1	1	$1.3 \cdot 10^3$	672	6.1	672	6.1
			7	10	$6.5 \cdot 10^{-4}$	288	14.2	2880	1.4
3072	128	256	1	1	$1.3 \cdot 10^3$	768	8.0	768	8.0
			8	11	$2.4 \cdot 10^{-1}$	320	19.2	3520	1.7
4096	152	304	1	1	$1.3 \cdot 10^3$	912	9.0	912	9.0
			9	12	$4.8 \cdot 10^{-1}$	372	22.0	4464	1.8
6144	176	352	1	1	$1.4 \cdot 10^3$	1056	11.6	1056	11.6
			10	13	$4.5 \cdot 10^{-2}$	424	29.0	5512	2.2
8192	200	400	1	1	$1.3 \cdot 10^3$	1200	13.7	1200	13.7
			11	14	7.5	474	34.6	6636	2.5

In Table 2 we provide estimates of the complexity of computing short discrete logarithms in these standardized safe-prime groups. The estimates were computed as in Sect. 6.3, for maximal $d = 2^m - 1$ and $\geq 99\%$ success probability, except that we accept to enumerate a limited number of vectors in L. We tabulate m, s and n, for $s = 1$, and for s the greatest tradeoff factor such that $n - s \leq 3$ with v close to two.

The number of group operations that need to be computed, in each run of the quantum algorithm, and overall in n runs, are tabulated for each tuple m, s and n, along with the advantage, defined as the quotient between $2 \lceil \log_2 r \rceil$, a low estimate of the number of group operations in each run of Shor's algorithm for general discrete logarithms, and the number of group operations, in each run or overall, in Ekerå–Håstad with our improved post-processing.

Note that the advantage is intentionally slightly underestimated as Ekerå–Håstad with our improved post-processing has $\geq 99\%$ probability of recovering d after n runs. More than one run of Shor's algorithm may be required to achieve a comparably high success probability, or a few more than $2 \lceil \log_2 r \rceil$ group operations may need to be computed in each run. The success probability depends on how the control registers are initialized, their lengths in relation to r, and on how the output from the quantum algorithm is post-processed.

As may be seen in Table 2, Ekerå–Håstad with our improved post-processing reduces the number of group operations in each run by up to a factor of 34.6 for these m, s and n. It achieves an advantage, not only in each run, but also overall, even for large tradeoff factors. We have verified the estimates by post-processing simulated outputs.

To aid interpretation, when an advantage is achieved in each run, it implies that Ekerå–Håstad's algorithm is easier to fit on a large-scale but constrained quantum computer compared to Shor's algorithm. Furthermore, it also implies that Ekerå–Håstad can be parallelized to achieve a reduction in the time required to solve a given problem instance. When an overall advantage is achieved, it implies that Ekerå–Håstad's algorithm requires less group operations to be computed overall in the n runs compared to a single run of Shor's algorithm.

Table 3 Estimated complexity of factoring RSA integers with $\ell = \lceil m/s \rceil$ and $s \geq 2$ in multiple runs, with $\geq 99\%$ success probability and maximal $d = 2^m - 1$

$\lceil \log_2 N \rceil$	m	s	n	v	in each run		overall	
					#ops	adv	#ops	adv
2048	1023	2	2	$1.4 \cdot 10^5$	2047	2.0	4094	1.0
		17	20	$3.3 \cdot 10^{-7}$	1145	3.6	22,900	0.18
3072	1535	2	2	$1.4 \cdot 10^5$	3071	2.0	6142	1.0
		21	24	$5.7 \cdot 10^{-9}$	1683	3.6	40,392	0.15
4096	2047	2	2	$1.4 \cdot 10^5$	4095	2.0	8190	1.0
		24	27	$1.0 \cdot 10^{-10}$	2219	3.7	59,913	0.14
6144	3071	2	2	$1.4 \cdot 10^5$	6143	2.0	12,286	1.0
		31	34	$8.2 \cdot 10^{-7}$	3271	3.8	111,214	0.11
8192	4095	2	2	$1.4 \cdot 10^5$	8191	2.0	16,382	1.0
		34	37	$5.1 \cdot 10^{-14}$	4337	3.8	160,469	0.10

A.2 RSA

Let p, q be two large distinct primes of length l bits. Then $N = pq$ is said to be an RSA integer. To factor N into p and q, we follow [5] and execute the below procedure:

Pick an element g, of unknown order r, uniformly at random from $\mathbb{Z}_N^*$. Let $\tilde{p} = (p-1)/2$ and $\tilde{q} = (q-1)/2$. Then r divides $2\tilde{p}\tilde{q} \,/\, \gcd(\tilde{p}, \tilde{q})$. Let $f(N) = (N-1)/2 - 2^{l-1} = 2\tilde{p}\tilde{q} + \tilde{p} + \tilde{q} - 2^{l-1}$. Compute $x = g^{f(N)} \equiv g^d$ for d on $0 \leq d < r$. Then $d = f(N) \bmod r = \tilde{p} + \tilde{q} - 2^{l-1}$, assuming $r > 2^{l-1}$, which holds with overwhelming probability. The discrete logarithm d is then short. To factor N, it suffices to compute d using the quantum algorithm of Ekerå and Håstad. Given d we may immediately solve $p + q = 2(d + 2^{l-1} + 1)$ and $pq = N$ for p and q.

Recall that in our analysis of the probability distribution, we required d to be on the interval $0 < d = \tilde{p} + \tilde{q} - 2^{l-1} < 2^m$, which implies $m = l - 1$. Furthermore, we required $r \geq 2^{\ell+m} + (2^\ell - 1)d$, where we let $\ell = \lceil m/s \rceil$ for s an integer. This implies that we must select $s \geq 2$ if the requirement on the size of r is to be respected with high probability.

In Table 3 we provide estimates of the complexity of factoring RSA integers. Again the estimates were computed as described in Sect. 6.3, for $\geq 99\%$ success probability and maximal $d = 2^m - 1$. We tabulate m, s and n, for $s = n = 2$, and for s the greatest tradeoff factor such that $n - s \leq 3$ whilst keeping v sufficiently small to avoid enumerating L.

The estimates are otherwise tabulated as described in Sect. A.1, except that we compute the advantage with respect to a single run of Shor's order finding algorithm. We have verified the estimates by post-processing simulated quantum algorithm outputs.

As may be seen in Table 3, an advantage is achieved in each run of the algorithm. For $s = 2$, we perform the same number of operations overall in the two runs, as Shor's algorithm does in a single run. Hence, the benefit of having a reduced circuit in each individual run comes at no cost in terms of the overall number of operations that need to be performed. For $s > 2$, we obtain a greater advantage in each individual run of the algorithm, at the expense of performing a greater number of operations overall than Shor's algorithm does in a single

Table 4 Estimated complexity of factoring RSA integers with $\ell = m - \delta$ for $\delta = 20$ in a single run, with $\geq 99\%$ success probability and maximal $d = 2^m - 1$

$\lceil \log_2 N \rceil$	m	ℓ	n	v	#ops	Adv
2048	1023	1003	1	$1.4 \cdot 10^9$	3029	1.35
3072	1535	1515	1	$1.4 \cdot 10^9$	4565	1.35
4096	2047	2027	1	$1.4 \cdot 10^9$	6101	1.34
6144	3071	3051	1	$1.4 \cdot 10^9$	9173	1.34
8192	4095	4075	1	$1.4 \cdot 10^9$	12,245	1.34

The advantage tends to $4/3$

run. Compared to the advantage Seifert's algorithm [15] achieves over Shor, Ekerå–Håstad with our improved post-processing achieves a further advantage by up to a factor of two.

A.2.1 Breaking RSA with an advantage in a single run

In the previous section, and in [5], we had to select $s \geq 2$ to respect the requirement that $r \geq 2^{\ell+m} + (2^\ell - 1)d$ where we let $\ell = \lceil m/s \rceil$. Assume that we instead let $\ell = m - \delta$, for some small positive integer δ. If δ is sufficiently large, then the requirement that $r \geq 2^{\ell+m} + (2^\ell - 1)d$ is respected with high probability. At the same time, if δ is not too large, we only need to enumerate a moderately large set of vectors in L in the classical post-processing.

Let us consider how δ affects the probability of respecting the requirement on r: The requirement that $r \geq 2^{\ell+m} + (2^\ell - 1)d$ may be simplified to $r \geq 2^{\ell+m+1} = 2^{2l-\delta-1}$, by using that $\ell = m - \delta$ where $m = l - 1$ and $d < 2^m$. Also $2^{2(l-1)} \leq \phi(N) = (p-1)(q-1) < 2^{2l}$, where ϕ is Euler's totient function, as p, q are large distinct l bit primes.

By Lemma 4 in the next section, that operates under slightly different assumptions that p, q are primes drawn from $[2, x]$ as $x \to \infty$ for proof-technical reasons, the probability that $r \geq \phi(N)/2^\tau$ is lower-bounded by $P_N(\tau)$. It is reasonable to assume that this result carries over to our situation where p, q are of large fixed length. This assumption is supported by simulations where $N = pq$ and $g \in \mathbb{Z}_N^*$ are sampled and r estimated, see Sect. A.2.3. Under this assumption, we have that $r \geq 2^{2(l-1)-\tau}$ with probability at least $P_N(\tau)$.

Equating this to the requirement that $r \geq 2^{2l-\delta-1}$, we have that $2(l-1) - \tau = 2l - \delta - 1$, which in turn implies $\tau = \delta - 1$. Let us pick $\delta = 20$. We then have that $P_N(\tau = \delta - 1) \geq 0.9998$. This is a very high probability of respecting the requirement on r.

At the same time, for $\delta = 20$, we expect to have to enumerate at most $v = 1.4 \cdot 10^9$ vectors in L in the classical post-processing to recover d with success probability $\geq 99\%$, see Table 4 where we have again estimated v as in Sect. 6.3 for maximal $d = 2^m - 1$.

Such an enumeration is easy to perform in practice. It is facilitated by the lattice being two-dimensional, and by the last component of the vector sought being d. The facts that $0 < d \leq 2^m$ and $d \equiv (N-1)/2 \pmod 2$ may hence be used to prune the enumeration. We have verified that this slightly tweaked version of Ekerå–Håstad's algorithm achieves $\geq 99\%$ success probability by post-processing simulated outputs.

Note that we tweaked the algorithm to ensure that $r \geq 2^{\ell+m} + (2^\ell - 1)d$. This requirement serves to prevent modular reductions from occurring in our analysis of the probability distribution. It may be possible to relax this requirement, at the expense of carrying out a more complicated analysis. We circumvent this complication by letting $\ell = m - \delta$.

A.2.2 On the probability of picking a generator with sufficiently large order

In this section we analyze the probability of picking $g \in \mathbb{Z}_N^*$ with sufficiently large order.

Lemma 4 *Let $N = pq$ for p, q distinct primes drawn uniformly at random from $[2, x]$. Pick g uniformly at random from $\mathbb{Z}_N^*$. Let r be the order of g. For fixed $\tau > 1$, and x sufficiently large as a function of τ, the probability that $r \geq \phi(N)/2^\tau$ is lower-bounded by $P_N(\tau)$ where*

τ	10	15	16	17	18	19	20
$P_N(\tau)$	0.9756	0.9985	0.9992	0.9995	0.9997	0.9998	0.9999

The bound is asymptotically correct in the limit as x tends to infinity.

Proof As $\mathbb{Z}_N^*$ is isomorphic to $\mathbb{Z}_{p-1}^+ \times \mathbb{Z}_{q-1}^+$ the order $r \leq \phi(N)/\gcd(p-1, q-1)$ where $\phi(N) = (p-1)(q-1)$. The order r is hence always *reduced* by a factor $\gcd(p-1, q-1)$ compared to $\phi(N)$ due to shared factors. It may be further reduced: This depends on g, and on how $p-1$ and $q-1$ factor. To understand the distribution of r, we therefore first seek an expression for the probability of a factor z dividing $p-1$, and $q-1$, respectively.

To this end, we use Siegel–Walfisz' theorem [18,19]. In essence it states that as $x \to \infty$ the probability over all primes $p \leq x$ that $p \equiv a \pmod{z}$ is $1/\phi(z)$, for $z \in [2, \ln^c(x)]$ a fixed integer, where c is an arbitrary positive constant, and a a fixed residue coprime to z. For $a = 1$ this implies that $p - 1 \equiv 0 \pmod{z}$, i.e. z divides $p-1$, with probability $1/\phi(z)$. The same holds for $q-1$. Furthermore, as $\phi(z_1 z_2) = \phi(z_1) \cdot \phi(z_2)$ for z_1, z_2 coprime factors, we may independently analyze the reductions caused by distinct prime powers.

To obtain the lower bound $P_N(\tau)$ on the probability of $r \geq \phi(N)/2^\tau$, we let $\mathcal{F}$ be the set of all primes less than 2^τ, and $\mathcal{F}'$ the set of all primes greater than 2^τ, and proceed as follows:

In Part 1 of the proof, we compute all reductions $\phi(N)/r \leq 2^\tau$ to which combinations of powers of factors in $\mathcal{F}$ give rise, and sum up the associated probabilities, to obtain the probability $\rho_N(\tau)$. In Part 2, we upper-bound the probability of factors in $\mathcal{F}'$ giving rise to a reduction in the order greater than 2^τ, and subtract it from $\rho_N(\tau)$ to obtain $P_N(\tau)$.

Part 1 For each $f \in \mathcal{F}$, let n be the greatest power of f such that $f^n \leq 2^\tau$. As f^e divides $p-1$, or equivalently $q-1$, with probability $1/\phi(f^e)$, we have that

$$\Lambda(e, f) = \begin{cases} 1 - 1/(f-1) & e = 0 \\ (1 - 1/f)/((f-1)\, f^{e-1}) & e \in [1, n] \\ 1/((f-1)\, f^n) & e = n+1 \end{cases}$$

is the probability that f^e is the greatest power of f to divide $p-1$, or equivalently $q-1$, for $e \in [0, n]$, and the probability that f^{n+1} divides $p-1$, or equivalently $q-1$, when $e = n+1$.

For pairwise combinations of $e_p, e_q \in [0, n+1]$, the probability is $\Lambda(e_p, f) \cdot \Lambda(e_q, f)$ that f^{e_p} and f^{e_q} divide $p-1$ and $q-1$, respectively, in the special manner described above.

In the formulation of this lemma, we pick g uniformly at random from $\mathbb{Z}_N^*$, and consider the order of g. This is equivalent to selecting g_p and g_q uniformly at random from the cyclic groups $\mathbb{Z}_{p-1}^+$ and $\mathbb{Z}_{q-1}^+$, respectively, and considering the order of $(g_p, g_q) \in \mathbb{Z}_{p-1}^+ \times \mathbb{Z}_{q-1}^+ \simeq \mathbb{Z}_N^*$.

The probability of selecting g_p such that f^{u_p} is the greatest power of f to divide g_p for $u_p \in [0, e_p)$, leading to a reduction in the order of g_p by a factor f^{u_p}, or such that f^{u_p} divides g_p for $u_p = e_p$, leading to a reduction in the order of g_p by a factor f^{u_p} when $e_p \leq n$, or by at least a factor f^{u_p} when $e_p = n + 1$, is $\Pi(u_p, e_p, f)$, where

$$\Pi(u, e, f) = \begin{cases} (1 - 1/f)/f^u & u \in [0, e) \quad \text{and } e > 0, \\ 1/f^e & u = e. \end{cases}$$

Analogously, the probability of selecting g_q that is divisible by f^{u_q} in the above special manner for $u_q \in [0, e_q]$, leading to corresponding reductions in the order of g_q, is $\Pi(u_q, e_q, f)$.

For the g_p and g_q selected, the order of (g_p, g_q) is hence reduced by a factor $f^{u_p+u_q+u_{pq}}$ for $e_p, e_q \in [0, n]$ due to $f \in \mathcal{F}$, where $u_{pq} = \min(e_p - u_p, e_q - u_q)$. The additional reduction by $f^{u_{pq}}$ is due to shared factors in the orders of g_p and g_q. If $e_p = n + 1$ and/or $e_q = n + 1$, the reduction is at least $f^{u_p+u_q+u_{pq}}$; it may be greater, but only if $f^{u_p+u_q+u_{pq}} > 2^\tau$, and all such cases may be treated jointly. Hence, we may model the order $\phi(N)$ as being reduced by a factor $f^{u_p+u_q+u_{pq}}$ due to $f \in \mathcal{F}$ with probability $\Lambda(e_p, f) \cdot \Pi(u_p, e_p, f) \cdot \Lambda(e_q, f) \cdot \Pi(u_q, e_q, f)$.

Given this analysis, the probability $\rho_N(\tau)$ that a factor at most 2^τ is lost due to prime factors in $\mathcal{F}$ can be evaluated numerically by computer: Exhaust all products of reductions that combinations of factors in $\mathcal{F}$ can give rise to, with the restriction that the product must not exceed 2^τ, and sum up the probabilities associated with each reduction, to obtain $\rho_N(\tau)$.

Part 2 For factors $f \in \mathcal{F}'$, we upper-bound the probability of the order $\phi(N)$ being reduced by at least a factor $f > 2^\tau$ as follows: Let $P(f)$ denote the probability that a factor f divides $p - 1$, and analogously $q - 1$. If f divides both $p - 1$ and $q - 1$, the order is reduced by at least a factor $f > 2^\tau$. The total probability of this event is $P(f)^2$. If f divides $p - 1$ but not $q - 1$, or vice versa, the order is reduced by a factor $f > 2^\tau$ with probability $1/f$. The total probability of this event is $P(f)(1 - P(f))(1/f) \leq P(f)/f$.

To summarize, the probability of reducing the order $\phi(N)$ by more than 2^τ via $f \in \mathcal{F}'$, when $f \leq \ln^c(x)$ so that $P(f) = 1/\phi(f)$ by Siegel–Walfisz' theorem, is upper-bounded by

$$\sum_{f \in \mathcal{F}'} \left(\frac{1}{\phi(f)^2} + \frac{2}{f\phi(f)} \right) \leq \sum_{f \in \mathcal{F}'} \frac{3}{(f-1)^2} \leq \sum_{z=2^\tau+1}^{\infty} \frac{3}{(z-1)^2} \leq \int_{2^\tau}^{\infty} \frac{3}{(z-1)^2} \, \mathrm{d}z = \frac{3}{2^\tau - 1}.$$

To handle the contribution from factors $f > \ln^c(x)$, we may instead use that $P(f) \leq \ln(x)/f$ by Claim 2, to obtain the upper bound

$$\sum_f \left(\frac{\ln^2(x)}{f^2} + \frac{2\ln(x)}{f^2} \right) \leq \int_{\ln^c(x)-1}^{\infty} \frac{3\ln^2(x)}{z^2} \, \mathrm{d}z = \frac{3\ln^2(x)}{\ln^c(x) - 1}$$

which, for any $c > 2$, may be seen to tend to zero in the limit as $x \to \infty$.

The table in the lemma results from the execution of the procedure described above, with $P_N(\tau) = \rho_N(\tau) - 3/(2^\tau - 1)$ and for a selection of values of τ, and so the lemma follows. □

Claim 2 *For p a prime drawn uniformly at random from $[2, x]$, and $z \geq 2$ an integer, the probability that z divides $p - 1$ is upper-bounded by $\ln(x)/z$ in the limit as $x \to \infty$.*

Proof The number of primes on $[2, x]$ is $x/\ln(x)$ in the limit as $x \to \infty$ by the prime number theorem, so the above claim only states that there are at most $(x/\ln(x)) \cdot (\ln(x)/z) = x/z$ primes p on $[2, x]$ such that z divides $p - 1$. This is true, as x/z is the number of integers up to x that are divisible by z in the limit as $x \to \infty$, and so the claim follows. □

In Lemma 4 above, we consider $N = pq$ with p, q distinct primes drawn uniformly at random from $[2, x]$ as $x \to \infty$. In practice, we would instead like to select $p, q \neq p$ uniformly at random from the set of all primes of length l bits, so that N is an RSA integer by the definition in this paper. In practice, we may also wish to require that N must be a $2l$ bit number. Imposing such restrictions on the sizes of p and q should not have any significant impact on the outcome of the analysis, for the reasons elaborated on in the next section.

If other special requirements are imposed on p and q, the analysis may however need to be adapted to meet these requirements. In particular, this is the case if it is required that $p-1$ and $q-1$ have no small divisors besides two. The success probability then increases.

A.2.3: Supplementary supporting simulations

The primary obstacle to selecting p, q as l bit primes in Lemma 4 is the unknown constant in the error term in Siegel–Walfisz' theorem. Asymptotically, the error term goes to zero regardless of the value of the constant, allowing us to state that z divides $p-1$ with probability $1/\phi(z)$ in Lemma 4 for p selected uniformly at random on $[2, x]$ as $x \to \infty$.

For large fixed x, taking the probability to be $1/\phi(z)$ is a reasonable approximation. However, without a good explicit bound on the error in the approximation, technical problems arise if one seeks to follow the approach taken in Lemma 4. In particular, problems arise when exhausting all combinations of products of small prime factor powers in Part 1 of the proof.

One way to obtain an estimate anyway is to instead run numerical simulations: Draw two distinct primes p, q uniformly at random from $(2^{l-1}, 2^l]$. Compute $N = pq$ and select a generator g uniformly at random from $\mathbb{Z}_N^*$. Let $r = \phi(N)/\gcd(p-1, q-1)$. For all prime factors $f \leq 2^\eta$ for some sufficiently large η, test if f divides r. If so, and if $g^{r/f} \equiv 1$, reduce r by a factor of f. Go back and test again recursively if f still divides r. Otherwise, process the next factor. The order r thus computed is a good heuristic estimate of the order of g.

We have run such experiments, in which we repeated the simulation procedure 10^6 times, for $l = 1024$, $\tau = 19$ and $\eta = 24$. On average, only ~ 130 problem instances failed to meet the requirement that $r \geq \phi(N)/2^\tau$. This is in agreement with the asymptotic bound. It furthermore indicates that the success probability is high already for 2048 bit RSA integers.

References

1. Babai L.: On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica **6**(1), 1–13 (1986).
2. Barker E., et al.: NIST SP 800-56A: Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography, 3rd revision (2018).
3. Diffie W., Hellman M.E.: New Directions in Cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976).
4. Ekerå M.: Modifying Shor's algorithm to compute short discrete logarithms. IACR ePrint Archive, Report 2016/1128 (2016).
5. Ekerå M., Håstad J.: Quantum algorithms for computing short discrete logarithms and factoring RSA integers. In: PQCrypto. Lecture Notes in Computer Science, vol. 10346, pp. 347–363. Springer, Cham (2017).
6. Fowler A.G., Mariantoni M., Martinis J.M., Cleland A.N.: Surface codes: towards practical large-scale quantum computation. Phys. Rev. A **86**(3), 032324 1–48 (2012).
7. Gillmor D.: RFC 7919: Negotiated Finite Field Diffie–Hellman Ephemeral Parameters for Transport Layer Security (TLS) (2016).
8. Kivinen T., Kojo M.: RFC 3526: More Modular Exponentiation (MODP) Diffie–Hellman groups for Internet Key Exchange (2003).

9. Lenstra A.K., Lenstra H.W., Lovász L.: Factoring polynomials with rational coefficients. Math. Ann. **261**, 515–534 (1982).
10. Mosca M., Ekert A.: The Hidden Subgroup Problem and Eigenvalue Estimation on a Quantum Computer. In: Proceeding from the First NASA International Conference, Quantum Computing and Quantum Communications, vol. 1509, pp. 174–188 (1999).
11. Parker S., Plenio M.B.: Efficient factorization with a single pure qubit and log N mixed qubits. Phys. Rev. Lett. **85**(14), 3049–3052 (2000).
12. Rivest R.L., Shamir A., Adleman L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978).
13. Schnorr C.-P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theor. Comput. Sci. **53**(2–3), 201–224 (1987).
14. Schnorr C.-P., Euchner M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. Math. Program. **66**(1–3), 181–199 (1994).
15. Seifert J.-P.: Using fewer qubits in Shor's factorization algorithm via simultaneous diophantine approximation. In: CT-RSA. Lecture Notes in Computer Science, vol. 2020, pp. 319–327. Springer, Berlin (2001).
16. Shor P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: SFCS. Proceedings of the 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE Computer Society, Washington DC (1994).
17. Shor P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997).
18. Siegel C.L.: Über die Classenzahl quadratischer Zahlkörper. Acta Arith. **1**(1), 83–86 (1935).
19. Walfisz A.: Zur additiven Zahlentheorie. II. Math. Z. **40**(1), 592–607 (1936).