# A fusion algorithm for solving the hidden shift problem in finite abelian groups


Wouter Castryck[1,3], Ann Dooms[2], Carlo Emerencia[2], Alexander Lemmens[1]

[1] imec-COSIC, Department of Electrical Engineering, KU Leuven, Belgium
[2] DIMA, Department of Mathematics and Data Science, VUB, Belgium
[3] Department of Mathematics: Algebra and Geometry, Ghent University, Belgium



**Abstract.** It follows from a result by Friedl, Ivanyos, Magniez, Santha and Sen from 2014 that, for any fixed integer $m > 0$ (thought of as being small), there exists a quantum algorithm for solving the hidden shift problem in an arbitrary finite abelian group $(G, +)$ with time complexity

$$\mathrm{poly}(\log |G|) \cdot 2^{O(\sqrt{\log |mG|})}.$$

As discussed in the current paper, this can be viewed as a modest statement of Pohlig–Hellman type for hard homogeneous spaces. Our main contribution is a simpler algorithm achieving the same runtime for $m = 2^t p$, with $t$ any non-negative integer and $p$ any prime number, where additionally the memory requirements are mostly in terms of quantum random access classical memory; indeed, the amount of qubits that need to be stored is $\mathrm{poly}(\log |G|)$. Our central tool is an extension of Peikert's adaptation of Kuperberg's collimation sieve to arbitrary finite abelian groups. This allows for a reduction, in said time, to the hidden shift problem in the quotient $G/2^t pG$, which can then be tackled in polynomial time, by combining methods by Friedl et al. for $p$-torsion groups and by Bonnetain and Naya-Plasencia for $2^t$-torsion groups.



*Keywords:* hidden shift, collimation sieve, hard homogeneous space


## 1 Introduction

In 1994 Simon's algorithm [24] laid the basis for Shor's celebrated polynomial-time quantum algorithm [23] for factoring integers and computing discrete logarithms. It is now a standard fact that Shor's algorithm is essentially equivalent to solving the *hidden subgroup problem* in finite abelian groups, which led to investigating this problem for non-abelian groups as well.

One of the most famous results in that direction is that of Kuperberg [16] from 2003, who showed that for any finite abelian group $G$, the hidden subgroup problem in the associated dihedral group $\mathrm{Dih}(G)$ can be tackled in quantum sub-exponential time, namely using

$$2^{O(\sqrt{\log |G|})} \tag{1}$$

operations, i.e., queries and gates. In fact, a more natural description of Kuperberg's algorithm is that it solves the *hidden shift problem* in $G$ (also called the *hidden translation problem*), which can be seen to be equivalent[1] to the hidden subgroup problem in $\mathrm{Dih}(G)$. Formally, this problem is defined as follows.

**Definition 1.1 (Abelian hidden shift problem).** *Given a finite abelian group $(G,+)$, a set $X$, and oracle access to injective functions $f_1, f_2 : G \to X$ for which there exists an $s \in G$ such that $f_1(g) = f_2(g+s)$ for all $g \in G$, find $s$.*

The literature contains versions with non-injective $f_1, f_2$, see e.g. [12], but these will not be considered here.

Kuperberg's result increased the interest in the hidden shift problem as a stand-alone problem. It gained relevance for public-key cryptography when Childs, Jao and Soukharev [8] tied it to the vectorization problem in hard homogeneous spaces such as CRS [10,22] and CSIDH [4], which is a candidate post-quantum replacement for the discrete logarithm problem. Note that the hidden shift problem also naturally appears as the security primitive of certain symmetric cryptosystems, see e.g. [1].

Unfortunately, Kuperberg's algorithm requires storage of a similar amount of qubits as (1). This was mitigated by Regev [21], who showed how to obtain polynomial space complexity at the expense of a small increase of the runtime; while Regev restricted his attention to cyclic 2-groups, the details for arbitary finite abelian groups were elaborated by Childs et al. [8, App. A]. In a 2011 follow-up paper [17], Kuperberg described the *collimation sieve*, which reachieves a time and space complexity of (1), but with an important bonus: the space requirements are in terms of quantum random access *classical* memory (QRACM), indeed the amount of qubits that need to be stored remains polynomial. Kuperberg provided details for cyclic 2-groups only; an adaptation that works for all finite cyclic groups was described by Peikert [19].

There exist, however, special families of groups in which the hidden shift problem becomes much easier. The basic example is where $G \cong \mathbb{Z}_2^r$ for some $r \geq 1$. Indeed, for such groups the associated dihedral group is abelian, allowing Shor's algorithm, which boils down to Simon's method in this case, to recover $s$ in polynomial time. This example can be generalized. For instance, a version of Simon that works for $G \cong \mathbb{Z}_p^r$ for any fixed prime $p$ can be found in a paper by Friedl, Ivanyos, Magniez, Santha and Sen [14]. In a different direction, this was generalized to $G \cong \mathbb{Z}_{2^t}^r$ for any fixed integer $t \geq 1$ by Bonnetain and Naya-Plasencia [2], who did so by incorporating ingredients from Kuperberg's first algorithm.

**Contributions.** The contributions of this paper are threefold:

*(i)* We extend the method of Bonnetain–Naya-Plasencia from groups of the form $\mathbb{Z}_{2^t}^r$ to arbitrary $2^t$-torsion groups, i.e., to groups of the form $\bigoplus_{i=1}^r \mathbb{Z}_{2^{t_i}}$

[1] To see one implication explicitly, we can write $\mathrm{Dih}(G) = \{(a,b) \mid a \in G,\, b \in \{\pm 1\}\}$ with composition law $(a_1,b_1)\cdot(a_2,b_2) = (a_1 + b_1 a_2, b_1 b_2)$. Then the function mapping $(a,1)$ to $f_1(a)$ and $(a,-1)$ to $f_2(a)$ is constant precisely on left cosets of the subgroup $\{(0,1),(s,-1)\}$, and recovering this subgroup clearly recovers the hidden shift $s$. For the other implication, see [13].

with $t_i \leq t$ for all $i$. Moreover, we show how to combine this with the method of Friedl et al., in order to obtain a polynomial-time quantum algorithm for tackling the hidden shift problem in finite abelian $2^t p$-torsion groups (for fixed $p, t$).

*(ii)* We further extend Peikert's adaptation of the collimation sieve, so that it works for any finite abelian group, rather than just cyclic groups.

*(iii)* We merge the algorithms from *(i)* and *(ii)*. When solving the hidden shift problem in an arbitary finite abelian group $G$, one can use collimation to produce phase vectors that only involve $2^t p$-torsion characters. At that point, one can switch to the polynomial-time algorithm from *(i)* for finding $s \bmod 2^t pG$. Then a run of algorithm *(ii)* on $2^t pG$ allows one to conclude.

Altogether, this leads to the following theorem:

**Theorem 1.2.** *For any fixed prime number $p$ and non-negative integer $t$, there exists a quantum algorithm for solving the hidden shift problem in any finite abelian group $(G, +)$, with time, query and QRACM complexity*

$$\mathrm{poly}(\log |G|) \cdot 2^{O(\sqrt{\log |2^t pG|})}$$

*and requiring storage of* $\mathrm{poly}(\log |G|)$ *qubits.*

We note that the runtime part of this statement is not new, although this is somewhat hidden in the quantum physics literature and does not seem well-known among cryptographers. Indeed, an application of [14, Prop. 4.13] to the chain $G \geq 2G \geq \cdots \geq 2^t G \geq 2^t pG \geq \{0\}$, in combination with [14, Prop. 2.2], leads to an algorithm with the same time complexity. In fact, by modifying the chain, one obtains the same result for any positive integer $m$, rather than just $m$'s of the form $2^t p$. However, our method is somewhat simpler and keeps the key advantage of the collimation sieve, namely that the main memory requirements are in terms of QRACM; only a polynomial number of qubits is needed.

As a consequence, if our finite abelian group $G$ has a large $2^t p$-torsion subgroup for certain small $p$ and $t$, or more generally a large $m$-torsion subgroup for a certain small $m$, then one can essentially discard this subgroup when assessing the hardness of the hidden shift problem. As discussed more extensively in Section 2, this observation can be viewed as a modest result of Pohlig–Hellman [20] type for the vectorization problem in arbitrary hard homogeneous spaces. In the specific case of CRS and CSIDH, it was already shown in [5, §5.1] how to get rid of part of the 2-torsion, using very different (classical) methods.

**Paper organization.** The implications of Theorem 1.2 and alike for the vectorization problem are discussed more elaborately in Section 2. In Section 3 we describe our polynomial-time quantum algorithm for solving the hidden shift problem in $2^t p$-torsion groups. Section 4 presents a version of the collimation sieve that works for arbitrary abelian groups. Then in Section 5 we show how to incorporate the first algorithm in this extended collimation sieve, leading to Theorem 1.2. Finally, in Section 6 we give some concluding remarks.

**Prerequisites.** All algorithms below rely on the *standard approach* for hidden shift finding which, at the cost of calls to $f_1, f_2$ and a quantum Fourier transform, produces length-two phase vectors, i.e. qubits of the form

$$\Psi(\chi) = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}\chi(s)|1\rangle, \tag{2}$$

with $\chi$ a *known* uniformly random element of $G^\vee = \{\,\text{homomorphisms } G \to \mathbb{C}^*\,\}$; see e.g. [21, §2.2] for more details. We will treat the standard approach as an oracle in its own right. Beyond this, our discussion is more or less stand-alone, but concise, so some familiarity with prior hidden-shift algorithms is convenient. We refer the reader to [3,7,19] for recent accounts.

**Acknowledgments.** This work was supported by the Research Council KU Leuven grant C14/18/067, by CyberSecurity Research Flanders with reference number VR20192203, and by the Research Foundation Flanders (FWO) through the WOG Coding Theory and Cryptography.

## 2 Consequences for the vectorization problem

The *vectorization problem* is a generalization of the discrete logarithm problem that was formally introduced by Couveignes [10]. Here, we assume that our finite abelian group $G$ acts freely and transitively on the set $X$, which means that we are given a map

$$G \times X \to X : (s, x) \mapsto s \star x$$

such that $0 \star x = x$, $(s_1 + s_2) \star x = s_1 \star (s_2 \star x)$ and $G \to X : s \mapsto s \star x$ is a bijection for all $s_1, s_2 \in G, x \in X$. Then the vectorization problem is about extracting $s$ from any given pair $x, s \star x$. By considering the maps

$$f_1 : G \to X : a \mapsto a \star (s \star x), \qquad f_2 : G \to X : a \mapsto a \star x$$

one sees that this indeed concerns an instance of the hidden shift problem. If the vectorization problem is hard, then $X$ is called a *hard homogeneous $G$-space*. In view of Kuperberg's algorithm and its successors, we know that quantum computers can solve the vectorization problem in time (1). Therefore, the best one can hope is that quantum adversaries cannot do better than this.

An efficient solution to the vectorization problem clearly breaks the Diffie–Hellman style key exchange protocol depicted in Figure 1. Note, however, that key recovery amounts to finding $(s_1 + s_2) \star x$ when given a triple $x, s_1 \star x, s_2 \star x$. This is called the *parallellization problem* which, a priori, could be easier than vectorization, but in the presence of quantum computers, both problems are in fact equivalent [15].

*Example 2.1.* Textbook Diffie–Hellman, based on exponentiation in a finite cyclic group $H$, arises by letting

$$G = \mathbb{Z}^*_{|H|}, \qquad X = \{\,\text{generators of } H\,\}.$$

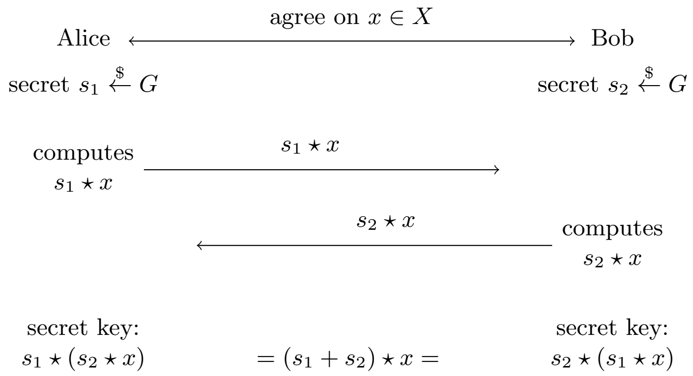


**Fig. 1.** Diffie–Hellman key exchange in a hard homogeneous $G$-space $X$

In this context, the vectorization problem, resp. the parallellization problem, becomes the discrete logarithm problem, resp. the computational Diffie–Hellman problem. Quantum computers break these problems using Shor's algorithm, therefore we focus on classical adversaries, to which both problems are still believed to be equivalent [18]. Then by the well-known Pohlig–Hellman reduction [20], the discrete logarithm problem in $H$ is essentially as hard as that in its largest prime order subgroup. In particular, the discrete logarithm problem is weakened by the presence of many small prime factors in $|H|$. However, assuming that $|H|$ is a large prime, we are unaware of classical exploits of the structure of the acting group $G = \mathbb{Z}^*_{|H|}$.

*Example 2.2.* The only known instantiation believed to be secure against quantum adversaries is due to Couveignes [10] and Rostovtsev–Stolbunov [22,25], the fastest version of their construction being CSIDH [4]. Here $G = \mathrm{Cl}(\mathcal{O})$ is the ideal-class group of an order $\mathcal{O}$ in an imaginary quadratic number field and

$$X = \mathcal{E}\ell\ell_{\mathbb{F}_q}(\mathcal{O}, t) :=$$
$$\{\,\text{elliptic curves } E/\mathbb{F}_q \text{ with trace of Frobenius } t \text{ and } \mathrm{End}_{\mathbb{F}_q}(E) \cong \mathcal{O}\,\} \,/ \cong_{\mathbb{F}_q},$$

with $\mathbb{F}_q$ a finite field and $t$ an integer such that $X$ is non-empty. The action is isogeny-wise; see the cited references for details. Using the Tate pairing on elliptic curves it is possible, in *classical* polynomial time, to recover the hidden shift $s$ modulo a certain subgroup $H$ satisfying $2G \leq H \leq G$; concretely $H$ arises is the joint kernel of the quadratic characters of $G$ that have a polynomially small modulus [5, §5.1]. This can be used to reduce the hidden shift problem in $G$ to that in $H$, which can be viewed as a modest statement of Pohlig–Hellman type, in the sense that the vectorization problem is typically weakened by the presence of a large 2-torsion subgroup. Note the following difference with the standard Pohlig–Hellman reduction: here we are concerned with an exploit of the structure of the *acting* group, rather than of a group that is acted upon (and, of course, the full Pohlig–Hellman reduction is a much stronger exploit).

Up to our knowledge, so far, it was left unnoticed that this Pohlig–Hellman type statement allows for a vast generalization in the presence of quantum computers: indeed, from Friedl et al. [14, Prop. 2.2, Prop. 4.13] we learn that the vectorization problem in any hard homogeneous space can be solved in time

$$\mathrm{poly}(\log |G|) \cdot 2^{O(\sqrt{\log |mG|})},$$

for any fixed $m \in \mathbb{Z}_{>0}$, which one should think of as being small.[2] Thus, quantumly, the vectorization problem is weakened by the presence of a large $m$-torsion subgroup. We stress that, in contrast with [5], neither the algorithm by Friedl et al. nor our qubit-friendly special case from Theorem 1.2 recovers the hidden shift $s$ modulo $mG$ in polynomial time, which would be needed to break decisional versions of the parallellization problem.

*Example 2.3 (continuation of Example 2.2).* For $m = 2$ we do not learn much new. Note that in the specific case of CSIDH one always has $H = 2G$; for CRS it may happen that $H > 2G$, and then the remaining gap can be bridged by our algorithm from Theorem 1.2. On the other hand, beyond $m = 2$, class groups of imaginary quadratic orders seem well-protected against these types of reductions. Indeed, Gerth's extension [11] of the Cohen–Lenstra heuristic [9, §9] predicts that $2G$ has a strong tendency to be cyclic, i.e., it is likely that for all primes $p$ the $p$-torsion part of $2G$ has rank 0 or 1. More precisely, the "probability" that the rank equals $r$ is

$$\approx \; p^{-r^2} \; \cdot \; \prod_{k=1}^{r} (1 - p^{-k})^{-2} \; \cdot \; \prod_{k=1}^{\infty} (1 - p^{-k}),$$

which decays very quickly with $r$. E.g., the probability that $2G$ contains a 3-torsion subgroup of rank at least 10 is about $5.2 \cdot 10^{-31}$. Nevertheless, some extra care may be desirable when setting up CRS or CSIDH using a class group whose structure is unknown.

## 3 Polynomial-time hidden shifts in $2^t p$-torsion groups

Fix a prime number $p$ and a positive integer $t$. In this section, we describe a polynomial-time quantum algorithm for solving the hidden shift problem in a finite abelian group $(G, +)$ where every element $g$ satisfies $2^t pg = 0$. This part is heavily inspired by Bonnetain–Naya-Plasencia [2], Csáji [6] and Friedl et al. [14]. We remark that [14] contains results for more general abelian groups of low

[2] The choice of $m$ strongly affects the hidden constant in the $O$-notation which, according to [14, Prop. 4.13], is $\Omega(rp)$ with $p$ the largest prime factor of $m$ and $r$ the number of prime factors.

exponent, but these do not use the standard approach to hidden shift finding, which seems to make them incompatible with the collimation sieve. Without loss of generality we can assume that $p > 2$ and that $G = \mathbb{Z}_p^m \times \mathbb{Z}_{2^{t_1}} \times \cdots \times \mathbb{Z}_{2^{t_n}}$ for certain integers $m, n \geq 0$ and $t = t_1 \geq t_2 \geq \ldots \geq t_n \geq 1$.

**Step 3.1.** We describe a routine for producing length-two phase vectors $\Psi(\chi)$ such that $\chi^p = 1$. Following Kuperberg [16, §3], two phase vectors $\Psi(\chi_1)$ and $\Psi(\chi_2)$ can be combined to yield $\Psi(\chi_1\chi_2)$ or $\Psi(\chi_1\chi_2^{-1})$, each with probability 1/2. More generally, one can merge any $k \geq 2$ phase vectors $\Psi(\chi_1), \ldots, \Psi(\chi_k)$ to obtain a phase vector of the form

$$\Psi(\chi_1^{\pm 1} \cdots \chi_k^{\pm 1}). \tag{3}$$

We call a phase vector $\Psi(\chi)$ *$\ell$-divisible* if $\chi^{2^{t-\ell}p} = 1$. Our routine merges fresh, 0-divisible phase vectors into 1-divisible ones, 1-divisible phase vectors into 2-divisible ones, and so on, until we find $t$-divisible phase vectors as requested.

For each $\ell$, we let $r_\ell$ denote the largest integer such that $t_1, \ldots, t_{r_\ell} \geq t - \ell$. Consider $r_\ell + 1$ phase vectors $\Psi(\chi_i)$ that are $\ell$-divisible. Each $\chi_i$ is of the form

$$\chi_i : (g_1, \ldots, g_m, h_1, \ldots, h_n) \mapsto e^{2\pi \mathbf{i}\left(\frac{a_{i,1}g_1 + \ldots + a_{i,m}g_m}{p} + \frac{b_{i,1}h_1}{2^{t_1}} + \ldots + \frac{b_{i,n}h_n}{2^{t_n}}\right)}$$

for certain $a_{i,j} \in \mathbb{Z}_p$ and $b_{i,j} \in \mathbb{Z}_{2^{t_j}}$. For all $j \leq r_\ell$ it holds that $2^{\ell + t_j - t} | b_{i,j}$, in view of the $\ell$-divisibility. By defining $c_{i,j} = b_{i,j}/2^{\ell + t_j - t} \bmod 2$, we obtain $r_\ell + 1$ vectors $(c_{i,1}, \ldots, c_{i,r_\ell})$ that are necessarily $\mathbb{Z}_2$-linearly dependent. So we can find integers $d_1, \ldots, d_{r_\ell + 1} \in \{0, 1\}$ such that $d_1 c_{1,j} + \ldots + d_{r_\ell+1} c_{r_\ell+1,j} \equiv 0 \bmod 2$ for $j = 1, \ldots, r_\ell$. We then merge those $\Psi(\chi_i)$'s for which $d_i = 1$, in order to end up with a qubit $\Psi(\chi)$ which, when writing

$$\chi : (g_1, \ldots, g_m, h_1, \ldots, h_n) \mapsto e^{2\pi \mathbf{i}\left(\frac{a_{\chi,1}g_1 + \ldots + a_{\chi,m}g_m}{p} + \frac{b_{\chi,1}h_1}{2^{t_1}} + \ldots + \frac{b_{\chi,n}h_n}{2^{t_n}}\right)},$$

has the property that all ratios $b_{\chi,j}/2^{\ell + t_j - t}$ are even. Note that the unpredictable signs in (3) have no influence on this. We conclude that $\Psi(\chi)$ is $(\ell + 1)$-divisible.

By pipelining this for $\ell = 0, \ldots, t-1$, we obtain our desired routine. Each output requires at most $(r_0 + 1) \cdots (r_{t-1} + 1)$ fresh phase vectors as input. Note, by the way, that the phase vectors $\Psi(\chi_i)$ for which $d_i = 0$ can be recycled.

**Step 3.2.** Next, we describe how to use phase vectors $\Psi(\chi)$ with $\chi^p = 1$ for computing the first $m$ components of the hidden shift $s = (s_1, \ldots, s_m, s'_1, \ldots, s'_n)$. We apply the Hadamard transform, yielding $(1 + \chi(s))/2\,|0\rangle + (1 - \chi(s))/2\,|1\rangle$, and we measure. Each time we measure 1, we learn that the coefficient $1 - \chi(s)$ cannot be zero, hence $\chi(s) \neq 1$. Since $\chi^p = 1$ we can write

$$\chi : (g_1, \ldots, g_m, h_1, \ldots, h_n) \mapsto e^{\frac{2\pi \mathbf{i}}{p}(a_{\chi,1}g_1 + \ldots + a_{\chi,m}g_m)}$$

and thus we see that $a_{\chi,1}s_1 + \ldots + a_{\chi,m}s_m \neq 0$ in $\mathbb{Z}_p$. This implies $(a_{\chi,1}s_1 + \ldots + a_{\chi,m}s_m)^{p-1} = 1$, which gives a linear equation in the monomial expressions of degree $p - 1$ in $s_1, \ldots, s_m$. If we never measure 1, then we know that

$(s_1, \dots, s_m) = (0, \dots, 0)$. In the other case, from [14] we learn that the expected number of tries needed to end up with a full-rank system of linear equations is bounded by

$$p\binom{m+p-2}{p-1} = O(m^{p-1}).$$

This allows us to find the exact values of the degree $p-1$ monomials in $s_1, \dots, s_m$, from which it is easy to determine the vector $(s_1, \dots, s_m)$ up to multiplication by an unknown scalar. We are left with $p-1$ options, which can be tested one by one, for instance by transforming $\Psi(\chi)$ into

$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}\chi(\tilde{s})^{-1}\chi(s)|1\rangle,$$

for some concrete guess $\tilde{s}$, before feeding it to the Hadamard transform (indeed, if we then measure 1, we know that the $\mathbb{Z}_p^m$-part of the guess is wrong), or simply by proceeding with Step 3.3 by trial and error. An alternative option is to equip $G$ with a dummy $\mathbb{Z}_p$-factor, where the hidden shift has known component $s_0 = 1$.

**Step 3.3.** Once we know $s_1, \dots, s_m$, we define

$$f_1'(h_1, \dots, h_n) = f_1(0, \dots, 0, h_1, \dots, h_n),$$
$$f_2'(h_1, \dots, h_n) = f_2(s_1, \dots, s_m, h_1, \dots, h_n).$$

For all $h_1, \dots, h_n$ we have $f_1'(h_1, \dots, h_n) = f_2'(h_1 + s_1', \dots, h_n + s_n')$. Thus we face an instance of the hidden shift problem in $\mathbb{Z}_{2^{t_1}} \times \dots \times \mathbb{Z}_{2^{t_n}}$. We again use our routine from Step 3.1, but this time, instead of creating $t$-divisible phase vectors, we make them $(t-1)$-divisible, so that they are of the form $\Psi(\chi)$ where $\chi^2 = 1$. We can use this to find the parity of $s_1', \dots, s_n'$ in the same way as above, with the bonus that *every* measurement now produces an exact linear equation in the $s_i'$ (in other words, we basically run Simon's algorithm at this stage). This can be used to reduce to the hidden shift problem in a $2^{t-1}$-torsion subgroup; an iterative application eventually retrieves all of $s$.

**Complexity.** The procedure is summarized in Algorithm 1. Overall we need $O(m^{p-1}r_0 \cdots r_{t-1}) = \text{poly}(\log |G|)$ phase vectors $\Psi(\chi)$ to find $s$; recall that we view $p$ and $t$ as constants. The corresponding calls to the standard approach dominate the runtime of our algorithm.

## 4 Collimation for products of cyclic groups

We now describe our version of the collimation sieve for arbitrary finite abelian groups $G$, thereby extending Peikert's method from [19]. We start from a chain of strict inclusions $\{1\} = G_0 \leq G_1 \leq \ldots \leq G_M = G^\vee$, where the groups $G_i$ are such that all quotients $G_i/G_{i-1}$ are cyclic, say isomorphic to $\mathbb{Z}_{k_i}$. Throughout, we fix quotient homomorphisms $q_i : G_i \to \mathbb{Z}_{k_i}$ with kernel $G_{i-1}$. We use these

**Algorithm 1:** Finding hidden shifts in finite abelian $2^t p$-torsion groups

**Input :** $G = \mathbb{Z}_p^m \times \mathbb{Z}_{2^{t_1}} \times \cdots \times \mathbb{Z}_{2^{t_n}}$, with $p$ odd, $t = t_1 \geq t_2 \geq \cdots \geq t_n \geq 1$
Access to phase vectors $\frac{1}{\sqrt{2}}(|0\rangle + \chi(s)|1\rangle)$ for known but uniformly random $\chi \in G^\vee$ and unknown but fixed $s \in G$

**Output:** $s$

1 **for** $\ell$ *from* 0 *to* $t$ **do**
2     Determine $r_\ell$ maximal such that $t_{r_\ell} \geq t - \ell$
3 **end**
4 **if** $m > 0$ **then**
5     Call for $(r_0 + 1)(r_1 + 1) \cdots (r_{t-1} + 1)$ phase vectors
6     **for** $j$ *from* 0 *to* $t - 1$ **do**
7         Divide the phase vectors into groups of size $r_j + 1$
8         Create a $(j + 1)$-divisible phase vector from every such group
9     **end**
10     Repeat Steps 5–10 until we can apply Friedl et al. to obtain $s \bmod pG$
11     Apply **Algorithm 1** on $pG \cong \mathbb{Z}_{2^{t_1}} \times \cdots \times \mathbb{Z}_{2^{t_n}}$
12 **else**
13     Call for $(r_0 + 1)(r_1 + 1) \cdots (r_{t-2} + 1)$ phase vectors
14     **for** $j$ *from* 0 *to* $t - 2$ **do**
15         Divide the phase vectors into groups of size $r_j + 1$
16         Create a $(j + 1)$-divisible phase vector from every such group
17     **end**
18     Apply Simon's algorithm to obtain $s \bmod 2G$
19     Apply **Algorithm 1** on $2G \cong \mathbb{Z}_{2^{t_1-1}} \times \cdots \times \mathbb{Z}_{2^{t_n-1}}$
20 **end**

to define a set of subsets of $G^\vee$, which play the role of the intervals in Peikert's algorithm:

$$\mathcal{A} = \{\, \chi \cdot q_i^{-1}(I) \,|\, \chi \in G^\vee,\ i \in \{1, \ldots, M\},\ I \text{ is an interval in } \mathbb{Z}_{k_i} \,\}.$$

Here, by an interval in $\mathbb{Z}_{k_i}$ we mean the reduction mod $k_i$ of a set of the form $\{a, a+1, \ldots, b\} \subseteq \mathbb{Z}$. Sets of the form $\chi \cdot q_i^{-1}(I)$ are said to be *at level* $i$. Note that if $I$ is a singleton, then such a set is also at level $i - 1$, since it can be rewritten as $\chi' \cdot q_{i-1}^{-1}(\{0, \ldots, k_{i-1} - 1\})$ for an appropriate $\chi' \in G^\vee$.

The algorithm revolves around handling phase vectors of arbitrary length $L$, by which we mean quantum states of the form

$$\Psi = \frac{1}{\sqrt{L}} \sum_{j=0}^{L-1} \chi_j(s) |j\rangle.$$

To each such phase vector we attach a support set $A_\Psi \in \mathcal{A}$ that contains all the $\chi_j$ appearing in it. The *density* of $\Psi$ is then said to be $\delta(\Psi) = L/|A_\Psi|$. For example, considering fresh length-two phase vectors $\Psi(\chi)$ yielded by the standard approach, and taking $A_{\Psi(\chi)} = G^\vee$, we find the tiny density $2/|G^\vee| = 2/|G|$.

**Collimation** is a combination-and-measurement process with the goal of producing phase vectors with an increased density. Concretely, two phase vectors

$$\Psi_1 = \frac{1}{\sqrt{L_1}} \sum_{j_1=0}^{L_1-1} \chi_{j_1}(s)|j_1\rangle \qquad \text{and} \qquad \Psi_2 = \frac{1}{\sqrt{L_2}} \sum_{j_2=0}^{L_2-1} \chi_{j_2}(s)|j_2\rangle$$

with support sets $A_{\Psi_1}$, $A_{\Psi_2}$ are tensored together, to get

$$\frac{1}{\sqrt{L_1 L_2}} \sum_{j_1=0}^{L_1-1} \sum_{j_2=0}^{L_2-1} (\chi_{j_1}\chi_{j_2})(s)|j_1\rangle|j_2\rangle.$$

We then take a partition of $A_{\Psi_1} + A_{\Psi_2} = \bigsqcup_{\ell=1}^{r} A_\ell$ into many disjoint elements of $\mathcal{A}$. For any $j_1, j_2$ we define $\ell_{j_1,j_2}$ to be the unique index $\ell$ such that $\chi_{j_1}\chi_{j_2} \in A_\ell$. We then compute the $\ell_{j_1,j_2}$ for all $j_1, j_2$ in our quantum state:

$$\frac{1}{\sqrt{L_1 L_2}} \sum_{j_1=0}^{L_1-1} \sum_{j_2=0}^{L_2-1} (\chi_{j_1}\chi_{j_2})(s)|j_1, j_2\rangle|\ell_{j_1,j_2}\rangle. \tag{4}$$

Upon measurement of the last register we find a value $\ell$, collapsing the state to

$$\frac{1}{\sqrt{L_3}} \sum_{\chi_{j_1}\chi_{j_2} \in A_\ell} (\chi_{j_1}\chi_{j_2})(s)|j_1, j_2\rangle.$$

We then classically compute a list of all $(j_1, j_2)$ satisfying $\chi_{j_1}\chi_{j_2} \in A_\ell$ and compute a bijection from this list to $\{0, \ldots, L_3 - 1\}$, to find a length $L_3$ phase vector with support set $A_{\Psi_3} = A_\ell$.[3]

In practice, we only combine support sets at the same level $i$. Thus, after taking out some global phase if needed, we can assume $A_{\Psi_1} = q_i^{-1}(\{0, \ldots, a_1\})$ and $A_{\Psi_2} = q_i^{-1}(\{0, \ldots, a_2\})$, so that $A_{\Psi_1} + A_{\Psi_2} = q_i^{-1}(\{0, \ldots, a_1 + a_2\})$. We then use partitions of the form

$$A_{\Psi_1} + A_{\Psi_2} = \bigsqcup_{\ell=1}^{r} q_i^{-1}(\{b_\ell, \ldots, b_{\ell+1}\}) \qquad (b_1 = 0, b_{r+1} = a_1 + a_2).$$

When the intervals $\{b_\ell, \ldots, b_{\ell+1}\}$'s become singletons, one can partition further into sets at level $i - 1$ if wanted, and even beyond; in the end, we will want $L_1$, $L_2$ and the size of the partition to be of a comparable size (denoted by $L$).

The *quality* of a collimation step combining $\Psi_1$ and $\Psi_2$ into $\Psi_3$ is defined as

$$Q = \delta(\Psi_3)/\sqrt{\delta(\Psi_1)\delta(\Psi_2)}.$$

The number of collimation steps required to obtain phase vectors of density $> 1$ has a major influence on the time complexity and is determined largely

[3] Note that a support set $A_\Psi$ is not intrinsic to its phase vector $\Psi$: in principle, it could be any set in $\mathcal{A}$ containing all the $\chi_j$ occurring in $\Psi$. It could therefore be useful to shrink this set after a collimation step, making it as small as possible.

by the quality of the collimation steps. To be more precise, if $Q$ is the average quality of the collimation steps and $F$ is the factor by which the density has to increase, then the number of collimation steps is in $2^{O(\log_Q F)}$. We are interested in the geometric average of the quality, which is $\exp(\mathbb{E}\log Q)$. In Appendix A we prove a lower bound on the expected value of the logarithm of the quality of a collimation step. When applied to $\Psi_1$ and $\Psi_2$ from above, it gives

$$\exp(\mathbb{E}\log Q) \geq \sqrt{L_1 L_2}\frac{\sqrt{(a_1+1)(a_2+1)}}{\min(a_1+a_2+1, k_i)}.$$

Since we want this number to be as large as possible, we want the phase vector lengths $L_1, L_2$ to be as large as possible. The factor $\sqrt{(a_1+1)(a_2+1)}/\min(a_1+a_2+1, k_i)$ is seen to be upper-bounded by 1. If $a_1$ and $a_2$ are of similar sizes then this factor is approximately $\frac{a_1+a_2+2}{2\min(a_1+a_2+1,k_i)}$, which is larger than $1/2$. If $a_1$ and $a_2$ are of very different sizes then the factor can get close to zero, so we typically want to collimate phase vectors with similar-sized support sets.

**Our algorithm.** To find our hidden shift $s$, we repeatedly call for length-two phase vectors $\Psi(\chi)$ using the standard approach. These can be combined into phase vectors of some power-of-2 length $L$, by taking tensor products. We then:

**Step 4.1.** Recursively collimate these low density phase vectors, forming higher density phase vectors, until that density exceeds 1;

**Step 4.2.** Thin out and regularize the resulting states;

**Step 4.3.** Either directly apply a Fourier transform, or tensor some of the regular states together and then apply a Fourier transform, so as to learn $q(s)$ for some non-trivial quotient homomorphism $q$ from $G$;

**Step 4.4.** Reduce to the hidden shift problem in the subgroup $G' = \ker q$, by choosing $s' \in G$ such that $q(s') = q(s)$, and noticing that the functions

$$f_1' : G' \to X : g \mapsto f_1(g), \qquad f_2' : G' \to X : g \mapsto f_2(g+s')$$

hide the shift $s' - s \in G'$.

We then repeat until all of $s$ is found. We discuss Steps 4.2–4.3 in more detail:

**Step 4.2.** In view of the *coupon collector's problem*, when the density of a length $L$ phase vector $\Psi$ exceeds 1 by a big enough factor, it is likely that every $\chi \in A_\Psi$ will occur in $\Psi$. Write $A = A_\Psi$. The idea will be to choose some function $f : \{0,\ldots,L-1\} \to \{1,\ldots,\ell\} \cup \{0\}$ such that for each $i = 1,\ldots,\ell$ and $\chi \in A$ there is exactly one $j \in \{0,\ldots,L-1\}$ such that $\chi_j = \chi$ and $f(j) = i$; see Figure 2 for an illustration. We then compute

$$\frac{1}{\sqrt{L}}\sum_{j=0}^{L-1}\chi_j(s)|j\rangle|f(j)\rangle$$

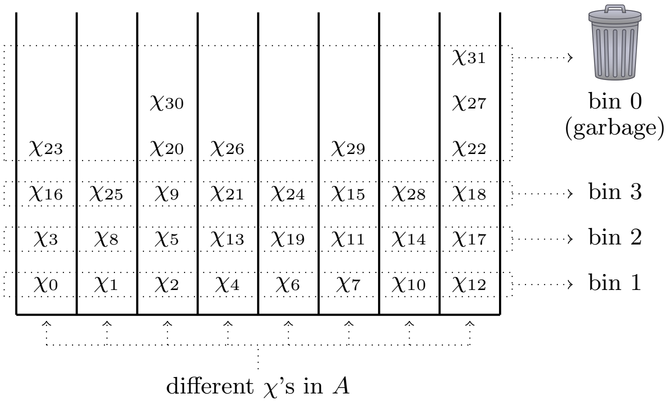


**Fig. 2.** Example construction of the function $f$ from Step 4.2, with $|A| = 8$ and $L = 32$ (with $\ell = 3$), where $f(j) = i$ is illustrated by $\chi_j$ being deposited in bin $i$.

and measure $f(j)$. If we measure 0, the process fails and we have to somehow recycle the phase vector and return to Step 4.1, or throw it away. Otherwise if $i = f(j) > 0$, we know that each $\chi \in A$ occurs exactly once in the resulting expression:

$$\frac{1}{\sqrt{|A|}} \sum_{f(j)=i} \chi_j(s)|j\rangle.$$

We then compute a bijection from $\{\, j \mid f(j) = i \,\}$ to $A$ and apply this using QRACM to obtain $(1/\sqrt{|A|}) \sum_{\chi \in A} \chi(s)|\chi\rangle$.[4]

**Step 4.3.** By multiplying $\Psi$ with a global phase if needed, we can assume that $1 \in A$. If $A$ is a subgroup of $G^\vee$ then the inverse Fourier transform yields $|q(s)\rangle$, where $q$ is the quotient morphism $G \to B := G/\ker A$, with $\ker A$ denoting the joint kernel of the $\chi$'s in $A$.[5] So upon measurement we know $q(s)$. More generally, if $A$ *contains* a non-trivial subgroup $H$ of $G^\vee$ such that $H \in \mathcal{A}$, then $A$ will be a union of cosets of $H$. Through measurement we can collapse $\Psi$ to a phase vector $\Psi'$ supported on one such coset. Again after taking out a global phase if needed, we will have $A_{\Psi'} = H$ and we can act as above.

[4] Remark that, in practice, we can be more lax and allow for incomplete phase vectors, choosing $f$ so that for each $i = 1, \ldots, \ell$ and $\chi \in A$ there is at most one $j \in \{0, \ldots, L-1\}$ such that $\chi_j = \chi$. Then some $\chi \in A$ could be missing in the resulting state, but this is tolerable to some extent. For simplicity, we stick to the complete case.

[5] Note that $A \cong B^\vee$ because each $\chi \in A$ is of the form $\chi' \circ q$ for some $\chi' \in B^\vee$.

Otherwise $A$ is at level 1, i.e., it is an interval $I = \{0, \ldots, a_1 - 1\}$ in $G_1 \cong \mathbb{Z}_{k_1}$. We can write our state $\Psi$ as

$$\frac{1}{\sqrt{a_1}} \sum_{j_1=0}^{a_1-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j_1} |j_1\rangle,$$

where $q : G \to \mathbb{Z}_{k_1}$ is a surjective homomorphism with kernel $\ker G_1$. Our goal is to find $q(s)$. For this we follow Peikert, who aims at finding phase vectors

$$\frac{1}{\sqrt{a_2}} \sum_{j_2=0}^{a_2-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j_2 a_1} |j_2\rangle, \tag{5}$$

leading to a tensor product

$$\frac{1}{\sqrt{a_1}} \sum_{j_1=0}^{a_1-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j_1} |j_1\rangle \otimes \frac{1}{\sqrt{a_2}} \sum_{j_2=0}^{a_2-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j_2 a_1} |j_2\rangle \;=\; \frac{1}{\sqrt{a_1 a_2}} \sum_{j=0}^{a_1 a_2 - 1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j} |j\rangle.$$

Similarly, if we can tensor with a phase vector

$$\frac{1}{\sqrt{a_3}} \sum_{j_3=0}^{a_3-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j_3 a_1 a_2} |j_3\rangle \qquad \text{to obtain} \qquad \frac{1}{\sqrt{a_1 a_2 a_3}} \sum_{j=0}^{a_1 a_2 a_3 - 1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j} |j\rangle,$$

and so on, we eventually find a phase vector

$$\frac{1}{\sqrt{a_1 \cdots a_r}} \sum_{j=0}^{a_1 \cdots a_r - 1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j} |j\rangle$$

with $a_1 \cdots a_r \geq k_1$. Then one measures $\lfloor j/k_1 \rfloor$ so as to obtain either

$$\frac{1}{\sqrt{k_1}} \sum_{j=0}^{k_1-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j} |j\rangle \qquad \text{or} \qquad \frac{1}{\sqrt{t}} \sum_{j=0}^{t-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) j} |j\rangle,$$

for some $t < k_1$. We end up with the first superposition if we measure $\lfloor j/k_1 \rfloor$ to be smaller than $\lfloor a_1 \cdots a_r / k_1 \rfloor$ and with the second if $\lfloor j/k_1 \rfloor$ is measured to equal $\lfloor a_1 \cdots a_r / k_1 \rfloor$.[6] In the first case we take the inverse Fourier transform to retrieve $q(s)$. In the second case we recycle the resulting state, tensoring it with a new vector, and once it is longer than $k_1$ try again.

We now explain how to produce a vector of the form (5); all subsequent steps are analogous. If $a_1$ happens to be coprime to $k_1$ then every phase vector supported on $G_1$ can be written as

$$\frac{1}{\sqrt{L}} \sum_{j=0}^{L-1} e^{\frac{2\pi \mathbf{i}}{k_1} q(s) b_j a_1} |j\rangle. \tag{6}$$

[6] Unless $k_1 \mid a_1 \cdots a_r$ in which case we always get the first superposition.

**Algorithm 2:** Finding hidden shifts in finite abelian groups

**Input :** Finite abelian group $G$, length parameter $L$, density parameter $\delta$
Access to phase vectors $\frac{1}{\sqrt{2}}(|0\rangle + \chi(s)|1\rangle)$ for known but uniformly random $\chi \in G^\vee$ and unknown but fixed $s \in G$

**Output:** $s$

1 Create chain of subgroups $\{1\} = G_0 \leq G_1 \leq G_2 \leq \cdots \leq G_M = G^\vee$ such that the groups $G_i/G_{i-1}$ are cyclic; this gives rise to a type $\mathcal{A}$ of support sets

2 **recursively** *(depth-first)***:**

3 | Create phase vectors $\Psi$ of length $\approx L$ by tensoring $\lceil \log_2(L) \rceil$ fresh length-two phase vectors; endow with support set $A_\Psi := G^\vee$

4 | Increase density by collimating phase vectors $\Psi_1, \Psi_2$ with support sets

$$A_{\Psi_1}, A_{\Psi_2} \text{ of size } \approx |G^\vee|/(L/2)^i$$

into a length $\approx L$ phase vector $\Psi_3$ with support set

$$A_{\Psi_3} \text{ of size } \approx |G^\vee|/(L/2)^{i+1}$$

5 **until** *we obtain a phase vector $\Psi$ with density $\geq 1 + \delta$;*

6 Thin out and regularize $\Psi$

7 If needed, remove global phase so that $1 \in A_\Psi$

8 **if** *$A_\Psi$ contains a non-trivial subgroup $H \leq G^\vee$ contained in $\mathcal{A}$* **then**

9 | Recover $s$ modulo $G' = \ker H$ by applying a quantum Fourier transform

10 | Run **Algorithm 2** on $G'$

11 **else**

12 | Repeat Steps 2–7 to obtain phase vectors with "dilated" support sets inside $G_1$ *(Peikert's method)*

13 | Take tensor products and shorten if needed, to find complete phase vector with support set $G_1$

14 | Recover $s$ modulo $G' = \ker G_1$ by applying a quantum Fourier transform

15 | Run **Algorithm 2** on $G'$

16 **end**

The reason is that every $\chi_j \in G_1$ is of the form $g \mapsto \exp(\frac{2\pi \mathbf{i}}{k_1} q(g) c_j)$ for some integer $c_j$. By adding an appropriate multiple of $k_1$ to $c_j$, we can always assume that $c_j$ is a multiple of $a_1$; this uses that $\gcd(a_1, k_1) = 1$. Then, in complete analogy with standard collimation, through a process of combination and measurement one can turn phase vectors of the form (6) into phase vectors of the desired form (5). When $a_1$ is not coprime to $k_1$, then we slightly shorten $\Psi$, replacing $a_1$ with the greatest integer smaller than $a_1$ that is coprime to $k_1$.

**Special case.** If $G = \mathbb{Z}_{2^n}$ and we start from the chain $1 = G_0 \leq G_1 \leq \cdots \leq G_n = G^\vee$ where each quotient $G_i/G_{i-1}$ has order 2, then our algorithm can be seen to collimate the "least-significant bits" of the phase multipliers; as such we recover Kuperberg's original version of the collimation sieve. On the other hand, if we start from the trivial chain $1 = G_0 \leq G_1 = G^\vee$, then collimation happens on the "most-significant bits" and we recover Peikert's method. See the preamble of [19, §3] for a related discussion.

**Complexity.** Summarizing pseudocode can be found in Algorithm 2. The asymptotic complexity of our algorithm, as well as its analysis, is very similar to that of Peikert's, leading to the statement of Theorem 1.2 in which $|2^t pG|$ is replaced by $|G|$. We omit the details, instead referring to [19], as well as to the next section, which contains a related analysis.

## 5 Merging both algorithms

We now describe our fusion algorithm, resulting in Theorem 1.2. The idea is to apply collimation, as outlined in the previous section, but in our chain of inclusions $\{1\} = G_0 \leq G_1 \leq \ldots \leq G_M = G^\vee$ we now let

$$G_1 = G^\vee[2^t p] = \{\, \chi \in G^\vee \mid \chi^{2^t p} = 1 \,\}$$

be the $2^t p$-torsion subgroup of $G^\vee$, rather than a cyclic group. The goal is to produce length-two phase vectors $\Psi(\chi)$ with $\chi \in G_1$, which can then be used as input to our algorithm from Section 3, ran on the $2^t p$-torsion group

$$G/\ker G_1 = G/2^t pG$$

(of which $G_1$ can be viewed as the dual), which will allow us to find $s \bmod 2^t pG$.

Concretely, we stop collimating as soon as the density exceeds some small multiple of $2/|G_1|$. Consider a resulting phase vector $\Psi$. By measuring to which coset of $G_1$ the $\chi$'s in $A_\Psi$ belong, it collapses to a phase vector

$$\frac{1}{\sqrt{L}} \sum_{j=0}^{L-1} \chi_j(s)|j\rangle$$

where all $\chi_j$'s belong to the same coset of $G_1$. By taking out a global phase if needed we can assume that this coset is $G_1$ itself. We expect $L \geq 2$, because the measurement decreases the numerator and the denominator of $\delta(\Psi)$ by a similar factor. If $L = 2$ then we succeeded, while if $L = 1$ then we failed. If $L > 2$ then measuring $\lfloor j/2 \rfloor$ typically yields a state with only two terms, as wanted. This procedure is summarized in Algorithm 3

Our algorithm then proceeds using $t + 2$ collimation rounds:

**Round 1.** In a first collimation round, we produce length-two phase vectors $\Psi(\chi)$ with $\chi \in G_1$ and feed them as input to Step 3.1, in order to obtain phase vectors $\Psi(\chi)$ with $\chi^p = 1$. These can then be used as input to Step 3.2, allowing us to find $s \bmod pG$.

**Rounds 2 to $t + 1$.** Knowing $s \bmod pG$, we can reduce to a hidden shift problem in the subgroup $pG$. We then proceed as in Step 3.3. That is, we again

**Algorithm 3:** Producing length-two phase vectors supported on the $2^t p$-torsion characters of a finite abelian group

**Input :** Finite abelian group $G$, length parameter $L$, density parameter $\delta$
Access to phase vectors $\frac{1}{\sqrt{2}}(|0\rangle + \chi(s)|1\rangle)$ for known but uniformly random $\chi \in G^\vee$ and unknown but fixed $s \in G$

**Output:** Phase vector $\frac{1}{\sqrt{2}}(|0\rangle + \chi'(s)|1\rangle)$ for known random $\chi' \in G^\vee[2^t p]$

1 Create chain of subgroups $\{1\} = G_0 \leq G_1 = G^\vee[2^t p] \leq G_2 \leq \cdots \leq G_M = G^\vee$ such that $G_i/G_{i-1}$ is cyclic for $i \geq 2$; this invokes a type $\mathcal{A}$ of support sets
2 **repeat**
3 Produce length $\approx L$ phase vector $\Psi$ of density $\approx 2/|G_1|$ using collimation; this is done in full analogy with Steps 2–5 of **Algorithm 2**
4 Restrict the support of $\Psi$ to a single coset of $G_1$ through measurement
5 Shorten $\Psi$ if needed, so that it has length $\leq 2$
6 **until** *length of* $\Psi$ *equals* 2;
7 Take out global phase to rewrite $\Psi$ as $\Psi(\chi')$ for some $\chi' \in G_1$

**Algorithm 4:** Finding hidden shifts in finite abelian groups containing a large $2^t p$-torsion subgroup

**Input :** Finite abelian group $G$, length parameter $L$, density parameter $\delta$
Access to phase vectors $\frac{1}{\sqrt{2}}(|0\rangle + \chi(s)|1\rangle)$ for known but uniformly random $\chi \in G^\vee$ and unknown but fixed $s \in G$

**Output:** $s$

1 Run **Algorithm 1** on $G/2^t pG$, where access to length-two phase vectors is now provided by **Algorithm 3** ran on $G$, rather than the standard approach
2 Run **Algorithm 2** on $2^t pG$

use collimation to produce input for Step 3.1, but as explained in Step 3.3, we now construct $(t-1)$-divisible phase vectors, which can be used to determine $s \bmod 2pG$ by means of Simon's algorithm. We then reiterate, until we find $s \bmod 2^t pG$.

**Round $t+2$.** We are left with solving a hidden shift problem in $2^t pG$, for which we do a complete run of the collimation algorithm from Section 4. Note: only at this stage, we perform Steps 4.2 (thin out and regularize) and 4.3 (Fourier transform).

**Complexity.** Our fusion algorithm is summarized in Algorithm 4. As for the complexity, let us focus on the renewed trade-off between the number of calls to the standard approach and the amount of QRACM needed, when compared to [19]. From Section 3 we know that, for rounds 1 to $t+1$, the required amount of length-two phase vectors $\Psi(\chi)$ with $\chi \in G_1$ is $\text{poly}(\log|G|)$. This means that for the collimation part, the number of fresh length-two phase vectors we need to call for using the standard approach is

$$\mathrm{poly}(\log |G|) \cdot \log L \cdot 2^{O(\log_Q(\delta_{\mathrm{end}}/\delta_{\mathrm{start}}))}.$$

Here $\delta_{\mathrm{start}} = L/|G^\vee| = L/|G|$ is the density of the phase vectors we feed to the collimation phase and $\delta_{\mathrm{end}} = 2/|G_1|$ is the targeted density. The factor $\log L$ comes from the number of length-two phase vectors we need to tensor together to make a phase vector of length $L$, and $Q$ denotes the average quality of a collimation step, which in view of our discussion from Section 4 we estimate as $L/2$. This gives

$$\mathrm{poly}(\log |G|) \cdot \log L \cdot 2^{O(\log_{L/2}(|G|/|G_1|))} \tag{7}$$

oracle queries. One sees: the larger the initial phase vector length $L$, the smaller the number of oracle queries and collimation steps.

However, larger values of $L$ lead to an increase of the resources needed for every collimation step. The dominating cost is in terms of QRACM, which is about importing classical information into a quantum state, or uncomputing it away. Concretely, this is used in all transitions of the form

$$\sum_{j=0}^{r-1} |j\rangle \quad \to \quad \sum_{j=0}^{r-1} |j, c_j\rangle$$

and vice versa, for some set of classically stored values $c_0, \ldots, c_{r-1}$. Following [7, App. A.1] we estimate the corresponding gate cost as $O(rw)$, with $w$ the number of bits of each $c_j$. The main such cost lies in handling (4), leading to the estimate

$$O(L^2 \log L). \tag{8}$$

The optimal balance between (7) and (8) is found by taking $\log L$ on the order of $\sqrt{\log(|G|/|G_1|)} = \sqrt{\log |2^t pG|}$. Together with the fact that round $t+2$ is ran on the group $2^t pG$, this gives the estimates in Theorem 1.2.

## 6 Conclusion

The hidden shift problem plays a central role in the quantum cryptanalysis of several candidate symmetric and public-key cryptosystems. For general finite abelian groups, the best result we have is an algorithm with sub-exponential runtime due to Kuperberg. For special families we have polynomial-time solutions, e.g., for groups of type $\mathbb{Z}_p^n$ for any fixed prime $p$ due to Friedl et al., and for groups of type $\mathbb{Z}_{2^t}^n$ for a fixed integer $t \geq 1$ due to Bonnetain and Naya-Plasencia.

In this paper, we merge the two latter solutions into one polynomial-time quantum algorithm for arbitrary finite abelian $2^t p$-torsion groups (for any fixed $p$ and $t$). We also adapt Kuperberg's most recent sub-exponential time algorithm, called the *collimation sieve*, to work for any finite abelian group $(G, +)$; this extends work of Peikert. Finally, we fuse both results into a single quantum algorithm for solving the hidden shift problem in $G$ in time

$$\mathrm{poly}(\log |G|) \cdot 2^{O(\sqrt{\log |2^t pG|})},$$

while keeping the key advantage of the collimation sieve, namely that the main memory requirements are in terms of quantum random access classical memory; only a polynomial number of qubits is needed. This can be seen as a memory-friendly special case of a result due to Friedl et al. Such results entail a security issue for hard homogeneous spaces when a large torsion subgroup is present; this can be viewed as a modest Pohlig–Hellman type result for the vectorization problem.

Possible tracks for future research include:

1. Making a more detailed complexity analysis, where the goal is to acquire a better understanding of the hidden constants, including how they depend on $p$ and $t$; this should allow for a better assessment of the impact on the security of concrete hard homogeneous spaces;
2. Further reducing the time and memory requirements of our algorithm, e.g., by devising an optimal strategy in choosing our subgroups $G_i$, or by decreasing the QRACM requirements by storing the characters $\chi$ in an extra register as in [7, p. 10-11];
3. Generalizing our results to other kinds of torsion, where the ultimate target is to replace $2^t p$ by any fixed integer $m$; for this, it would suffice to find a way of incorporating the collimation sieve into the work of Friedl et al.

## References


1. T.P. Berger, J. Francq, M. Minier, G. Thomas, *Extended generalized Feistel networks using matrix representation to propose a new lightweight block cipher: Lilliput*, IEEE Transactions on Computers **65**(7), pp. 2074-2089 (2016)
2. X. Bonnetain, M. Naya-Plasencia, *Hidden shift quantum cryptanalysis and implications*, Proceedings of Asiacrypt 2018 Part I, Lecture Notes in Computer Science **11272**, pp. 560-592 (2018)
3. X. Bonnetain, A. Schrottenloher, *Quantum security analysis of CSIDH*, Proceedings of Eurocrypt 2020 Part II, Lecture Notes in Computer Science **12106**, pp. 493-522 (2020)
4. W. Castryck, T. Lange, C. Martindale, L. Panny, J. Renes, *CSIDH: an efficient post-quantum commutative group action*, Proceedings of Asiacrypt 2018 Part III, Lecture Notes in Computer Science **11274**, pp. 395-427 (2018)
5. W. Castryck, J. Sotáková, F. Vercauteren, *Breaking the decisional Diffie–Hellman problem for class group actions using genus theory*, Proceedings of Crypto 2020 Part II, Lecture Notes in Computer Science **12171**, pp. 92-120 (2020)
6. G. Csáji, *A new quantum algorithm for the hidden shift problem in* $\mathbb{Z}_{2^t}^n$, preprint available at https://arxiv.org/abs/2102.04171 (2021)
7. J. Chávez-Saab, J.-J. Chi-Domínguez, S. Jaques, F. Rodríguez-Henríquez, *The SQALE of CSIDH: Square-root Vélu quantum-resistant isogeny action with low exponents*, preprint available at https://eprint.iacr.org/2020/1520 (2020)
8. A. M. Childs, D. Jao, V. Soukharev, *Constructing elliptic curve isogenies in quantum subexponential time*, Journal of Mathematical Cryptology **8**(1), pp. 1-29 (2014)
9. H. Cohen, H.W. Lenstra, *Heuristics on class groups of number fields*, Proceedings of Journées Arithmétiques 1983, Lecture Notes in Mathematics **1068**, pp. 33-62 (1983)

10. J.-M. Couveignes, *Hard homogeneous spaces*, unpublished, available at https://eprint.iacr.org/2006/291
11. F. Gerth III, *The 4-class ranks of quadratic fields*, Inventiones Mathematicae **77**, pp. 489-515 (1984)
12. W. van Dam, S. Hallgren, L. Ip, *Quantum algorithms for some hidden shift problems*, SIAM Journal on Computing **36**(3), pp. 763-778 (2006)
13. M. Ettinger, P. Høyer, *On quantum algorithms for noncommutative hidden subgroups*, Advances in Applied Mathematics **25**(3), pp. 239-251 (2000)
14. K. Friedl, G. Ivanyos, F. Magniez, M. Santha and P. Sen, *Hidden translation and translating coset in quantum computing*, SIAM Journal on Computing **43**(1), pp. 1-24 (2014)
15. S.D. Galbraith, L. Panny, B. Smith, F. Vercauteren, *Quantum equivalence of the DLP and CDHP for group actions*, preprint available at https://eprint.iacr.org/2018/1199 (2018)
16. G. Kuperberg, *A subexponential time quantum algorithm for the dihedral hidden subgroup problem*, SIAM Journal on Computing **35**(1), pp. 170-188 (2005)
17. G. Kuperberg, *Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem*, Proceedings of TQC 2013, Leibniz International Proceedings in Informatics **22**, pp. 20-34 (2013)
18. U.M. Maurer, S. Wolf, *The Diffie–Hellman protocol*, Designs, Codes and Cryptography **19**, pp. 147-171 (2000)
19. C. Peikert, *He gives C-sieves on the CSIDH*, Proceedings of Eurocrypt 2020 Part II, Lecture Notes in Computer Science **12106**, pp. 463-492 (2020)
20. S. Pohlig, M. Hellman, *An improved algorithm for computing logarithms over* GF($p$) *and its cryptographic significance*, IEEE Transactions on Information Theory **24**(1), pp. 106-110 (1978)
21. O. Regev, *A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space*, unpublished, available at https://arxiv.org/abs/quant-ph/0406151
22. A. Rostovtsev, A. Stolbunov, *Public-key cryptosystem based on isogenies*, unpublished, available at https://eprint.iacr.org/2006/145.pdf
23. P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Journal on Computing **26**(5), pp. 1484-1509 (1997)
24. D. R. Simon, *On the power of quantum computation*, SIAM Journal on Computing **26**(5), pp. 1474-1483 (1997). A preliminary version appeared in Proc. of the 35th Annual Symposium on Foundations of Computer Science, pp. 116-123 (1994)
25. A. Stolbunov, *Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves*, Advances in Mathematics of Communication **4**(2), pp. 215-235 (2010)

## Appendix A Expected quality of a collimation step

**Proposition A.1.** *When $\Psi_1$ and $\Psi_2$ are given, the expected value of $1/\delta(\Psi_3)$ has the upper bound*

$$\mathbb{E}\Big(\frac{1}{\delta(\Psi_3)}\Big) \leq \frac{|A_{\Psi_1} + A_{\Psi_2}|}{L_1 L_2},$$

*regardless of how $A_{\Psi_1} + A_{\Psi_2}$ is subdivided, with equality holding as long as every set $A_i$ in the subdivision contains at least one character $\chi_{j_1}\chi_{j_2}$ occurring in*

$\Psi_1 \otimes \Psi_2$. *Using this the expected value of the logarithm of the density can be bounded as*

$$\mathbb{E} \log \delta(\Psi_3) \geq \log(L_1 L_2 / |A_{\Psi_1} + A_{\Psi_2}|).$$

*If $Q$ is the quality of the collimation step then the expected value of the logarithm is bounded as*

$$\mathbb{E} \log Q \geq \log \Big( \sqrt{L_1 L_2} \frac{\sqrt{|A_{\Psi_1}||A_{\Psi_2}|}}{|A_{\Psi_1} + A_{\Psi_2}|} \Big).$$

*Proof.* We have

$$\mathbb{E}\Big(\frac{1}{\delta(\Psi_3)}\Big) = \mathbb{E}\Big(\frac{|A_{\Psi_3}|}{L_3}\Big) = \sum_{i=1}^{p} \mathbb{P}(A_{\Psi_3} = A_i) \frac{|A_i|}{|\{(j_1, j_2) | \chi_{j_1} \chi_{j_2} \in A_i\}|} =$$

$$\sum_{i=1}^{p} \frac{|\{(j_1, j_2) | \chi_{j_1} \chi_{j_2} \in A_i\}|}{L_1 L_2} \frac{|A_i|}{|\{(j_1, j_2) | \chi_{j_1} \chi_{j_2} \in A_i\}|} = \sum_{i=1}^{p} \frac{|A_i|}{L_1 L_2} = \frac{|A_{\Psi_1} + A_{\Psi_2}|}{L_1 L_2}.$$

Note that if $\mathbb{P}(A_{\Psi_3} = A_i) = 0$ then we simply omit that term from the sum. In this case we only have an upper bound on the expected value of one over the density, rather than an equality. This proves the first statement.

To obtain the second statement we use the result from probability theory that

$$\mathbb{E} \left( -\log \left( \frac{1}{X} \right) \right) \geq -\log \mathbb{E} \frac{1}{X},$$

for any random variable $X$ that only assumes positive values. This follows from Jensen's inequality and the fact that $-\log x$ is a convex function on $\mathbb{R}_{>0}$. This can be rewritten as

$$\mathbb{E} \log X \geq -\log \mathbb{E} \frac{1}{X}.$$

The second statement follows from this result by applying it to $X = \delta(\Psi_3)$. The third statement follows from the second and the definition of the quality $Q$. $\square$