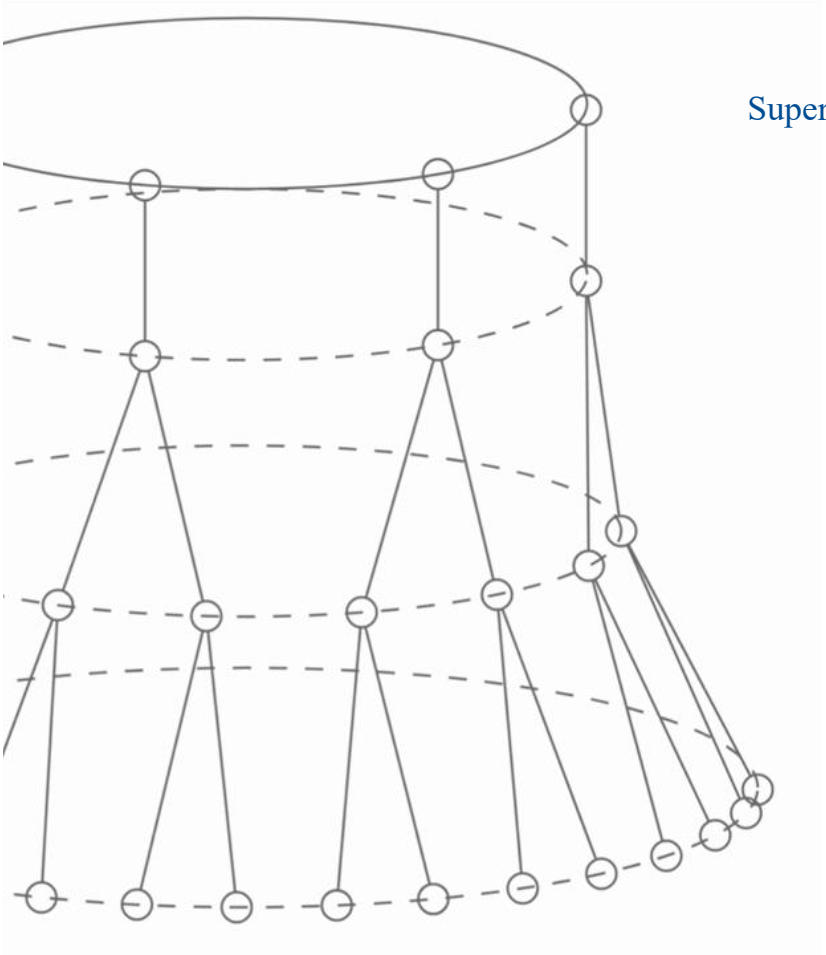
ULB Faculté des Sciences

Improved algorithms of post-quantum cryptographic group actions

Thesis submitted by Valerie Gilchrist
in fulfilment of the requirements of the PhD Degree in Computer Science
("Doctorat en informatique")
Academic year 2025-2026



Supervisor: Professor Christophe PETIT

Improved algorithms of post-quantum cryptographic group actions

by

Valerie Gilchrist

Submitted in Partial Fulfillment of the

Requirements for the Degree

Doctor of Philosophy

Supervised by

Professor Christophe Petit

Department of Computer Science
Université Libre de Bruxelles

Brussels, Belgium

2025

Table of Contents

Acknowledgments

There are many a "**thank you**" I have to give to those who supported me over the years, and who played a role in getting me past the metaphoric doctoral finish line. The first **thank you** is definitely to my supervisor, Christophe Petit. His expertise in the field and enthusiasm to break stuff is what enabled most of the projects I completed over the past 3 years. He has always been patient with me and made himself available when I needed it. I have never had to doubt whether he was "on my side" because of the support he showed me throughout. I cannot thank you enough, Christophe, for your mentorship over these years and also for having taken a bet on me three years ago.

Thank you to my internal doctoral committee, Jean-Michel Dricot and Olivier Markowitch, for their help over the years in making sure my PhD was headed in the right direction. **Thank you** to my external doctoral committee: Ben Smith, Monika Trimoska, Sabrina Kunzweiler, and Wouter Castryck. Their feedback on this thesis was detailed and certainly improved the overall quality of the manuscript. **Thank you** also to all of my coauthors of the works included in this thesis and those works that didn't make it in. A large part of my development as a researcher came from following your examples.

An extra big **thank you** to my parents, Claudia and Andrew. I cannot thank you enough for the sacrifices you made for Alan and I over the years. Without which I certainly would not have been able to indulge in *10 years* of post graduate studies. I am proud of being your daughter, and this PhD is only one of many ways I hope to make you proud as well.

Thank you Alan for proving me wrong enough times as kids that I developed a chronic desire to be right for once. This set me up nicely for a career in research. **Thank you** also for all the visits, phone calls, and for receiving all of my Canadian mail.

Thank you to the whole isogeny crypto gang from around the world! It's been so special to be a part of a community full of so many genuinely nice people. Special thanks to Laurane, Nana, and the Cox reading group. I don't know how I would have survived the imposter syndrome on my own. Y'all are the real ones.

Thank you to the Collective. When I first arrived in Belgium, you were my life line that made the PhD seem doable. Our *extremely* lively lunches and coffee breaks

made me look forward to going to the office everyday. My second life line came from the many people I have gotten to know in Brussels. **Thank you** Ines and Nassim for the great conversations over good food. **Thank you** Marta for our long chats about life and the world, and for introducing me to the Syrian place. **Thank you** Pat for going through this Brussels transition with me and for introducing me to your lovely friends.

Thank you especially to those who have been right there next to me on a daily basis through this entire thing. Ari, **thank you** for the countless pick-me-ups and shared meals. I have learned so much from you, I couldn't have asked for a better roommate. Marco, **thank you** for all our Marcoval time. We have had many an adventure, which really shaped my time in Brussels for the better. Abel, **thank you** for figuring out this PhD thing with me and for being the butt of our jokes. You are so loyal to all of your friends, and I feel privileged to be one of them (I hope). Dani, **thank you** for everything. Thank you for being my hype man when I needed it, and for pushing me to think in new ways when I needed it (of course, always with a silly joke on hand).

Finally, **thank you** to everyone who has supported me from afar. **Thank you** to my London family for all your support and warm wishes, it breaks my heart to miss all the birthdays and special moments. **Thank you** to my Mexicans for always listening to my qualms and for our debates about reality television and reggaeton. **Thank you** to the UW pals for making my first years in research so lovely that I wanted to continue on in research.

This PhD is dedicated to all of you.

Abstract

Our modern world is dependent on online communications and transactions. To ensure privacy in these digital actions, it is extremely common to use protocols such as the Diffie-Hellman key exchange scheme. Unfortunately, these systems have been shown to be vulnerable to attacks from a quantum adversary. This means that an overhaul of the field of cryptography is needed in order to anticipate and mitigate this threat.

Over the past years, the community has begun to answer this call by proposing and analyzing new techniques that are conjectured to be quantum safe. One such example is the study of cryptographic group actions. These mappings mimic some of the structure from the Diffie-Hellman protocol, but abstract it enough so that new mathematical operations can be employed that avoid quantum attacks. Currently, the main examples of group actions come from isogenies, codes, and lattices.

In order for a protocol to be used in the real-world we would hope for it to be secure, fast, and requiring little storage. This means that in these early stages of research and development of cryptographic group actions, it is essential to begin to answer some of the questions surrounding security and efficiency. This thesis contributes to this effort by focusing on these aspects across different group actions that have been proposed for use in the literature.

It is certainly true that each individual group action needs its own study of security. When designers are trying to achieve some advanced functionalities in a protocol, they may sometimes alter the underlying security assumptions. Each time this is done, the new assumptions need their own careful study as well. In this thesis we consider four different variants of group action security assumptions spanning the isogeny, tensor, linear code, and matrix code group actions. We analyze their security by either offering concrete attacks or giving a polynomial time reduction to a more well-studied assumption. This helps to indicate whether a problem is appropriate for use in cryptography, and what parameters to use if so.

In addition to security, this thesis is also inscribed in the effort to study the efficiency of group actions by including three works studying the isogeny group action and related subroutines. They span the core isogeny problem for oriented elliptic curves, evaluating isogenies from a kernel generator, and supersingularity testing. With an effort from the

community to improve overall runtimes, we can hope to get closer to reasonable speeds for real-life use.

By studying the security and efficiency of cryptographic group actions, this thesis shines light on group actions as a promising framework for post-quantum cryptographic solutions.

Contributors and Funding Sources

This work was supervised by a doctoral committee consisting of Professors Christophe Petit, Jean-Michel Dricot, and Olivier Markowitch. This material is based upon work supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium.

The articles included in this dissertation were joint works with other researchers and will be stated clearly when relevant. Note that all of the articles list the contributing authors alphabetically, as is common in the field of cryptography (and more broadly, in mathematics).

Part I

Preliminaries

1 Introduction

Cryptography is an integral part of our modern world. It is used in messaging, online banking, data storage, and connecting to the internet. Suppose, for example, that Alice would like to send a secure message to Bob. The modern approach to doing this has two main steps. The first step is for Alice and Bob to agree on a secret shared key. The second step will be for Bob to use this shared key to encrypt his message so that when it arrives to Alice, she can decrypt using the same shared key and read the message. Anyone who intercepts the message will not have the shared key, and so can't read the contents of the message. Thus, the secrecy of the shared key is pivotal to the security of the protocol.

Among the most common approaches to implement this first step of agreeing on a secret key comes from Diffie and Hellman, via the aptly named *Diffie-Hellman Key Exchange Protocol*. This protocol allows two users to select a shared secret key through communications over public channels.

Example 1 (Diffie-Hellman (DH) key exchange)**.** *Fix a group G together with a generator, g. These make up the public parameters for the protocol.*

Alice and Bob will each select a secret integer a, b respectively. Alice will send g^a to Bob, and Bob will send g^b to Alice. Then using their own secret exponent, they can each compute the shared secret $(g^a)^b = (g^b)^a$.

Here, Alice and Bob publicly sent g, g^a, g^b, and so to ensure security we require it to be infeasible for an adversary to recover a, b or g^{ab} given this information. The task of computing a given (g, g^a) is called the *discrete logarithm problem* (DLP). The task of computing g^{ab} given (g, g^a, g^b) is called the *Computational Diffie-Hellman* (CDH) problem. Thus, in order for the DH key exchange to be secure, we must select a group G and generator g such that the DLP and CDH problems are hard. In the original paper by Diffie and Hellman [48], G was chosen to be the multiplicative group of integers modulo a sufficiently large prime p. Another very common choice of G comes from *elliptic curves*.

Definition 1 (Elliptic curve)**.** *An elliptic curve is an algebraic curve of genus 1 with a distinguished rational point. For certain fields k (i.e. k of characteristic not equal to 2 or 3), the points on a curve E over k is the set $\{(x, y)\} \subset k^2$ satisfying an equation*

$$E : y^2 = x^3 + ax + b,$$

for $a, b \in k$, together with O_E, the "point at infinity".

As it turns out, the set of points on an elliptic curve forms an abelian group, where the operation is additive and O_E serves as the identity element. Using addition in this group, we can define scalar multiplication on a point by repeated adding. Computing discrete logarithms in these groups is conjectured to be hard for classical computers, and yet the arithmetic is relatively fast with low memory requirements. This means we can implement DH using an elliptic curve.

Example 2 (Elliptic curve Diffie-Hellman (ECDH))**.** *Together, Alice and Bob agree on an elliptic curve E, and a generating point P.*

They each select a secret scalar a, b respectively. Alice computes $[a]P$ and sends it to Bob. Bob computes $[b]P$ and sends it to Alice. They each apply their own secret scalar to arrive at the shared secret $[a][b]P = [b][a]P$.

Again, it is integral to ECDH that it be hard to recover a given access to $P, [a]P$. This is called the *Elliptic curve discrete logarithm problem* (ECDLP), and it is currently believed to be a hard problem for classical computers to solve. Hence, so-called elliptic curve cryptography (ECC) has gained much attention as a setting in which the DH protocol can be implemented, along with several other protocols. Currently ECC is widely used in applications such as Transport Layer Security (TLS), Bitcoin, and messaging applications such as WhatsApp, Signal, Facebook Messenger, and Skype.

Unfortunately for cryptographers, it is not possible to *prove* the "hardness" of computational problems like DLP or CDH. So even though DH in integer groups and on elliptic curves have been around for several decades, that is not to say that they are impenetrable to attack. What's more, in 1994, Peter Shor showed in [93] that DLP, as used in Example 1, could be broken by a quantum computer. This may appear to raise some panic considering DH is still widely used today, however, a quantum computer powerful enough to run Shor's algorithm on cryptographically relevant DLPs does not yet exist. The exact timeline of when we can expect such a quantum computer is not clear, but the report from Mosca and Piani [81] indicates that experts estimate it to be between 10 to 30 years. Interestingly, ECDLP is expected to be broken even sooner than other DLP problems because of the short integers that are used. Thus, it is imperative to find alternatives to classical DH before that time arrives.

We call a cryptographic protocol *post-quantum* if it can resist both classical and quantum adversaries. Since this resistance cannot be proven mathematically, the community gains confidence that a scheme can resist attack when many parties have attempted to attack it and failed. It is not only DH that will need to be replaced with

a post-quantum analogue, but *all* cryptography that relies on any group DLP. This is a big ask from the community, and so a large undertaking has begun, searching for new problems that avoid attack from quantum algorithms. Currently, the main candidates come from lattices, hash functions, codes, multivariate polynomials, and isogenies. Many protocols using these structures have already been proposed to meet the demands of different contexts. For example, the American National Institute of Standards and Technology (NIST) has launched long-term international processes to select post-quantum algorithms for standardization of key exchange, digital signatures, and multi-party computations [82].

We return to the problem of updating DH in a post-quantum setting. DH is among the most widely used protocols, and so finding replacements that are quantum-safe is extremely important. Since Shor's algorithm can work in any group, it is not as simple a fix as replacing the group. One approach is to use a *group action*. Briefly, a group action $\star : G \times X \to X$ is an operation where a group is *acting* on a set. Indeed, in Example 1, we had the multiplicative integers modulo $Ord(g)$ *acting* on the set of elements in G. In Example 2, we had the integers modulo $Ord(P)$ acting on the set of elliptic curve points. So each of these examples were in fact group actions, but ones where DLP and CDH are not post-quantum. As we will see later on, commutative group actions will fit easily into DH-like protocols. With this new description of DH from group actions, it leaves room for many more (post-quantum) contexts in which it can be implemented. Specifically, we will be looking for group actions where the set X is not a group (in the hopes of avoiding Shor's attack).

The first, and most studied, example of a post-quantum group action was from elliptic curves and mappings between them called *isogenies*. As we will see in the proceeding chapter, we can implement a DH-like key exchange scheme using group actions. Because of this, they have grown in popularity in the field. We have now seen protocols using group actions from lattices, codes, tensors, trilinear forms, and cubic forms. In fact, we have seen an increase in articles that describe frameworks for particular protocols using group actions. They take some protocol, for example a digital signature, and show how it can be instantiated with a group action. This provides a sort of "copy and paste" protocol where hypothetically a user needs only paste a group action satisfying some particular properties. Of course, it is never as simple as this, and often each choice of group action needs to be specially tailored to the protocol to avoid losing desirable properties related to efficiency, memory, and security. Some examples of these copy and paste frameworks have been made for ring signatures [17, 21, 16], multi-signatures [42], threshold signatures [11], blind signatures [54], VRFs [68], commitment schemes [40, 62, 29], updatable encryption [73, 80], multiparty computation [3, 63], password-authenticated key exchange [1, 79], non-interactive key exchange [52], and even quantum money [99] and quantum encryption [59].

Many of these frameworks tend to stray away from the core DLP or CDH problems, instead using some variants of them. These variants may allow designers to

achieve higher functionalities or improved efficiency but at the cost of relying on lesser known security assumptions. If these protocols are to be used in a real-world setting, they would certainly need careful study from a security perspective. In the same vein, if these schemes are to be implemented on a large scale, their run time must be reasonable. Currently, none of the post-quantum group actions can compete with the speed of classical ECDH.

While analyzing security and improving efficiency may appear to be disjoint tasks, they are actually very closely related. Both require the careful study of protocols, in which case it is not certain if improvements or vulnerabilities will be spotted. In fact, it's not even clear if an improvement to efficiency may later lead to an attack or vice versa. For example, in Chapter 9, we gave several improvements to supersingularity testing algorithms which are relevant to the isogeny group action. As a part of that work, we computed a specific constant that pops up when computing the scalar multiplication of an elliptic curve point in projective coordinates. This formula was later used by Robert [90, Sect. 6.4] in the first attack in years on classical ECDH implementations. We briefly outline it here.

Example 3 (Monodromy leak)**.** *Elliptic curve points can be written in* projective *coordinates $P = (X : Y : Z)$, where two points are said be equivalent if they vary up to a scalar. In other words,*

$$(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z)$$

for any non-zero integer λ. Thus, in ECDH, when a public key $[n]P$ is computed, it is actually computed up to some scalar λ_n. Usually, the last step of the protocol involves dividing out this common factor in order to arrive at a canonical final output.

Among our contributions in Chapter 9, we computed an explicit function f such that $f(n) = \lambda_n$ for a particular choice of elliptic curve model and scalar multiplication formulae. In [90, Sect. 6.4], Robert points out that if λ_n is not factored out before publishing $[n]P$, then the secret n can be recovered directly by computing the preimage of λ_n in f. Even when it is factored out, some information about λ_n could still be recovered using side channel analysis. Robert verifies the correctness of the function f using his so-called cubical *arithmetic, which also provides a framework for recovering n in even more contexts outside of this particular choice of curve model and curve arithmetic formulae.*

In this thesis we address computational challenges of post-quantum group actions, with examples of both security and efficiency analyses. We begin in Chapter 2 with a thorough treatment of the most popular group actions being used in post-quantum cryptography. We define them, recall the state-of-the-art attacks on them, and give a survey of hard problems used in the literature. These associated hard problems must be studied carefully before the wider community can gain confidence in them for real-world use. Towards this end, Part II includes three works of cryptanalysis. Among some

of the most popular group actions in cryptography, we investigate how the hardness of "core" DLP problems compare to the hardness of variants of DLP problems.

The first such work is

Analysis of Code-equivalence Problems and Applications to the Security of Blind Signatures, in collaboration with Laurane Marco, Christophe Petit, and Gang Tang.

It studies two new hard problems that are variants of DLP problems from code-based cryptography. One of the variants is derived from the Linear Code Equivalence (LCE) problem. We show that while the problem may appear to give more information than the core LCE Problem, the additional information is already implied by LCE. As such, we give a polynomial-time reduction between LCE and this variant. The second problem we consider is a variant of the Matrix Code Equivalence (MCE) problem. We demonstrate the importance of not reusing keys by showing how even one reuse can lead to a full key recovery attack.

The second cryptanalytic work considers a group action from *tensors*. These are structures that can be viewed as a list of matrices. The paper

Solving the Tensor Isomorphism Problem for special orbits with low rank points: Cryptanalysis and repair of an Asiacrypt 2023 commitment scheme in collaboration with Laurane Marco, Christophe Petit, and Gang Tang,

considers a commitment scheme that uses tensors. The authors suggest using two very structured tensors as a starting point for the scheme. We show that their choice renders the scheme totally insecure by detailing a polynomial-time classical attack. We also give a repair for their scheme and show that it is statistically binding and computationally hiding.

The final work investigating cryptanalysis is

Another Look at the Quantum Security of the Vectorization Problem with Shifted Inputs, in collaboration with Paul Frixons, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Lam Pham.

In it we explore a DLP variant where in addition to receiving $g, E, g^z \star E$, an adversary also has access to $g^{c_i z} \star E$ for several consecutive integers c_i, and is asked to recover z. We look at two schemes where this problem is used via an elliptic curve group action. We outline how a quantum algorithm from Childs and van Dam can be used to attack this problem, and show concrete improvements over the state-of-the-art for index ranges $i \in [2^8]$ or larger. This reasserts the pattern that variants of DLP problems tend to be strictly weaker than the core problem.

In addition to cryptanalysis, this thesis is also inscribed in the effort to improve the efficiency and memory requirements of algorithms used in group action based cryptography. The efficiency of these algorithms is imperative to the feasibility of implementing them on a large scale. As such, there are three papers included that consider algorithms used in isogeny based cryptosystems.

The first of these works is

Improved algorithms for ascending isogeny volcanos, and applications in collaboration with Steven Galbraith and Damien Robert,

and it is about the core computational isogeny problem. That is, given two elliptic curves, compute an isogeny mapping one to the other. This work only applies to oriented elliptic curves (ordinary or supersingular), but does not require knowledge of the degree of the isogeny. We extend previous work that uses pairings to recover information about the secret isogeny to include vertical isogenies instead of just horizontal ones. We compute asymptotic complexities for the algorithms, and show a small improvement over state-of-the-art in isogeny volcanos with small crater.

The next work is

Fast and Frobenius: Rational Isogeny Evaluation over Finite Fields, in collaboration with Gustavo Banegas, Anaëlle Le Dévéhat, and Benjamin Smith.

In it we consider isogenies that are represented by a kernel generator and how to evaluate points at these isogenies. We decrease the total number of arithmetic operations necessary to do so, and give further savings when the generator is defined over a field extension. This algorithm is central to both computing isogeny walks in protocol design, but also to subroutines used in cryptanalysis (for example, in computing isogenies during group action evaluation algorithms).

The last work is

Efficient supersingularity testing over $\mathbb{F}_p$ and CSIDH key validation, in collaboration with Gustavo Banegas and Benjamin Smith.

Here, we consider a subroutine used in the key exchange scheme CSIDH. As a part of key validation, CSIDH must verify whether an input curve is supersingular prior to executing the rest of the scheme. We improve upon two algorithms from existing literature and show how they compare to the algorithm originally proposed for use by CSIDH. One of these new algorithms gives a deterministic proof of supersingularity, but performs best when the input curves are expected to be ordinary (which is not the case for CSIDH). The second algorithm is probabilistic, but we show that the probability of returning a false positive is very small for cryptographically sized primes, and it has a better run time than the state-of-the-art when the input is expected to be supersingular.

2 Group Actions

We begin with an overview of general group actions, before detailing some prominent examples from the literature.

2.1 General Group actions

In 1990, Brassard and Yung [22] motivated their study of cryptographic group actions using bit commitment schemes. Later in 1997 (released publically in 2006), Couveignes [36] framed his study of the isogeny group action as a search for a *hard homogenous space*. Today, group actions are touted as a useful replacement for the classical Diffie-Hellman problem in a post-quantum setting. As we will see in this section, there are several different examples of cryptographic group actions that have gained attention in the field, leading to different protocols and applications of them. We will give an introductory overview of group actions, their properties, and some of their uses.

Definition 2 (Group action)**.** *For a group G and a set X, a group action is an operation $\star : G \times X \to X$ that satisfies*

- *Identity : let e denote the identity element of G, then $e \star x = x$ for all elements $x \in X$;*
- *Compatibility : for any $g_1, g_2 \in G, x \in X$, then $g_1 \star (g_2 \star x) = (g_1 g_2) \star x$ or $g_1 \star (g_2 \star x) = (g_2 g_1) \star x$. In the former case, the action is called a* left *group action, in the latter it is a* right *group action.*

We say the group G is "acting" on the set X. We can denote such a group action as $(G, X, \star)$.

Depending on the use case, it may be desirable to have a group action satisfying some additional property or properties. The following are some examples of such properties :

- Commutative : the group G is commutative ;
- Transitive : for any $x, y \in X$ there exists $g \in G$ such that $y = g \star x$;
- Faithful : for any $g \in G$, either g is the identity or there exists $x \in X$ such that $x \neq g \star x$;
- Free : $g \in G$ is the identity if and only if there exists some $x \in X$ such that $x = g \star x$;
- Regular : the group action is both free and transitive.

Fix a group action $(G, X, \star)$. Then the *orbit* of an element $x \in X$ is defined to be

$$Orb(x) := \{g \star x : g \in G\}.$$

Further, the *stabilizer* of x is defined as

$$Stab(x) := \{g \in G : g \star x = x\}.$$

Note, we can always achieve transitivity by restricting a group action to include only the orbit of some fixed set element. If this element has a trivial stabilizer, then the action will also be free and thus faithful and regular as well.

With cryptographic applications in mind, Alamati, De Feo, Montgomery, and Patranabis [2] were the first to make the distinction between what they coined an *effective group action* (EGA) and a *restricted* effective group action (REGA). Informally, the former is a group action that can be efficiently computed and thus implemented in cryptographic protocols. The latter relaxes the requirements to be efficiently computable *most* of the time. They showed that in practice a REGA is sufficient for some applications. Formally, the definitions are as follows.

Definition 3 (Effective group action (EGA))**.** *An effective group action is a group action $\star : G \times X \to X$ where both G and X are finite, and there exist efficient polynomial-time algorithms for computing the following tasks.*

(a) Membership testing : decide if a given input represents an element of G (resp. an element of X);

(b) Equality testing : decide if two bit strings represent the same group element;

(c) Sampling : sample an element $g \in G$ according to some distribution $\mathcal{D}_G$;

(d) Group operation : compute gh for any $g, h \in G$;

(e) Group inversion : compute g^{-1} for any $g \in G$;

(f) Unique set representation : for any $x \in X$ compute a canonical representation $\hat{x}$ of x;

(g) Origin : compute the distinguished point $x_0 \in X$ known as the origin;

(h) Group action : for any $g \in G, x \in X$, compute $g \star x$.

Definition 4 (Restricted effective group action (REGA))**.** *A restricted effective group action is a group action $\star : G \times X \to X$, together with some generating set of $G = \langle g_1, \ldots g_n \rangle$, where G is finite and $n = poly(\log(|G|))$, and where there exist efficient polynomial-time algorithms for computing the following tasks.*

(a) Membership testing : decide if a given input represents an element of X;

(b) Unique set representation : for any $x \in X$ compute a canonical representation $\hat{x}$ of x;

(c) Origin : compute the distinguished point $x_0 \in X$ known as the origin;

(d) Group action : for any $i \in [n], x \in X$, compute $g_i \star x$ and $g_i^{-1} \star x$.

Security. In order for group actions to be used in cryptography, we would like to derive some hard problems from them. We generalize the classical Diffie-Hellman key exchange protocol as an example.

Example 4. *Fix a group action $(G, X, \star)$. Select a set element $x \in X$. Then $(\star, G, X, x)$ will be the public parameters.*

Alice and Bob each select secret group elements $g_A, g_B \in G$ respectively. They each compute and publish $g_A \star x, g_B \star x$ respectively. Now Bob can compute $g_B \star (g_A \star x)$ and Alice can compute $g_A \star (g_B \star x)$. If G is commutative, then by compatibility, both Alice and Bob arrive at the shared secret $(g_B \cdot g_A) \star x = (g_A \cdot g_B) \star x$.

We can see from this example that we would like it to be hard for an adversary to compute g_A given access to $(x, g_A \star x)$. We call this problem *vectorization*.

Problem 1 (Vectorization)**.** *Given a group action $(G, X, \star)$, and set elements $x, g \star x$ where $x \in X, g \in G$, recover g.*

This problem is often referred to as the *Group Action Inverse Problem (GAIP)* in the literature.

Even more than that, however, an adversary could still attack the system without recovering the particular secret keys. We call this notion *parallelization*.

Problem 2 (Parallelization)**.** *Given a group action $(G, X, \star)$, and set elements $x, g_1 \star x, g_2 \star x$ where $x \in X, g_1, g_2 \in G$, compute $(g_1 \cdot g_2) \star x$.*

The vectorization problem can always be viewed as an instance of a hidden shift problem. Broadly speaking, the hidden shift problem asks that given two injective functions, f_0, f_1, such that (in multiplicative notation) $f_0(g) = f_1(gs)$ for a secret value s, recover the secret shift s. Thus, given an instance of vectorization $(G, X, \star, x, s \star x)$, we can define the functions

$$f_0 : g \mapsto g \star (s \star x),$$
$$f_1 : g \mapsto g \star x.$$

This problem has been shown to be solvable in quantum subexponential time and super polynomial space. Kuperberg's algorithm to solve this problem [67] works in any finite abelian group and has a complexity of $2^{O(\sqrt{\log |G|})}$ and uses $2^{O(\sqrt{\log |G|})}$ qubits. Regev [87] later gave an algorithm with only polynomial space requirements, but at the cost of an increased run time. It also restricted the use case from any abelian group to only $\mathbb{Z}_{2^n}$, but his algorithm was later generalized by Childs, Jao, and Soukharev to any finite abelian group [32]. This generalized version uses $poly(\log(|G|))$ space and has a complexity of $L_{|G|}(1/2, \sqrt{2}) = 2^{O(\sqrt{\log |G|}\sqrt{\log \log |G|})}$.

Cryptographic group actions. We define a *cryptographic group action* to be a REGA such that *vectorization* and *parallelization* are hard.

The most commonly studied post-quantum cryptographic group actions come from lattices, codes, and isogenies. The Lattice Isomorphism Problem (LIP) was first explored for cryptographic use by [51, 75] but not in the context of a group action. It gained further attention when it was used in the signature scheme HAWK [50], a submission in the NIST [82] standardization competition for digital signatures. It wasn't until the work of [13] that LIP was modeled as a cryptographic group action. It used the language of *quadratic forms* to define the action. Let $\mathcal{L}$ be a lattice with basis B. Then the symmetric matrix $Q = B^T B$ is defined to be the quadratic form of $\mathcal{L}$. We say that two quadratic forms Q, Q' are isomorphic when there exists some unimodular $U \in GL_n(\mathbb{Z})$ such that $Q' = U^T Q U$. This gives rise to a group action where the group $GL_n(\mathbb{Z})/\langle \pm I_n \rangle$ acts on the isomorphism class of a fixed quadratic form.

Proposition 1. *Let Q be a non-zero quadratic form. Let $[Q]$ denote the set of quadratic forms isomorphic to Q. Then the map defined as*

$$\star : GL_n(\mathbb{Z})/\langle \pm I_n \rangle \times [Q] \to [Q],$$
$$V \star Q_0 = V^T Q_0 V,$$

is a group action.

Proof. See [13, Prop. 1]. □

In [13, Prop. 2.3] the group action was proven to be transitive and faithful, as well as free under the condition that Q has a trivial stabilizer group. This formulation lead to further analysis on its security [24, 31] and new interest in building primitives such as commitment schemes [76] from it.

In the proceeding sections we give a more careful treatment of the group actions from codes and isogenies respectively. These families of group actions will be the ones of focus for the remainder of the work.

2.2 Algebraic codes

Throughout, we will be working over the finite field $\mathbb{F}_q$, where q is a prime power. We let $Mat(m \times n, q)$ denote the set of $m \times n$ matrices with entries over $\mathbb{F}_q$.

Definition 5 (Linear code)**.** *An $[n, m]$* linear code *$\mathcal{C}$ is an m-dimensional linear subspace of $\mathbb{F}_q^n$. The elements in the code are called* codewords*.*

An $[n, m]$ linear code $\mathcal{C}$ may be represented by a *generator matrix*. This is a matrix whose rows constitute a basis for $\mathcal{C}$. That is, a matrix $G \in Mat(m \times n, q)$ such that $\mathcal{C} = \{vG : v \in \mathbb{F}_q^m\}$.

With the goal of creating post-quantum cryptography, several different equivalence relations have been defined on linear codes. It is common to select some particular group of matrices, and observe how it acts on the set of linear codes. As was pointed out in [55, Sect. 5], one can formulate a group action on the set of linear codes. It makes use of the set of *monomial matrices*, denoted $Mon(n, q)$. These are matrices of a specific structure, namely $Mon(n, q) := \{DP : D \in D(n, q), P \in S_n\}$, where $D(n, q)$ denotes $n \times n$ invertible diagonal matrices over $\mathbb{F}_q$ and S_n is the set of $n \times n$ permutation matrices.

In defining the group action, we will use, however, the *opposite group* of monomial matrices. That is, the set of monomial matrices together with the group operation defined by

$$A \cdot B = BA.$$

We choose this construction so that (left) compatibility is satisfied.

Proposition 2. *Let $Mon(n, q)^{\mathsf{opp}}$ denote the opposite group of $n \times n$ monomial matrices over $\mathbb{F}_q$, and $G_{Mon} := \left(GL(m, q) \times Mon(n, q)^{\mathsf{opp}}\right)$. Then the map*

$$\star : G_{Mon} \times Mat(m \times n, q) \to Mat(m \times n, q),$$

$$(A, Q) \star G = AGQ,$$

is a group action.

Proof. Note that the $m \times m$ identity matrix $I_m \in GL(m, q)$, and the $n \times n$ identity I_n is also a monomial matrix. Hence (I_m, I_n) acts as an identity element in $(\star)$.

The group action is defined using left-right multiplication, so indeed it also has compatibility since

$$(A_1, Q_1) \star \Big((A_2, Q_2) \star G \Big) = (A_1, Q_1) \star A_2 G Q_2 = A_1 A_2 G Q_2 Q_1 = (A_1 A_2, Q_1 Q_2) \star G.$$

□

By restricting these group actions to one orbit, we can achieve some of the desirable properties from earlier. The matrix generating the orbit, however, must have a trivial stabilizer group. This means that the stabilizer group is exactly the tuple of identity matrices up to scalars.

Proposition 3. *Let $\Lambda := \{(\lambda I, \lambda^{-1} I) : \lambda \in \mathbb{F}_q\}$. Fix $G_0 \in Mat(m \times n, q)$ such that its stabilizer group $Stab(G_0)/\Lambda$ is trivial.*

Let $[G_0] := \{AG_0Q : A \in GL(m, q), Q \in Mon(n, q)\}$ denote the equivalence class of G_0 under monomial equivalence.

Then the group action

$$\star : G_{Mon} \times [G_0] \to [G_0],$$

$$(A, Q) \star G = AGQ,$$

is regular.

Proof. Since we have restricted the group action to one orbit, then any $G_1, G_2 \in [G_0]$ can be written as $G_1 = A_1 G_0 Q_1$ and $G_2 = A_2 G_0 Q_2$. Thus $G_1 = A_1 A_2^{-1} G_2 Q_2^{-1} Q_1$, so $(\star)$ is transitive.

It remains to prove that $(\star)$ is free (which will then directly imply that it is regular). If $(A, Q) \in G_{Mon}$ is the identity, then $(A, Q) = (I_m, I_n)$, and we see that $(A, Q) \star G_0 = I_m G_0 I_n = G_0$. For the converse, let $(A, Q) \in G_{Mon}$ be such that $AGQ = G$ for some $G \in [G_0]$. By transitivity, there exists some $(A_0, Q_0) \in G_{Mon}$ such that $A_0 G_0 Q_0 = G$. Thus

$$A A_0 G_0 Q_0 Q = A_0 G_0 Q_0.$$

Since G_0 has a trivial stabilizer group, we get that $A_0^{-1} A A_0 = \pm I_m$ and $Q_0 Q Q_0^{-1} = \pm I_n$. This directly gives us that $A = \pm I_m$ and $Q = \pm I_n$. □

The hardness of the vectorization problem for this group action is formalized in the *Linear Code Equivalence* (LCE) problem.

Problem 3 (Linear Code Equivalence (LCE))**.** *Let $\mathcal{C}_0, \mathcal{C}_1$ be two linear $[n, m]$ codes with generator matrices $G_0, G_1 \in Mat(m \times n, q)$ respectively. Decide if there exists $(A, Q) \in G_{Mon}$ such that $G_1 = (A, Q) \star G_0$. If they exist, compute A, Q.*

The LCE problem is at the heart of the LESS signature scheme [19], a current Round 2 contestant in the NIST signature competition [82]. Following the publication of LESS, several other schemes have used LCE and its variants to build cryptographic schemes, such as threshold signatures [12], ring and identity-based signatures [8], multi-signatures [41], and blind signatures [55].

The *Permutation Code Equivalence* (PCE) problem is also commonly referenced in the literature, which restricts the group $Mon(n, q)$ to only the set of permutations S_n, i.e. it fixes the diagonal matrix being used to the identity matrix. We state it in terms of generator matrices.

Problem 4 (Permutation Code Equivalence (PCE))**.** *Let $\mathcal{C}_0, \mathcal{C}_1$ be two linear $[n, m]$ codes with generator matrices $G_0, G_1 \in Mat(m \times n, q)$ respectively. Decide if there exists $A \in GL(m, q)$, $P \in S_n$ such that $G_1 = (A, P) \star G_0$. If they exist, compute A, P.*

Currently the state-of-the-art attack on LCE uses Prange's Information-Set Decoding (ISD) algorithm [86]. This attack, first introduced in 1962, aims at finding low Hamming weight code words. With these code words, an algebraic attack can more quickly recover the secret action, as shown in attacks such as [71, 14, 9]. The LESS specification document [7, App. C.3] analyzes how Grover's algorithm can be used to gain a quantum speed-up in the ISD subroutine. However, with the quantum resource cost estimates from NIST, it gives no improvement over the classical attacks.

Works such as [23] and [39] have published attacks on variants of LCE, where an adversary has extra information about a particular instance. For example, it has been shown in the former work that the following problem, used in a ring signature [8], can be solved in polynomial time when $m = n/2$.

Problem 5 (Inverse Linear Code Equivalence (ILCE))**.** *Let $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ be three linear $[n, m]$ codes with generator matrices $G_0, G_1, G_2 \in Mat(m \times n, q)$ such that $G_1 = AG_0M, G_2 = A^{-1}G_0M^{-1}$ for some matrices $A \in GL(m, q)$ and $M \in Mon(n, q)$. Recover M.*

The set of *matrix codes* may also be used in a group action.

Definition 6 (Matrix code)**.** *A $[m \times n, k]$* matrix code *$\mathcal{C}$ is a k-dimensional $\mathbb{F}_q$-linear subspace of $Mat(m \times n, q)$.*

Consider the map vec : $Mat(m \times n, q) \to \mathbb{F}_q^{mn}$ that concatenates all the rows of a matrix. Through this map, we can view an $[m \times n, k]$ matrix code as an $\mathbb{F}_q$-subspace of $\mathbb{F}_q^{mn}$, i.e. a linear $[mn, k]$ code. This means the $[m \times n, k]$ matrix code can be represented with a generator matrix $G \in Mat(mn \times k, q)$. One notion of equivalence on these matrix codes considers how (rank-preserving) isometries act on them. Note, we will again use an opposite group in our construction in order to ensure that (left) compatibility is satisfied.

Proposition 4. *Let $G := GL(m,q) \times GL(n,q)^{\mathsf{opp}}$, and $\mathcal{C}([m \times n, k], q)$ denote the set of k-dimensional $m \times n$ matrix codes over $\mathbb{F}_q$. Then the map*

$$\star : G \times \mathcal{C}([m \times n, k], q) \to \mathcal{C}([m \times n, k], q),$$

$$(A, B) \star \mathcal{C} = A\mathcal{C}B,$$

is a group action.

Proof. The action of $(\star)$ has (rank-preserving) isometries acting on the set of matrix codes, hence the output is also guaranteed to be a matrix code. The identity element is (I_m, I_n), and compatibility follows by the properties of left-right multiplication since

$$(A_1, B_1) \star \Big((A_2, B_2) \star \mathcal{C}\Big) = (A_1, B_1) \star A_2\mathcal{C}B_2 = A_1A_2\mathcal{C}B_2B_1 = (A_1A_2, B_1B_2) \star \mathcal{C}.$$

□

When restricting the group action from Proposition 4 to one specially chosen orbit, the proof of regularity follows very similarly to that of Proposition 3.

The signature scheme MEDS [34], a Round 1 submission to the NIST competition, relies on the hardness of MCE, which we state in terms of the above group action.

Problem 6 (Matrix Code Equivalence (MCE))**.** *Given two $[m \times n, k]$-matrix codes, $\mathcal{C}_0, \mathcal{C}_1$, determine if there exists $(A, B) \in G$ such that $\mathcal{C}_1 = (A, B) \star \mathcal{C}_0$. If so, compute A, B.*

Several analyses have been done on MCE which would support the hardness of vectorization for this group action. The analyses in [34, 35] both use algebraic attacks and also a "Leon-like" algorithm to estimate the security. The Leon-like algorithm, so named from Leon's algorithm [71], refers to building lists of code words with low rank from each of the two public codes, and then looking for collisions among these lists. Attacks such as [37, 88] instead make use of algorithms for attacking equivalent problems, such as the Quadratic Maps Linear Equivalence (QMLE) problem.

As it turns out, matrix codes, which are $k \times m \times n$ objects, can also be viewed as *tensors*. This equivalence will give further support to its conjectured hardness.

2.3 Tensor Isomorphism Family

Fix a finite field $\mathbb{F}_q$ for a prime power q, and k, m, n positive integers. Then given bases $\{e_i\}_{i=1}^k$, $\{f_i\}_{i=1}^m$, $\{g_i\}_{i=1}^n$ of $\mathbb{F}_q^k$, $\mathbb{F}_q^m$, $\mathbb{F}_q^n$, respectively, a *3-tensor* $v \in \mathbb{F}_q^k \otimes \mathbb{F}_q^m \otimes \mathbb{F}_q^n$ is defined as

$$v = \sum_{i=1}^{k} \sum_{j=1}^{m} \sum_{l=1}^{n} v(i,j,l) e_i \otimes f_j \otimes g_l,$$

for elements of $\mathbb{F}_q$ $\{v(i,j,l)\}_{i,j,l}$. Alternatively, a 3-tensor v can be represented as a 3-way array of field elements

$$v = [[v(i,j,l)]]_{i,j,l=1}^{k,m,n} \in \mathbb{F}_q^{k\times m\times n}.$$

These structures can easily be generalized to d-tensors for $d > 3$ in the natural way. However, the most pertinent case to this work, and the case most commonly used in the literature, is precisely $d = 3$.

One common equivalence problem on tensors is concerned with the exact choice of bases $\{e_i\}_{i=1}^k$, $\{f_i\}_{i=1}^m$, $\{g_i\}_{i=1}^n$ of $\mathbb{F}_q^k$, $\mathbb{F}_q^m$, $\mathbb{F}_q^n$. The *Tensor Isomorphism* (TI) problem asks whether two tensors are equal up to a change of basis.

Proposition 5. *Define* $G := GL(k,q) \times GL(m,q) \times GL(n,q)$. *Let* $\mathbf{V}$ *be the tensor space* $\mathbf{V} = \mathbb{F}_q^k \otimes \mathbb{F}_q^m \otimes \mathbb{F}_q^n$.

Define the map

$$\star : G \times \mathbf{V} \to \mathbf{V},$$

$$(A,B,C) \star \sum_{i,j,l=1}^{k,m,n} v(i,j,l) e_i \otimes e_j \otimes e_l = \sum_{i,j,l=1}^{k,m,n} v(i,j,l) Ae_i \otimes Be_j \otimes Ce_l,$$

where all $v(i,j,l) \in \mathbb{F}_q$. *Then* $(\star)$ *is a group action.*

Proof. Since any $(A,B,C) \in G$ is made up of strictly invertible matrices and thus of full rank, then the action maps the bases of $\mathbb{F}_q^k, \mathbb{F}_q^m, \mathbb{F}_q^n$ to new bases. Thus, this result is still a tensor in $\mathbf{V}$.

The identity element for $(\star)$ is naturally (I_k, I_m, I_n), for identity matrices I_i of dimensions $i \times i$ over $\mathbb{F}_q$. Compatibility follows

$$\begin{aligned}
&(A_1,B_1,C_1) \star (A_2,B_2,C_2) \star \sum_{i,j,l} v(i,j,l) e_i \otimes e_j \otimes e_l \\
&= (A_1,B_1,C_1) \star \sum_{i,j,l} v(i,j,l) A_2 e_i \otimes B_2 e_j \otimes C_2 e_l \\
&= \sum_{i,j,l} v(i,j,l) A_1 A_2 e_i \otimes B_1 B_2 e_j \otimes C_1 C_2 e_l \\
&= (A_1A_2, B_1B_2, C_1C_2) \star \sum_{i,j,l} v(i,j,l) e_i \otimes e_j \otimes e_l.
\end{aligned}$$

□

By restricting this group action to the orbit of one tensor $v \in \mathbf{V}$, we can achieve transitivity. Similarly to Proposition 3, if v has a trivial stabilizer group (up to scalars), then the corresponding group action will also be regular. The interesting question becomes which tensors *do not* have a trivial stabilizer group? An example of such a tensor is addressed in Chapter 5.

The hardness of inverting this group action is conjectured in TI.

Problem 7 (Tensor Isomorphism (TI) Problem)**.** *Given two tensors $v_0, v_1 \in \mathbf{V}$, determine if there exists $(A, B, C) \in G$ such that $(A, B, C) \star v_0 = v_1$. If so, compute (A, B, C).*

The algorithm from [74, Thm. 1] solves an average case instance of Problem 7 in $q^{O(n)}$ finite field operations. Their approach takes inspiration from the combinatorial techniques used to solve worst-case instances of the Graph Isomorphism Problem. While TI has not often been used as the direct hard problem in cryptographic schemes, it has been shown to be polynomial-time equivalent to several popular problems. This family of problems has been named TI-complete problems, and includes cubic form equivalence, MCE (Prob. 6) and the Alternating Trilinear Forms Equivalence (ATFE) problem [57]. The latter was used in the signature scheme ALTEQ [97]. We define ATFE here, though we do not study its security directly in this thesis.

Definition 7. *A* trilinear form *is a map $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ that is linear in its arguments. It is* alternating *if $\phi(x, y, z) = 0$ whenever two of $\{x, y, z\}$ are equal.*

The general linear group can act on an alternating trilinear form by acting on each of the input values. This defines a group action in a natural way.

Proposition 6. *Let $ATF(n, q)$ denote the set of all alternating trilinear forms defined over $\mathbb{F}_q^n$. The map*

$$\star : GL(n, q) \times ATF(n, q) \to ATF(n, q),$$

$$A \star \phi(x, y, z) = \phi(Ax, Ay, Az),$$

is a group action.

Proof. Since A is invertible, it acts as a change of basis for the input values, so $A \star \phi$ will still be a trilinear form. It is still alternating since any two of $\{x, y, z\}$ are equal if and only if the same two of $\{Ax, Ay, Az\}$ are equal, thus $A \star \phi$ will evaluate to zero.

The identity element is naturally the identity matrix $I_n \in GL(n, q)$. Compatibility follows since

$$A\star(B\star\phi(x, y, z)) = A\star(\phi(Bx, By, Bz)) = \phi(ABx, ABy, ABz) = (AB)\star\phi(x, y, z).$$

□

The hardness of this group action is conjectured in the following problem.

Problem 8 (Alternating Trilinear Form Equivalence (ATFE))**.** *Let $\phi, \psi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q$ be two alternating trilinear forms. Determine if there exists $A \in GL(n, q)$ such that $\phi = A \star \psi$. If so, compute A.*

Beullens [15] gave heuristic algorithms for solving ATFE (Prob. 8) in dimensions $n = 9, 10, 11$. These attacks can be avoided with increased values of n and q, meaning vectorization for this group action is still believed to be hard for carefully chosen parameters. Since TI (Prob. 7) and MCE (Prob. 6) are proven to be polynomial-time equivalent, Beullens' attack may also apply in instances of these other group actions. However, in [34] it was shown that the attack did not perform any better than the state-of-the-art attack on MCE.

2.4 Elliptic curves

The last group action we will discuss comes from elliptic curves. We already encountered elliptic curves in our examples of classical cryptography. While ECC has been shown to be insecure against quantum adversaries, a post-quantum analogue has developed, based on mappings between elliptic curves called *isogenies*. We give a very brief overview of the core concepts necessary for our exploration of the associated group actions, but a more detailed account can be found in the textbook from Silverman [94]. Again, we will be working over finite fields of the form $\mathbb{F}_q$, for a prime power $q = p^r$, for $r \geq 1$.

Definition 8 (Isogeny)**.** *Fix two elliptic curves E, E' over $\mathbb{F}_q$. An isogeny $\phi : E \to E'$ is a non-constant rational map that fixes the identity point.*

Every isogeny ϕ can be written using the Frobenius map π as $\phi = \phi_{sep} \circ \pi^n$, for $n \geq 0$. If the maximal choice of n is $n = 0$, then we say that ϕ is *separable*. Going forward, we will only consider separable isogenies. In this case, we define the *degree* of these separable isogenies to be the size of its kernel. Namely,

$$\deg \phi := \# \ker \phi = \#\{P \in E(\bar{\mathbb{F}}_q) : \phi(P) = 0\}.$$

The isogeny graph. Consider a graph whose vertices are elliptic curves (up to isomorphism over $\bar{\mathbb{F}}_q$), and whose edges are isogenies. That is, two vertices are connected if and only if their corresponding elliptic curves are isogenous. This graph is called the *isogeny graph*. By limiting the graph to only include edges that correspond to isogenies of degree ℓ, we obtain the *ℓ-isogeny graph*. Interestingly, by looking at the entire ℓ-isogeny graph over extension fields, we can get several structured disjoint components. Some of these components have one cycle, where each vertex in the cycle is a root in a tree. An example of this structure is given in Figure 2.1, where it is more obvious why these components are called *isogeny volcanoes*.

The cycle in the volcano is called the *crater*, and the leaf nodes in the trees form the *floor*. The distance of a vertex from the crater is called the *depth*. Vertices with the same depth will correspond to curves whose endomorphism rings have the same

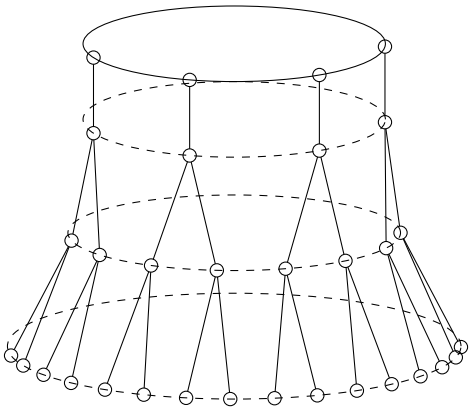


Figure 2.1: Example 2-isogeny volcano.

discriminant. Vertices on the crater correspond to curves whose endomorphism ring have discriminant equal to the fundamental discriminant, Δ. In an ℓ-isogeny volcano, a vertex with depth k will correspond to a curve whose endomorphism ring have discriminant $\Delta' = \ell^{2k}\Delta$. See the exposition by Sutherland [96] for further discussion about isogeny volcanoes.

In other instances there is only one component in the ℓ-isogeny graph that is a finite $(\ell + 1)$-regular Ramanujan graph (with very good mixing properties). See the lecture notes from De Feo [56, Sect. 11] for further discussion about these components.

The endomorphism ring. Now we consider isogenies whose domain and codomains are equal. These isogenies are called *endomorphisms*. Fix some elliptic curve E over $\mathbb{F}_q$. The set of endomorphisms (up to isomorphism) on E, denoted $End(E)$, forms a ring. Take for example the scalar multiplication map

$$[n] : P \mapsto nP.$$

This endomorphism is well defined for any integer n, on any elliptic curve. Thus $\mathbb{Z} \subseteq End(E)$ for all choices of E. Another example is the *Frobenius* map, which for some $r \in \mathbb{Z}$ is defined as

$$\pi^r : (x, y) \mapsto (x^{p^r}, y^{p^r}).$$

If E is defined over $\mathbb{F}_{p^r}$ then π^r is a non-trivial endomorphism of E. Sometimes $End(E) = \mathbb{Z}[\pi]$, but not always. See the doctoral thesis of Kohel [65] or that of Leroux [72] for more details about endomorphism rings of elliptic curves.

In general, $End(E)$ may either be an order in a quaternion algebra, or an order of an imaginary quadratic field. This gives an alternative definition of ordinary and supersingular curves. In the former case, E is supersingular, and in the latter case, E would be ordinary. We give several more definitions of supersingularity, and algorithms for determining whether a curve is supersingular in Chapter 9.

The isogeny group action. We will first focus on the set of ordinary curves. Consider a curve E whose endomorphism ring is isomorphic to some order $\mathcal{O}$. Then $\mathcal{O}$ is an order in $\mathbb{Q}(\sqrt{\Delta}) = End(E) \otimes_{\mathbb{Z}} \mathbb{Q}$. The *ideal class group* of $\mathcal{O}$ is the quotient of invertible fractional ideals, $I(\mathcal{O})$, by the principal fractional ideals, $P(\mathcal{O})$, which we denote by $\mathrm{cl}(\mathcal{O}) = I(\mathcal{O})/P(\mathcal{O})$. We use $[\mathfrak{a}]$ to denote the ideal class of the ideal $\mathfrak{a}$. For a more detailed study of class groups, see the textbook from Cox [38, Sect. 7]. Then with these structures, as was pointed out by Waterhouse[98, Thm 4.5], we can define a group action where the class group is acting on the set of ordinary elliptic curves with endomorphism ring isomorphic to $\mathcal{O}$. This set of curves can alternatively be defined as elliptic curves with complex multiplication (CM) by $\mathcal{O}$.

Let $\mathcal{E}\ell\ell(\mathcal{O})$ denote the set of elliptic curves whose endomorphism ring is isomorphic to $\mathcal{O}$. Let $[\mathfrak{a}] \in \mathrm{cl}(\mathcal{O})$ be an ideal class. Then $[\mathfrak{a}]$ acts on any elliptic curve $E \in \mathcal{E}\ell\ell(\mathcal{O})$ via an isogeny

$$\varphi_{\mathfrak{a}} : E \to E/[\mathfrak{a}],$$

where $\ker(\varphi_{\mathfrak{a}}) = \cap_{\alpha \in \mathfrak{a}} \ker(\alpha)$ and $E/[\mathfrak{a}]$ is the codomain of $\varphi_{\mathfrak{a}}$. This gives rise to the following group action.

Proposition 7. *Following the notation above, define*

$$\star : cl(\mathcal{O}) \times \mathcal{E}\ell\ell(\mathcal{O}) \to \mathcal{E}\ell\ell(\mathcal{O}),$$

$$[\mathfrak{a}] \star E = E/[\mathfrak{a}].$$

Then $(\star)$ is a free and transitive group action.

Proof. See the combined proofs of [98, Thm 4.5] and [92, Thm 4.5]. □

Note that $\mathrm{cl}(\mathcal{O})$ is a commutative group, meaning the group action itself is commutative and thus closely resembles the structure used in Diffie-Hellman. This is the first example of a commutative group action we have seen in this thesis, and indeed, it is rare for a group action to be commutative. This makes the isogeny group action particularly attractive to protocol designers.

Among the first to construct a key exchange scheme using this group action was Couveignes [36] and then Rostovtsev and Stolbunov [91]. These schemes are often referred to together in the literature as CRS, from the authors' initials. The main caveat of CRS is that it is extremely slow, and *far* from realistic for cryptographic use. The work of De Feo, Kieffer, and Smith [45] proposed several optimizations to the original CRS algorithm, one of which was searching for a curve, E_0, such that $\#E_0(\mathbb{F}_{p^r})$ was divisible by as many small primes as possible. This would ensure that the group action can be computed over $\mathbb{F}_{p^r}$, hence, minimizing r was integral to improving the efficiency of the algorithm. Even with these improvements, one group action evaluation still took about 8 minutes to execute.

For this reason, the authors of CSIDH [28] made the astute observation that for a supersingular curve, E_1, it is certain that $\#E_1(\mathbb{F}_p) = p+1$. This makes it easier to select curves with rational torsion for many small primes, meaning the computations can be done over the base field $\mathbb{F}_p$. You may recall, however, that $End(E_1)$ is an order in a quaternion algebra. Thus, in order for the group action to be defined, CSIDH restricted the endomorphism ring to include only $\mathbb{F}_p$-rational endomorphisms. This ensures that the ring will indeed be an order of an imaginary quadratic field. This can be seen in the isogeny graph since including only the $\mathbb{F}_p$-rational isogenies restricts the supersingular graph to being an isogeny volcano of depth 0 or 1, depending on the value of $p \bmod 4$.

The question of computing the isogeny group action in practice, however, is highly non-trivial. Ideals in $\mathrm{cl}(\mathcal{O})$ can (usually) be written as the product of several small prime ideals of the form

$$\mathfrak{a} = \prod_i \mathfrak{l}_i^{e_i},$$

for integers e_i and primes $\mathfrak{l}_i = (\ell_i, \pi - \lambda)$. Using this construction, the group action becomes a restricted effective group action (REGA, see Definition 4). The authors of CSI-FiSh [18] thus opted to precompute a relation lattice of these small primes in order to compute the group action. This precomputation is very costly. It took an estimated 43 core years to compute the CSIDH-512 relation lattice in CSI-FiSh [18]. With access to a quantum computer, however, the relation lattice could be computed in polynomial time.

Use in cryptography. The isogeny group action was used itself in CSIDH [28] to describe a Diffie-Hellman-like key exchange scheme. Since then, the direct vectorization problem, and variants thereof, have been used in numerous protocols such as [5, 18, 25, 47] and many more. In Chapter 6 we consider one instance of how altering the core vectorization problem of the CSIDH group action affects security.

The SCALLOP [44] group action is the same but focuses on imaginary quadratic orders with large, prime conductors inside quadratic fields of small discriminant. This gives more control over the size of the class group and leads to concrete improvements in evaluation time. The follow-up works SCALLOP-HD [30], and later PEARL-SCALLOP [4] and KLaPoTi [85], provided further improvements to the evaluation time of the group action. This change in parameter selection may make the SCALLOP group action vulnerable to different attacks than the CSIDH group action. We give an example of such an attack in Chapter 7, though it does not directly affect any of the proposed parameter sets for any of SCALLOP variants. In Clapoti(s) [84], Robert and Page give a probabilistic algorithm for realizing either the CSIDH or CRS group actions. While their algorithms are polynomial time, they still require the use of dimension 4 or 8 isogenies. This was mostly avoided in KLaPoTi [85] by giving a new norm-solving equation which allows the use of 2-dimensional isogenies when computing the CSIDH group action, giving some efficiency improvements over Clapoti(s). Most recently,

PEGASIS [43] gave a polynomial time algorithm to compute the CSIDH class group action using 4-dimensional isogenies. Despite the dimension, PEGASIS uses smaller primes than KLaPoTi and so appears to be the most efficient effective elliptic curve group action implementation to-date.

Security. The best known classical attack on the CSIDH group action is a meet-in-the-middle attack. This can be done in time $O(\sqrt{\#\mathrm{cl}(\mathcal{O})})$ (assuming keys are sampled from the entire class group), which is exponential in time. The currently best known quantum attack comes from Childs, Jao and Soukharev [32], who framed the problem of recovering a secret isogeny between any two fixed ordinary or oriented elliptic curves as a hidden shift problem. When using a generalised version of Regev's algorithm, along with a subroutine from Bisson [20] for computing the endomorphism ring of a curve, they were able to recover the secret isogeny in time $L_p(\frac{1}{2}, \frac{1}{\sqrt{2}} + \sqrt{2})$ and polynomial space.

However, these estimates rely heavily on the quantum costs of algorithms for computing endomorphism rings, isogenies, and even the group action itself. With recent developments in the field, it is expected that these costs can be significantly improved. Already, simply by swapping out the subexponential subroutine from Bisson, for the polynomial algorithm from Clapoti [84], we may estimate the cost of the algorithm using Regev's approach to be $L_p(\frac{1}{2}, \sqrt{2})$ and requiring polynomial space. It is not yet clear what the actual quantum security of the group action would be with further improvements to subroutines taken from recent literature.

2.5 Variants of group action vectorization

In this section we survey the current literature for protocols that use variants of the group action vectorization problem from Problem 1. These problem statements are written to be easily digestible, and so forgo the usual formality of probability distributions, adversarial games, etc. For more formal statements of these problems, see the protocols themselves which are frequently cited throughout.

In all of the following problems, let $(G, X, \star)$ be a cryptographic group action.

2.5.1 Decisional DH variants

Problem 9 (Group Action Decisional Diffie-Hellman Problem ($\mathsf{GA-DDH}$))**.** *Let $x \in X$, and $g_1, g_2, h \in G$.*

Given $(x, g_1 \star x, g_2 \star x, h \star x)$, determine whether $h = g_1 g_2$ or h was sampled randomly.

This is the decisional version of the parallelization problem from Problem 2. It is central to *many* protocols.

Problem 10 (k-power Group Action Decisional Diffie-Hellman Problem ($\mathsf{kGA-DDH}$))**.** *Let $x_1 \in X$, $1 < k < N$ an integer, and $g \in G$. Given $(k, x_1, g \star x_1, x_2)$, where $x_2 \in X$, determine whether $x_2 = g^k \star x_1$ or whether it was sampled uniformly randomly from X.*

The case $k = 2$ is especially commonly used.

This problem is implemented using isogenies in the quantum money scheme from [99] and the threshold scheme from [46]. The hardness of this problem is supported in [53].

Problem 11 (Master Group Action Decisional Diffie-Hellman Problem ($\mathsf{MGA-DDH}$))**.** *Let $x \in X$, $g_1, \ldots g_n, h \in G$, and $e_1, \ldots e_n \in \mathbb{Z}$.*

Given $(x, g_1 \star x, \ldots g_n \star x, h \star x, \{e_i\})$, determine whether h was randomly sampled or $h = \prod_{i=1}^{n} g_i^{e_i}$.

This problem was used in the verifiable random function Capybara [68] using the isogeny group action. In this setting, the adversary has additional strength in that they have access to an oracle that on input $(v_1, \ldots v_n)$ outputs $\prod_{i=1}^{n} g_i^{v_i} \star x$. The adversary may also select the integer exponents $e_1, \ldots e_n$ so long as they have Hamming weight larger than 1 and have not been queried before.

Problem 12 (Group Action Pseudorandomness Problem (GAPR))**.** *Let $x, x' \in X$.*

Given (x, x') determine whether there exists some $g \in G$ such that $x' = g \star x$.

Note that when $(\star, G, X)$ is transitive, this problem is trivial. It becomes an interesting problem when X is made up of at least two disjoint orbits. In which case it is asking to determine whether x, x' belong to the same orbit.

A group action where this problem is hard was called *weak pseudorandom* in [2]. The tensor group action was shown to be weak pseudorandom in [61]. This problem was also used in [80] to build an updatable encryption scheme. A version of this problem was used in [21] to build a ring signature. That is, given x_0, and $\bar{x} = (x_1, \ldots x_n)$, the adversary is asked to determine whether $\bar{x}$ was sampled from a random distribution, or whether it was sampled from the orbit of x_0.

Problem 13 (Distinguished Decisional Group Action Diffie-Hellman Problem ($\mathsf{ddGA-DDH}$))**.** *Let $x_0, x_1 \in X$ be distinguished elements of X, lying in distinct orbits under the action of G. Let $x_2 \in X$.*

Given (x_0, x_1, x_2) determine whether there exists $g_0 \in G$ such that $x_2 = g_0 \star x_0$ or there exists $g_1 \in G$ such that $x_2 = g_1 \star x_1$.

This problem was used to construct a commitment scheme in [40]. They gave an example instance using the tensor group action. We showed that this instance was insecure due to the choice of distinguished elements in Chapter 5. A similar formulation of this problem was also used to construct a commitment scheme in [62], which was instantiated using the lattice group action.

Problem 14 (Squaring Decisional Group Action Problem ($\mathsf{sdGA-DH}$))**.** *Let* $y, y', x \in X$, $g \in G$.

Given (y, y') *decide whether* y *and* y' *were randomly sampled or whether* $(y, y') = (g \star x, g^2 \star x)$.

This problem was used with the isogeny group action in the construction of the (linkable) ring signature Calamari [17].

2.5.2 Multi-sample DH variants

When a secret group element g is used in a group action computation, the output of which is made publicly available, we call this output a *sample* of g. Thus, when the same secret is used in more than one computation, we call it a *multi-sample* instance of g. This means that an adversary has more information about g, which may lead to weaker security.

Problem 15 (Inverse Group Action Problem (IGAP))**.** *Let* $x \in X, g \in G$.

Given $(x, g \star x, g^{-1} \star x)$*, compute* g.

In the case of the CSIDH group action, this problem is equivalent to pure vectorization because of the *twisting* operation available to elliptic curves. We will discuss twisting later, in Section 2.5.4. However, as we saw in Section 2.2, this problem is called the Inverse Linear Code Equivalence (ILCE) Problem when instantiated using linear codes. ILCE has been shown to be insecure for certain parameter choices, but it is unclear whether it is secure when implemented with other group actions.

In the blind signature construction from [54], they still use the linear code group action but avoid these attacks by including additional randomness. We study this variant in Chapter 4, and show that it is polynomial time equivalent to LCE.

Problem 16 (Modified Inverse Group Action Problem (MIGAP))**.** *Let* $x \in X, g \in G$.

Given $x, c := g \star x$*, and* $c' := g^{-1} \star x$*, compute* $h \in G$ *such that* $h \star c' = c$.

This problem is a relaxed version of IGAP, and was used in the matrix code group action context in the blind signature from [66]. We study its security in Chapter 4.

Problem 17 (Multiple Sample Vectorization)**.** *Let* $g \in G$*, and* $x_1, \ldots x_n \in X$.

Given $\{(x_i, g \star x_i)\}_{i=1}^n$*, recover* g.

This problem is necessary for the practical hardness of the updatable encryption scheme from [73]. For this reason, the authors point out that group actions using linear maps such as the alternating trilinear form or tensor group actions are not amenable to this hard problem.

Problem 18 (Search Linear Hidden Shift Problem)**.** *Let n be a parameter that is polynomially sized in the security parameter. Denote vectors $\mathbf{g} = (g_1, \dots g_n) \in G^n$, and $\mathbf{s} = (s_1, \dots s_n) \in \{0,1\}^n$. Then for any m (that is also polynomial in the security parameter), recover $\mathbf{s} \in \{0,1\}^n$ given*

$$\{(x_i, \mathbf{g}_i, (\prod_{j=1}^{n} g_{ij}^{s_j}) \star x_i)\}_{i \in [m]}$$

where all of the $\mathbf{g}_i \in G^n, x_i \in X$ are sampled independently for $i \in [m]$.

This problem is given in [2, Def. 5.1] along with a decisional version that asks to distinguish sets of this form from a uniformly random one. They also give a search to decision reduction between the problems in Lemma 5.3 of the same work. Further, the authors do not rule out the option of reusing the same $x \in X$ value, instead of sampling a new x_i for each sample.

Problem 19 (One Out of r Group Action Problem))**.** *Let $x_1, \dots x_r \in X$.*

Given $(x_1, \dots x_r)$, compute, if any, $g \in G$ and an index $j \neq 1$ such that $x_1 = g \star x_j$.

This problem is proposed for use in one of the optimizations for the threshold signature Cutting the GRASS [11] and in [16] when constructing a group signature.

Vectorization reduces to this problem. Suppose we are given an instance of vectorization to solve, $(x, x' = g \star x)$. Then, we can sample random $h_1, \dots h_n \in G$, and provide $(x, h_1 \star x', \dots h_n \star x')$ to a solver of the One Out of r Group Action problem. Say we are provided g' such that $g' \star x = h_j \star x'$ for some j. Then we obtain our solution to vectorization, namely $g = h_j^{-1} g'$.

Problem 20 (Vectorization with Shifted Inputs Problem)**.** *Let $x \in X, g \in G$.*

Given

$$x \cup \{(c_i, g^{c_i} \star x)\}_{i=0}^{M},$$

where $\mathbb{C}_M = \{c_0 = 0, c_1 = 1, c_2, \dots, c_M\}$ is a set whose pairwise sums and differences are invertible modulo the order of G, compute g.

This problem is used in the threshold signature CSI-SharK [5] and the ID protocol BCP [6], both of which are instantiated using the isogeny group action.

We analyze the security of this problem in Chapter 6, emphasizing the case when the c_i are consecutive integers.

Problem 21 (ζ_d-Ring Group Action Inverse Problem)**.** *Let N denote the order of G and ζ_d denote the d^{th} root of unity over $\mathbb{Z}_N$. Let $x \in X, g \in G$.*

Given

$$x \cup \{g^{(\zeta_d^j)} \star x\}_{j \in [d]},$$

where $d|\lambda(N)$ (here λ is the Carmichael function), compute g.

This problem was used in the (partially) blind signature CSI-Otter [64] with the isogeny group action.

Problem 22 (n-Chain Group Action DH ($\mathsf{nChain-GA-DH}$))**.** *Let $x_0 \in X$ and $g_0, \dots g_{n-1} \in G$.*

Define $x_{i+1} := g_i \star x_i$ for $i = 1, \dots n$. Given $\{x_i\}_{i=0}^n$, and access to an oracle that given (x_i, x_{i+1}) returns g_i for at most $n-1$ queries, find g such that $x_n = g \star x_0$.

This problem was used in [10] as a generalization of the *One More Discrete Logarithm* Problem from [83].

Problem 23 (Polynomial Group Action Inversion Problem (pGAIP))**.** *Let f be a polynomial of the form $f(x) = a(x+b)^3 + c$. Given $m \in \mathbb{Z}, y \in X, g \in G$ and*

$$g^{f(m)} \star y,$$

compute $(m', g^{f(m')} \star y)$ for some other $m' \in \mathbb{Z}$.

This problem was presented in the isogeny-based OPRF from [47]. In their setting an adversary additionally has access to an oracle that on input of some m_0, outputs $g^{f(m_0)} \star y$. The adversary is then tasked with providing $(m', g^{f(m')} \star y)$ for some m' that has not been queried already. In [69], Lai reduces the problem to a vectorization problem over a smaller group.

2.5.3 Related computational problems

Problem 24 (search Group Action Stabilizer Problem (sGASP))**.** *Let $e \in G$ be the identity element of G.*

Given $x \in X$ such that $Stab(x) \neq \{e\}$, compute $h \in G$ such that $h \star x = x$ and $h \neq e$.

This problem directly underlies the commitment scheme from [62] and the multi-signature scheme from [42]. Though, this problem is indirectly used in several other contexts as well. As an example, it was essential to our attack on a tensor group action implementation in Chapter 5.

Problem 25 (Square Inverse Group Action ($\mathsf{SqInv} - \mathsf{GA}$))**.** *Let $x \in X$, $g \in G$.*

Given $(x, g \star x)$ compute a tuple $(y, y_0, y_1) \in X^3$ such that $y_0 = g^2 \star y$ and $y_1 = g^{-1} \star y$.

This problem is employed in the password-authenticated key exchange scheme from [1].

2.5.4 DH with a twist

Recall that Problem 25 asks an adversary to compute $g^2 \star x$ and $g^{-1} \star x$ given access to $g \star x$. We call the latter operation the *twisting* operation, and $g^{-1} \star x$ is called the *twist* of $g \star x$.

When studying elliptic curves, the (quadratic) twist has a different meaning. Fix a curve $E \in \mathcal{E}\ell\ell(\mathcal{O})$ such that $E : y^2 = f(x)$. Then its quadratic twist is isomorphic to the curve $E' : d \cdot y^2 = f(x)$ where d is any non-square modulo p. This begins to resemble the group action notion of a twist because in general, we have that

$$(g \star E)^t = g^{-1} \star E^t,$$

where we use the t to denote quadratic twisting. In general, the $\mathbb{F}_p$-rational curves used in CSIDH will be isomorphic to their twists over $\mathbb{F}_{p^2}$ but not over $\mathbb{F}_p$. The starting curve, however, is specially selected so that it *is* $\mathbb{F}_p$-isomorphic to its twist. Denote this curve by E_0 (it has j-invariant 1728, and p is $3 \bmod 4$). Then we get that

$$(g \star E_0)^t = g^{-1} \star E_0^t = g^{-1} \star E_0.$$

This is why computing twists is efficient in the CSIDH group action.

Hence, the following problem is easy in the CSIDH group action when x is chosen to be E_0.

Problem 26 (Computational Inverse Group Action Problem)**.** *Given $x, g \star x$, compute $g^{-1} \star x$.*

The hardness of the following problem in the CSIDH group action is used in [70, 68]. It is proven in [70, Prop. 2.2] that this problem is equivalent to Problem 26.

Problem 27 (Reciprocal Group Action Problem)**.** *After committing to $x' \in X$, given $x, g \star x$, compute $(g \star x', g^{-1} \star x')$.*

A group action in which twisting is efficient can be desirable in order to achieve extra functionalities in protocol design. For example, in the signature scheme CSI-FiSh [18], the authors take advantage of the twisting action in the CSIDH group action to decrease soundness and thus decrease the number of necessary repetitions of the

protocol. In the blind signature CSI-Otter [64], the authors use it to achieve a second method for randomizing a set element—something that had been an obstacle to building a Schnorr blind signature using group actions. In [66], the authors also build a blind signature construction emulating the Schnorr blind signature but in the matrix code group action. As such, they must overcome the need for a twisting action, which they do by publishing additional information in both the public key and in the commitment phase. In Chapter 4, we investigate the impact this approach has on security.

2.6 Contributions

Cryptographic group actions provide an exciting framework from which to build post-quantum cryptography. Their mathematical structure is amenable to protocol design, with each one possessing its own pros and cons. The scientific community has already begun using them in numerous different contexts, but in these early stages it is pivotal to study their security carefully before the wider community can adopt them for use in real-world applications. With more confidence in their security, the efficiency of these algorithms will be a central question when deciding when and where to use these schemes.

In what follows, I outline the six precise instances in which I have contributed to the study of cryptographic group actions. The topics covered in these works aim to develop the state-of-the-art of group actions so that we may get closer to using them in a cryptographic setting in the real-world. As such, they assess both the safety of some hard problems derived from group actions, and provide efficiency improvements that would make them more attractive to protocol designers. The corresponding articles are included in their entirety in the proceeding chapters of this thesis.

2.6.1 Cryptanalysis

New cryptosystems are being introduced constantly. Even though we will likely never have a total guarantee of security for any practical protocol, currently the cryptographic community gains trust in it once *many* different parties have assessed its security. This can be through attacks on the system itself, giving more meaning to the choice of parameters, or attempted attacks that only affect unbalanced parameters/variants. After several years of active cryptanalysis attempts, we can hope that a scheme be implemented for public use.

A now infamous example of this process at work comes from the isogeny-based key exchange scheme SIDH [60]. It had made it through to the final round of the NIST key exchange mechanism competition [82], but wasn't yet approved for standardization due to lack of cryptanalysis. Not long after, a series of attacks [26, 78, 89] were published proving that the scheme was totally insecure.

To this end, I have included three works pertaining to cryptanalysis in this thesis. In the papers, we identify hard problems that have already been proposed for use in protocols. All of these problems are derived from core vectorization problems from group actions but add some additional information in order to achieve some extra functionalities. We give analyses on these problems in order to begin the process of determining whether these problems will be safe for use in the long run.

Algebraic code group action. A blind signature is a digital signature where the signer does not know the content of the message they are signing. In Chapter 4, we consider two different blind signatures that rely on group actions from codes. Each of these signatures presented their own variants of linear or matrix code hard problems (respectively). Among other reasons, both works are attempting to avoid the attacks on ILCE (Problem 5) by creating more differences between the two public codes.

In [55], the first signature we consider, the authors used a variant of the Linear Code Equivalence problem (LCE, Problem 3).

Problem 28 (Diagonal-masked Inverse Linear Code Equivalence (DmILCE))**.** *Given* $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ *three linear* $[n, m]$ *codes with generator matrices* $G_0, G_1, G_2 \in \mathbb{F}_q^{m\times n}$*, and* $A_0, A_1 \in GL(m, q), D_0, D_1 \in D(n, q), P \in S_n$ *such that* $G_1 = A_0 G_0 D_0 P, G_2 = A_1 G_1 D_1 P^{-1}$*. Compute* P*.*

At first sight, this problem appears to publish more information than the core LCE problem because it is publishing three interrelated codes instead of two. However, the paper considers equivalence classes of the form $[G]_D := \{AGD : A \in GL(m, q), D \in D(n, q)\}$. Under this definition of equivalence, we have that $[G_0]_D = [G_2]_D$. This means that the "extra" information published in DmILCE does not contribute very much new information. As a result, we give a polynomial-time reduction between DmILCE and LCE. This provides some confidence in the security of DmILCE.

In [66], the second blind signature we consider, the authors use a variant of the Matrix Code Equivalence problem (MCE, Problem 6).

Problem 29 (Modified Inverse Matrix Code Equivalence (MIMCE))**.** *Given three matrix codes,* $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$*, such that* $\mathcal{C}_1 = A\mathcal{C}_0 B^T$ *and* $\mathcal{C}_2 = A^{-1}\mathcal{C}_0 B^{-T}$*, for* $A \in GL(m, q), B \in GL(n, q)$ *(anti)symmetric; find* D, F *(anti)symmetric such that* $\mathcal{C}_2 = D\mathcal{C}_1 F^T$*.*

As is pointed out in the original work, this problem is strictly easier than the core MCE problem because of the use of (anti)symmetric matrices. This reduces the number of unknown variables in an algebraic attack, making it more efficient. We additionally show that if the same secret key is used only twice, this would be sufficient to run a full key recovery attack. The attack on ILCE may be adapted for this instance, but our analysis found that this methodology would require between 6 to 11 repeated uses of the secret key before it can be recovered. Therefore, our tailored attack constitutes an improvement over the state-of-the-art for this problem.

Tensor group action. A commitment scheme allows a user to publicly commit to a secret value, with the ability to reveal the secret value at a later time. These schemes should be *hiding*, meaning the public commitment should not leak anything about the secret value; and *binding*, meaning the user cannot change the secret value they committed to. In [40], the authors provide a framework for building a commitment scheme from non-transitive group actions. As an example instance, they used the tensor group action to implement their construction.

The core idea of the protocol is to select two distinguished tensors belonging to different orbits. A user will select one of the tensors, and apply an isomorphism to it. The conjecture of the paper was that the result would look sufficiently random such that no malicious party could recover the starting tensor. The two distinguished tensors suggested by the paper were very structured owing to the fact that the bases being used did not iterate over all possible indices but just a subset of them. As an example, for dimension $n = 3$, these tensors were of the form

$$t_0 = \sum_{i=1}^{3} e_i \otimes e_i \otimes e_i = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$t_1 = \sum_{i=1}^{2} e_i \otimes e_i \otimes e_i = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

As a first contribution in Chapter 5, we provide a distinguishing attack on this scheme. We make use of the fact that computing rank 0 points is very efficient: these are just linear combinations of the matrices in the tensor such that they sum to the zero matrix. Thus, rank 0 points can be computed using linear algebra. We prove that all tensors in the orbit of t_0 have no rank 0 points. On the other hand, tensors in the orbit of t_1 have one rank 0 point (up to scalar multiplication). This provides an efficient distinguisher for the commitment scheme that directly breaks hiding.

The second contribution of the paper is a computational attack that directly recovers the isomorphism used to compute the commitment. We construct an algebraic attack where we first reduce the solution space by computing known elements of the stabilizer group of t_0 or t_1 respectively. We then prove that part of the isomorphism could be recovered by computing rank 1 points (these are linear combinations of the matrices in the tensor that sum to a rank 1 matrix). The rest of the isomorphism can be solved for using a Groebner basis.

The last contribution in the paper is a repair to their commitment scheme. We suggest replacing t_0 and t_1 by two tensors that are generated uniformly randomly. With overwhelming probability, these tensors will belong to distinct orbits, and are expected to have no low rank point. We show that this protocol is statistically binding and computationally hiding.

CSIDH group action. Both the ID protocol from [6] and the threshold signature from [5] rely on the hardness of the Vectorization with Shifted Inputs Problem (Problem 20). They instantiated the problem using the group action from supersingular elliptic curves defined over $\mathbb{F}_p$, like in CSIDH. In the language of this group action, the problem states that given the pairs $(c_i, [\mathfrak{g}^{c_i z}]E)_{i=0}^{M}$ where $\mathfrak{g}$ is fixed and the elements of $\mathbb{C}_M = \{c_0 = 0, c_1 = 1, c_2, \ldots, c_M\}$ are such that their pairwise sums and differences are coprime, it is hard to recover $z \in \mathbb{Z}_N$.

Neither of the two works gives a security analysis of the problem. This may lead a reader to believe that the hardness is the same as that of one single instance of the group action. In Chapter 6 we describe a quantum attack on this problem. We tailor the algorithm of Childs and van Dam [33] to the problem setting, showing that the extra samples give an advantage to a quantum adversary.

One of the largest subroutines in the Childs-van Dam algorithm can be framed as a knapsack problem in the ℓ_∞-norm. We use this instance to define a lattice, in which we want to recover lattice points within a bounded distance from a target point. In general, the literature on solving such problems focuses mainly on the ℓ_2-norm, meaning we had to adapt the methods to the problem setting. In the end, we settle on two approaches : one from enumeration and one from sieving.

The enumeration approach aims to enumerate all possible candidate lattice vectors inside a bounded cube, and check which candidates satisfy the knapsack requirements. This algorithm is technically simple to describe, and requires very little memory, but the gate costs are very high due to the large volume of candidate solutions.

The sieving approach first samples many elements from the lattice, and then obtains smaller elements by keeping the differences of "close" elements. This algorithm is more efficient than the enumeration approach in terms of quantum gates, but the memory requirements are significant.

We compare our work to the standard quantum attack from Kuperberg and show that starting at around 2^8 samples, we can see improvements in the gate complexity. Overall, the biggest savings come from the fact that the Childs-van Dam algorithm requires significantly less calls to the group action oracle. This work also demonstrates how a quantum algorithm different than Kuperberg can be used in isogeny based cryptography, and more broadly, in group action based cryptography.

2.6.2 Efficiency

The efficiency of algorithms related to these group actions is extremely important to the feasibility of using them in practice. The most obvious algorithm to be considered is the computation of the group action, but subroutines used in this computation or in larger protocols are also important for future use.

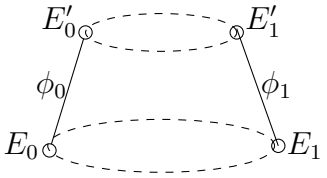
In what follows I outline three works in this thesis that give improvements to algorithms related to the elliptic curve group actions.

Computational isogeny problem. The computational isogeny problem asks for a solution to the vectorization problem in the elliptic curve group action. In [27, 77] the authors showed how pairings on elliptic curve points can be used to recover information about a particular instance of vectorization. These works, as well as ours, discuss the problem in the language of isogenies instead of vectorization. Thus, going forward we will discuss how to recover a secret isogeny.

The authors of [27] gave a careful description of when the Tate pairing can be used to construct a non-trivial self-pairing. Later in [77], the authors gave a more general description of this context using *sesquilinear pairings*, which allowed them to loosen the factorization requirements on the target curve's discriminant. Both of these works focused on recovering *horizontal* isogenies. These are isogenies between curves in the elliptic curve volcano that have the same depth. In Chapter 7, we focus on using pairings to recover *vertical* isogenies. These are ℓ-isogenies between curves that have different depths in the ℓ-isogeny volcano.

We start by giving a simplified framework of [77] using only Weil pairings, making it more accessible to non-experts. This framework expands the use cases to include vertical isogenies instead of just horizontal ones. These pairing computations gain some *partial* information about the secret isogeny. We encode the remaining information into conic equations, and treat the degenerate and non-degenerate cases of these equations individually. Finally, we show how to use Kani's Lemma to recover the final representation of the isogeny.

This computation is then used to solve the more general computational isogeny problem. Namely, given any two curves E_0, E_1, in the volcano, we can first guess the curves E_0', E_1', that lie "directly above" them on the crater. That is, two curves lying on the crater, such that there exist vertical isogenies between them and the target curves. We use our approach using pairings and Kani to recover these vertical isogenies, ϕ_0, ϕ_1, and find a path connecting E_0', E_1', via meet-in-the-middle. See the figure below as reference.



We give exact complexity costs for this algorithm, and obtain a conservative improvement over the state-of-the-art when the volcano has a small crater. We also exemplified how the algorithm could be used in the context of the (PEARL)SCALLOP group action. We get an improvement over the state-of-the-art if the conductor is significantly larger than the fundamental discriminant, but only if the fundamental discriminant is smooth with few factors. These requirements on the parameters are extremely specific, and so do not lead to an attack on any (PEARL)SCALLOP parameter sets. Instead,

it reaffirms the folkloric sentiment that protocols should choose to use elliptic curves whose discriminants are not very smooth.

Isogeny evaluation. There are several ways to represent a (cyclic) isogeny, ϕ. One example is to give a generator, G, for its kernel group. Then, how can this information be used to explicitly evaluate ϕ on some input point? A standard approach is to compute the entire isogeny using the formulae from Vélu. This method is efficient when G is defined over the base field, say $\mathbb{F}_p$. If G is only defined over an extension field of degree k, even though ϕ is defined over $\mathbb{F}_p$, then the computation becomes increasingly more expensive as k increases.

A key aspect of computing isogenies according to Vélu's formulae is the computation of the *kernel polynomial*. This is a polynomial of the form

$$D(X) := \prod_{P \in S} (X - x(P)),$$

where $x(P)$ denotes the x-coordinate of P, and $S \subset \langle G \rangle$ is any subset such that

$$S \cap -S = \emptyset \qquad \text{and} \qquad S \cup -S = \langle G \rangle \setminus \{0\}.$$

The set S is essentially all of the scalar multiples of G up to negation. Computing S could be done by taking the first $\frac{\ell-1}{2}$ multiples of G. This may involve starting with G, and sequentially adding G several times. In Chapter 8 we outline how to improve the efficiency of computing the set S using a *modular differential addition chain* (MDAC). This is a sequence of point operations made to minimize the number of computations necessary.

As an example, suppose 2 is a primitive root modulo ℓ. Then every element modulo ℓ can be written as a power of 2, meaning we can compute S using only point doubling. This gives improvements since an arithmetic model can be chosen where doubling is cheaper than single addition. Cases where 2 is not a primitive root modulo ℓ can be handled on a case-by-case basis so as to still prioritize doublings over additions. If G is defined over $\mathbb{F}_{q^k}$ for $k > 1$, then we can further save by using the Frobenius endomorphism (which is cheaper than doubling or adding). Since the group $\langle G \rangle$ is Galois stable, it means that progressively applying Frobenius to a point will act as multiplication by an eigenvalue of order k. Thus, we need only use an MDAC to compute the generators of the individual Frobenius orbits, and then apply Frobenius to recover the rest of the orbits.

Isogeny computation from a kernel generator plays a role in both protocol design and cryptanalysis. It is used in CSIDH to compute isogeny walks, however, the isogenies can always be computed over the base field $\mathbb{F}_p$. This means that using an MDAC may lead to small improvements, but the additional savings coming from using Frobenius will not be realized. A larger improvement could be seen if implementing CRS using ordinary elliptic curves, where several of the isogenies are expected to be computed

over (hopefully small) field extensions. This work can also be used in cryptanalysis. Currently, computing group action evaluations either quantumly or classically involves a huge number of isogeny evaluations. Thus, any improvements to the cost of isogeny evaluation can help to minimize the total number of arithmetic operations being used.

Supersingularity testing. CSIDH is often touted as one of the only post-quantum non-interactive key exchange schemes. Thus, even though there exists a subexponential quantum attack on CSIDH, it still receives a lot of attention from the community. As such, it continues to be studied and optimized in the hopes of improving its efficiency enough for real-world use. In Chapter 9, we consider the subroutine that is used to verify input public keys. If this step were skipped, it may leave a user vulnerable to attacks from a malicious party.

In CSIDH, the key verification constitutes checking that the curve is indeed an elliptic curve (and not singular) and to check whether it is supersingular. The former is trivially done using the parameters given in the curve equation. Checking for supersingularity, however, is a more involved task. The algorithm originally proposed for this task relies on the fact that the group of $\mathbb{F}_p$-rational points on supersingular curves has order $p+1$ over $\mathbb{F}_p$. Hasse's Theorem [58] states that it is sufficient to sample a point of order larger than $4\sqrt{p}$ in order to prove that the curve has order $p+1$. The algorithm begins by randomly sampling a point P from the curve, and computing $Q_{\ell_i} := [\frac{p+1}{\ell_i}]P$ for each ℓ_i dividing $p+1$. If $Q_i \neq \infty$ but $[\ell_i]Q_i = \infty$ then we can conclude that $\ell_i \mid ord(P)$. With enough iterations we can prove that the order of P is larger than $4\sqrt{p}$, as required by Hasse's Theorem to complete the proof of supersingularity. Using this probabilistic random sampling algorithm as a baseline, in Chapter 9, we identify two more supersingularity testing algorithms that were more efficient either when the input curve is expected to be ordinary or when it is expected to be supersingular.

The first algorithm we consider comes from Sutherland [95]. In it, he makes use of the fact that the ordinary graph is a volcano, and so it is extremely structured. Given an elliptic curve, if it were in a volcano graph, then taking a descending walk in the graph over $\mathbb{F}_{p^2}$ would ultimately terminate when you reach the floor of the volcano. If it were in the supersingular graph, there would be no end to the walk. Thus, by computing the maximum height of an ordinary volcano, we can upper bound how long it should take to reach the floor. If we never reach the floor, then the curve was supersingular. One problem in this algorithm is that it is hard to determine whether a particular walk is descending or not. Thus, in a 2-isogeny graph, the algorithm must use 3 non-backtracking walks in parallel. We improved upon this algorithm by recognizing that the input curve in the CSIDH setting will necessarily be in the $\mathbb{F}_p$ part of the volcano, i.e. at depth 0 or 1. This means that once one of the 3 walks arrives at the $\mathbb{F}_{p^2}$ part, this will be a descending path. We also improved the method for walking in the volcano and the bound on the maximum length of walk in the $\mathbb{F}_{p^2}$ part. It is worth noting that this is the only algorithm outlined in our work that gives a definitive proof of supersingularity.

Overall, this algorithm performed the best when the curves being tested were ordinary since the algorithm terminates quickly. This is not the case in CSIDH, however, where most input curves are expected to be supersingular.

The second algorithm we considered was from Doliskani [49]. He uses a definition of supersingularity that states that a curve is supersingular if and only if its p-torsion group is trivial. Division polynomials offer a method for verifying this definition. Specifically, the m^{th}-division polynomial evaluated at an elliptic curve point will equal 0 if and only if that point is an m-torsion point. This means, that when viewed as a polynomial over $\mathbb{F}_p[X]$, the roots of the polynomial correspond directly to all of the m-torsion points. In the case of supersingular elliptic curves, the p^{th}-division polynomial when viewed over $\mathbb{F}_p[X]$ must not have any roots, and so must be exactly a constant polynomial. Specifically, it should be ± 1. The main caveat is that computing division polynomials becomes increasingly difficult for large values of p—as is the case in cryptographic settings. In our work, we take advantage of the link between division polynomials and scalar multiplication. Namely, that scalar multiplication by p can be written in terms of the p^{th}-division polynomial. Thus, we use one single scalar multiplication applied to a random point over $\mathbb{F}_{p^2}$ to probabilistically check whether the division polynomial is identically ± 1. We prove that the probability of obtaining a false positive is $O(1/p)$, which is extremely small for cryptographically sized p. This significantly simplified Doliskani's algorithm, and ultimately made it the most efficient option for use in CSIDH key validation.

3 Conclusion

Cryptographic group actions continue to gain attention in the community as a building block on which to create cryptography. When commutative, they provide an easy analogue to the Diffie-Hellman key exchange scheme, and an abstract mathematical structure that allows them to be studied in generality and applied to many contexts. In this work, we began by summarizing the key properties of some of the most popular group actions. We also surveyed some of the hard problems derived from group actions that appear in the literature. These problems are still academically young and need more study. This thesis therefore works towards this goal by presenting examples of studies of both cryptanalysis and efficiency.

Cryptanalysis With the number of group action based protocols increasing, it is inevitable that new, sometimes ad hoc, security assumptions will be introduced. Of course, it is sometimes difficult to avoid significant alterations to a core hard problem when designers are trying to achieve advanced functionalities or improve upon particular cost metrics. Thus, using less secure problems does not mean the protocol is not valuable, it simply means that assessing the exact security of these new problems is necessary in order to select parameters appropriately.

In this thesis, we considered four problems (across three articles) that were variants of core group action based hard problems. The problems spanned some of the most popular cryptographic group actions, using structures such as linear codes, matrix codes, tensors, and isogenies. Importantly, all the problems were already proposed for use by protocol developers in published works. Of the four, we showed that three of them had strictly weaker security than their respective core DLP problems. This reasserts the already prevalent idea in the cryptographic community that altering a core problem, be it by adding additional information or otherwise, will decrease the overall security of the problem. These studies help to pinpoint where the exact security defect is, and whether the problems still remain usable for cryptography. While it is not reasonable to expect designers to only use already established DLP problems, hopefully a designer reading this thesis would be reminded of the risks in altering hard problems.

As such, any new assumptions made must be studied independently, and cannot "piggy back" their security off of other problems without a rigorous security reduction.

Efficiency As the security of group action hard problems begins to be better understood, it is important to also develop their use cases in protocols. Improving their efficiency or memory requirements will make them more attractive to protocol designers and cryptographic engineers. It also provides an opportunity to better understand the subtleties of what may be a complicated algorithm, which at times can also lead to improvements in security.

Isogenies, for example, were the first cryptographic group action to gain attention on a larger scale. While the security of isogeny hard problems should definitely continue to be studied, it is also mature enough to begin asking questions of practicality of implementation. Isogeny based cryptography is especially desirable because it requires very little space to store private and public keys, but unfortunately the runtime of most of the protocols is not competitive with alternatives such as those coming from lattice based cryptography. To try to minimize the difference in efficiency, there have been recent developments in improving the runtime of isogeny related algorithms.

In this thesis we studied the efficiency of three algorithms that play a role in isogeny based cryptography. Namely the core computational problem, which is relevant to security; the isogeny evaluation algorithm, which can play a role in both protocol design and cryptanalysis; and supersingularity testing, which is used in CSIDH key validation. With continued effort from the overall community, we can expect the run time of isogeny protocols to continue decreasing.

The studies into security and efficiency presented in this thesis demonstrate that group actions remain an interesting mathematical tool for cryptography with much potential as a solution to quantum adversaries. While these studies begin to answer some of the open problems from the field, much more work is still needed. Going forward, cryptanalysis should be the first priority in addressing the suitability of a scheme to be used in practice. Once a scheme becomes more mature in this regard, efficiency becomes a pressing matter in its own right.

Part II

Cryptanalysis

4 Security Analysis of Code-equivalence Problems

Publication data. Valerie Gilchrist, Laurane Marco, Christophe Petit, Gang Tang. Analysis of Code-equivalence Problems and Applications to the Security of Blind Signatures.

My Contribution. All four authors contributed equally to the theoretical development of the work. Myself and L.Marco split the majority of the writing, coding, and experiments between us.

On the security of two blind signatures from code equivalence problems

Valerie Gilchrist[1], Laurane Marco[2], Christophe Petit[1,3] and Gang Tang[3]

[1] Université Libre de Bruxelles, Belgium
[2] EPFL, Switzerland
[3] University of Birmingham, United Kingdom

Abstract. The Linear Code Equivalence (LCE) problem and the Matrix Code Equivalence (MCE) problem are two examples of code-based hard problems that have gained attention as candidates for use in post-quantum cryptography. They are straightforward to implement, can be viewed as group actions, and offer a good trade-off between compactness and performance in the realm of post-quantum group actions.

With the community gaining confidence in the security of these problems, new variants of these problems have been introduced to achieve particular functionalities in advanced protocols or efficiency improvements. A natural question is then whether the problem variants are as secure as the original ones.

In this work, we consider two problem variants of LCE or MCE.

We first consider a variant based on LCE, and reduce it to the original LCE assumption. This problem was presented in a prior version of the blind signature scheme, proposed by Duong, Khuc, Qiao, Susilo and Zhang [DKQ+25].

Second, we analyse an MCE variant, MIMCE, proposed in the context of another blind signature scheme, by Kutcha, Legrow and Persichetti [KLP25], where we consider a multi-sample version of MIMCE which we solve in polynomial time.

E-mail: valerie.gilchrist@ulb.be (Valerie Gilchrist), laurane.marco@epfl.ch (Laurane Marco), christophe.petit@ulb.be (Christophe Petit), g.tang.1@bham.ac.uk (Gang Tang)

*Authors listed in alphabetical order: see https://www.ams.org/profession/leaders/CultureStatement04.pdf. Valerie Gilchrist is supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium; Christophe Petit is partly supported by EPSRC through grant number EP/V011324/1.

Date of this document: 2025-11-12.

1 Introduction

Group-action cryptography has risen as a natural extension of discrete-logarithm based cryptography in the post-quantum setting. Many protocols are designed in the setting of abstract group actions and later instantiated from concrete problems. Examples of these "cut and paste" group action frameworks include multi-signatures [DFMS24], commitment schemes [DFG23, JWL+25], threshold signatures [BBMP24, BBD+25], and blind signatures [DKQ+25]. Isogeny-based cryptography has commonly been considered the main candidate for such concrete instantiations, with the main caveats being that these systems tend to suffer from slow runtimes and extremely technical implementations.

Recently, more and more proposals using code-based problems have appeared in the community. Code-based protocols are attractive because they are more efficient than their isogeny counterparts, are easier to understand, and thus are also easier to implement correctly. On the other hand, they offer less compact protocols and the resulting group action is non-abelian, which can be restrictive for the design of more advanced protocols. The main problem considered is the *Code Equivalence Problem*, which asks whether two codes belong to the same equivalence class under the action of some group, and, when it is the case, to compute the acting group element. This problem was first introduced and analyzed in [SS13], and then later found applications in the design of digital signature schemes (such as LESS [BMPS20], and then MEDS [CNP+23b]).

Following these initial proposals, the cryptographic community developed code equivalence problems in two orthogonal directions.

The first direction consists in varying the group that acts on the code itself. For example LESS ([BMPS20]) considers the action of the group of monomial matrices $\mathsf{Mono}(n, q)$ on the set of $[n, m]_q$ linear codes (See definition 1). This is called the Linear Code Equivalence problem (LCE, Problem 1). On the other hand, MEDS ([CNP+23b]) considers the action of $\mathrm{GL}(n, q) \times \mathrm{GL}(m, q)$ on the set of $[mn, k]$ matrix codes (or equivalently $[mn, k]_q$ linear codes), which results in the Matrix Code Equivalence Problem (MCE, Problem 5). These problems have further subdivided themselves into sub-categories. For instance, the action of a monomial matrix, as considered in LCE, can be decomposed as the product of a diagonal matrix by a permutation matrix. When setting the diagonal matrix to the identity, one therefore obtains the so-called Permutation Code Equivalence problem (PCE, not studied in this work). In the case of MCE, one can impose further conditions on the acting matrices of $\mathrm{GL}(n, q) \times \mathrm{GL}(m, q)$, for example imposing their symmetry, or anti-symmetry, and considering the action of $\mathrm{Sym}(m, q) \times \mathrm{Sym}(n, q)$ instead. This observation underlines the MIMCE problem, which will be studied in section **??**. Or, one can also consider the action of $\mathrm{GL}(n, q) \times \mathrm{GL}(m, q)$ on distinguished low-rank elements as suggested in [DFG23], although this has been proven insecure in [GMPT24].

The second direction in which these code-equivalence problems have evolved is that of considering inverse and multi-sample problems.
More specifically, inverse problems look into the following scenario: given three codes, where two of them are related to a third one by a group element g and its inverse respectively, how hard is it to compute g? We illustrate this situation in Figure 3(a), where $\mathcal{C}_i$ are linear codes and $g \in \mathsf{G}$ is a group element.

This gives rise to the Inverse Linear Code Equivalence problem ILCE (Problem 3), when the $\mathcal{C}_i$'s are linear codes and $\mathsf{G} = \mathsf{Mono}(n, q)$, and its counterpart for matrix codes IMCE (Problem 6) when the $\mathcal{C}_i$'s are matrix codes and $\mathsf{G} = GL(m, q) \times GL(n, q)$. ILCE has been shown to be polynomially solvable when $m = n/2$ by [BCD+24], but no such results exists for IMCE. To overcome the attack on ILCE, [DKQ+25] introduces the Diagonal-masked Inverse Linear Code Equivalence Problem (Problem 4), a variant of ILCE which will be studied in Section 3. Similarly, [KLP25] introduce the MIMCE problem, a variant of IMCE which instantiates our diagram with the $\mathcal{C}_i$'s as matrix codes and $\mathsf{G} = \mathrm{Sym}(m, q) \times \mathrm{Sym}(n, q)$.

(a) Inverse Problem (b) t-sample Problem

Figure 3: Illustration of Inverse and t-sample Code Equivalence Problem

A natural extension of the inverse problem scenario, is the following: what if we give two pairs of codes related by the same group elements? Does that make the problem easier, and if so by how much? And what about the case where one were to give t such additional pairs of codes? This is what we call t-sample problems, which we illustrate in Figure 3(b).

[BCD+24] showed that the 2-LCE Problem (i.e $t = 2$ above, Problem 2) is solvable in polynomial-time when $n = m/2$. Additionally, [BCD+24] also gives bounds on the number of samples t needed to solve the t-MCE problem. We will investigate similar results for MIMCE.

We summarize this discussion in Table 1.

Table 1: Overview of code equivalence problems

Code Type	Linear $[n, m]_q$		Matrix $[nm, k]_q$	
Group acting	$\mathsf{Mono}(n, q)$	$S_n(q)$	$\mathrm{GL}(n, q) \times \mathrm{GL}(m, q)$	$\mathrm{Sym}(n, q) \times \mathrm{Sym}(m, q)$
Problem	LCE	DmILCE	MCE	MIMCE
Usage	Signature [BMPS20]	Blind signature [DKQ+25]	Signature [CNP+23b]	Blind signature [KLP25]
Hardness	Hard [BBPS23]	Section 3	Hard [CNP+23a]	Section ??
Inverse Problem	ILCE, ring signatures [BBN+22], broken for $n = m/2$	-	IMCE Hard [CNP+23a]	-
$t-$sample Problem	2-LCE, Threshold signatures [BBMP24] broken for $n = m/2$	Section 3	Hard for small enough t [BCD+24]	Section 4

Our contributions We begin with a summary of the relevant computational problems in Section 2, including a formal reduction between two problems that will be of interest to us later on. In Section 3 we study a variant of the LCE problem. Namely, we will study the DmILCE problem which was introduced in Versions 1 and 2 of the blind signature scheme [DKQ+25][1]. We show that these problems are polynomial-time equivalent,

[1]Shortly after the original version of our work was released, [DKQ+25] updated their own paper to alter the formulation of the DmILCE problem.

strengthening confidence in this hardness assumption for future use in cryptography.

Then in Section 4, we consider a variant of the MIMCE problem where an adversary has access to several samples. We first adapt the result of Budroni et al. [BCD+24] for this problem variant, showing that between 6-11 samples (depending on the code dimension) are needed to recover the secret in polynomial time. We improve upon this result by providing a polynomial-time algorithm that recovers the secret using only 2 samples, regardless of the dimension. This discourages attempts at building cryptographic primitives such as ring, group, or threshold signatures from the multiple sample version of this problem.

2 Preliminaries

Notations Throughout this work, $\mathbb{F}_q^{m\times n}$ denotes the set of $m \times n$ matrices with entries in $\mathbb{F}_q$; $D(n, q)$ denotes the set of $n \times n$ invertible diagonal matrices over $\mathbb{F}_q$; S_n denotes the set of $n \times n$ permutation matrices; $\mathsf{Mono}(n, q)$ denotes the set of $n \times n$ matrices that are the product of a diagonal matrix by a permutation matrix , called *monomial matrices*; I_n denotes the $n \times n$ identity matrix.

2.1 Algebraic codes

In this paper we will give an analysis of several computational problems proposed for use in cryptography that employ algebraic codes. The problems of interest to us will use two main families of codes: *linear codes* and *matrix codes*.

Definition 1 (Linear code)**.** An $[n, m]_q$ *linear code* $\mathcal{C}$ is an m-dimensional linear subspace of $\mathbb{F}_q^n$. The elements in the code are called *codewords*.

Definition 2 (Matrix code)**.** A $[m \times n, k]$ *matrix code* $\mathcal{C}$ is a k-dimensional $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q^{m\times n}$.

Let $vec(\cdot)$ be the map that concatenates the rows of a matrix together. That is,

$$vec : \mathbb{F}_q^{m\times n} \to \mathbb{F}_q^{mn}, \quad \begin{bmatrix} -\vec{v}_1- \\ \vdots \\ -\vec{v}_m- \end{bmatrix} \mapsto \vec{v}_1||\cdots||\vec{v}_m$$

Under this map an $[m \times n, k]$ matrix code can also viewed as an $[mn, k]$ linear code. One can represent a code by a *generator matrix*.

Definition 3 (Generator matrix)**.** Given a $[n, m]$ linear code $\mathcal{C}$, a matrix $G \in \mathbb{F}_q^{m\times n}$ such that $\mathcal{C} = \{G^T m : m \in \mathbb{F}_q^n\}$ is called a generator matrix for $\mathcal{C}$.

As mentioned in the introduction, code equivalence problems share a common general framework: they study the action of a group G on a certain set of codes under some equivalence relation. The modularity of the problems resides on which group is acting, which type of code is used and how the equivalence relation is specified, and we survey relevant problems in the next paragraphs. We keep our discussion brief, but a more careful description of these topics can be found in [BMPS20] for linear codes, and in [CNP+23b] for matrix codes.

2.1.1 Linear codes

The Linear Code Equivalence (LCE) problem focuses on the action of $\mathsf{G} = \mathsf{Mono}(n, q)$, the group of monomial matrices, on the set of $[n, m]_q$ linear codes under the equivalence relation $[\mathcal{C}_0]_\sim = \{AG_0 : A \in \mathrm{GL}(m, q)\}$, where G_0 is the generator matrix of $\mathcal{C}_0$. Note that

this equivalence relation is trivial on the code itself, since generator matrices are always defined up to a change of basis. Hence, we say that two codes $\mathcal{C}_0, \mathcal{C}_1$ are equivalent under the action of monomial matrices, if $[C_1]_\sim = [C_0 M]_\sim$ for some $M \in \mathsf{Mono}(n, q)$.
Note that we can decompose a matrix $M \in \mathsf{Mono}(n, q)$ as $M = DP$ for $D \in D(n, q), P \in S_n$. This decomposition will allow us to better highlight the links between the different code equivalence problems. We formalize the LCE problem below.

Problem 1 (Linear Code Equivalence (LCE))**.** *Let $\mathcal{C}_0, \mathcal{C}_1$ be two linear $[n, m]_q$ codes with generator matrices $G_0, G_1 \in \mathbb{F}_q^{m \times n}$ respectively. Decide if there exists $A \in \mathrm{GL}(m, q)$, $D \in D(n, q)$, $P \in S_n$ such that $G_1 = AG_0DP$. If such matrices exist, compute some choice of A, D, P.*

Remark 1. Note that when D is fixed to be the identity matrix I_n, the induced action reduces to the permutation group S_n acting on the set of equivalence classes of linear codes. This corresponds to the *Permutation Code Equivalence* (PCE) Problem, which lies outside the scope of this work.

Solving LCE is believed to be hard for both classical and quantum adversaries under the appropriate parameter choices [SS13, BBPS23]. As a result, LCE has been used to construct the digital signature LESS [BMPS20]. Moreover, several variants of LCE have also appeared in the literature to accommodate the specific requirements of advanced cryptographic protocols.

The 2-LCE and *Inverse Linear Code Equivalence* (ILCE) problem are two notable examples. The 2-LCE was first introduced in [BBMP24] in the context of threshold signatures, where the adversary is given two LCE instances sharing the same secret monomial M.

Problem 2 (2-LCE)**.** *Let $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ be four $[n, m]_q$ linear codes with generator matrices $G_0, G_1, G_2, G_3 \in \mathbb{F}_q^{m \times n}$ such that $G_1 = AG_0M$ and $G_3 = A'G_2M$ for some matrices $A, A' \in \mathrm{GL}(m, q)$ and $M \in \mathsf{Mono}(n, q)$. The* 2-LCE *problem is asked to compute such M.*

The Inverse Linear Code Equivalence problem can be viewed as a special case of 2-LCE where the two samples are even more strongly related. ILCE was first introduced in [BBN+22] in the context of ring signatures.

Problem 3 (Inverse Linear Code Equivalence (ILCE))**.** *Let $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ be three linear $[n, m]_q$ codes with generator matrices $G_0, G_1, G_2 \in \mathbb{F}_q^{m \times n}$ such that $G_1 = AG_0M, G_2 = A^{-1}G_0M^{-1}$ for some matrices $A \in \mathrm{GL}(m, q)$ and $M \in \mathsf{Mono}(n, q)$. The* ILCE *problem is asked to compute such M.*

The ILCE problem, as well as the 2-LCE, have been shown by [BCD+24] to be solvable in polynomial time when $m = n/2$. The *Diagonal-masked Inverse Linear Code Equivalence* problem was introduced in Versions 1 and 2 of [DKQ+25] in the context of blind signatures. This problem corresponds to a different group action as well as a different equivalence relation for linear codes. Namely, given a linear code $\mathcal{C}$ represented by its generator matrix G, then the equivalence class of G is now defined as $[G]_D := \{AGD : A \in \mathrm{GL}(m, q), D \in D(n, q)\}$, and we study the action of the permutation group S_n on the set of linear codes under this new equivalence relation.

Problem 4 (Diagonal-masked Inverse Linear Code Equivalence (DmILCE))**.** *Given $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ three linear $[n, m]_q$ codes with generator matrices $G_0, G_1, G_2 \in \mathbb{F}_q^{m \times n}$, and $A_0, A_1 \in \mathrm{GL}(m, q), D_0, D_1 \in D(n, q), P \in S_n$ such that $G_1 = A_0G_0D_0P, G_2 = A_1G_1D_1P^{-1}$. The* DmILCE *problem is asked to compute such P.*

In Section 3 we will show that DmILCE is actually polynomial time equivalent to LCE.

2.1.2 Matrix codes

As with linear codes, the central hard problem associated with matrix codes is to determine whether two codes are equivalent under the action of a specified group with respect to a given equivalence relation. In this case, we consider the action of $\mathrm{GL}(m,q) \times \mathrm{GL}(n,q)$ on the set of matrix codes under the standard notion of linear equivalence, although the involved parameters differ from the linear code setting. This setting is formalized by the *Matrix Code Equivalence* problem defined below.

In what follows, $A^T \otimes B^T$ denotes the Kronecker (tensor) product of the matrices $A^T \in \mathbb{F}_q^{m\times m}, B^T \in \mathbb{F}_q^{n\times n}$ which is itself a matrix in $\mathbb{F}_q^{mn\times mn}$.

Problem 5 (Matrix Code Equivalence (MCE))**.** *Given two matrix codes, $\mathcal{C}_0, \mathcal{C}_1$, where G_0 is the generator matrix of $\mathcal{C}_0$ and G_1 that of $\mathcal{C}_1$, determine if there exists $A \in \mathrm{GL}(m,q), B \in \mathrm{GL}(n,q)$ such that $\mathcal{C}_1 = A\mathcal{C}_0B^T$. In another words, such that $G_1 = SG_0(A^T \otimes B^T)$ for some matrix $S \in \mathrm{GL}(k,q)$. If so, compute A, B.*[2]

This problem is at the basis of the signature scheme MEDS [CNP+23b].

Note that, a $[m \times n, k]$ matrix code $\mathcal{C}$ can be viewed as a k-dimensional subspace of $\mathbb{F}_q^{m\times n}$, generated by a basis $(C_1, \ldots, C_k)$, where each $C_s \in \mathbb{F}_q^{m\times n}$. Let $\mathcal{C}_{ij}^{(s)}$ denote the (i,j)-entry of C_s. Then C naturally induces a trilinear form $\mathcal{C}(x,y,z) = \sum_{i,j,s=1}^{m,n,k} C_{ij}^{(s)} x_i y_j z_s$, where $x \in \mathbb{F}_q^m, y \in \mathbb{F}_q^n, z \in \mathbb{F}_q^k$. Intuitively, this form encodes the interaction between the row space, column space, and basis index of the code. Under this representation, two matrix codes $\mathcal{C}_0$ and $\mathcal{C}_1$ are equivalent if and only if their associated trilinear forms satisfy

$$\mathcal{C}_1(x,y,z) = \mathcal{C}_0(Ax, By, Sz)$$

for A, B, S as above.

The *Inverse Matrix Code Equivalence* Problem (IMCE), also introduced in [CNP+23b], is the inverse problem that corresponds to MCE.

Problem 6 (Inverse Matrix Code Equivalence (IMCE))**.** *Given three matrix codes, $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$, such that $\mathcal{C}_1 = A\mathcal{C}_0B^T$ and $\mathcal{C}_2 = A^{-1}\mathcal{C}_0B^{-T}$, for $A \in \mathrm{GL}(m,q), B \in \mathrm{GL}(n,q)$, find A, B up to equivalence.*

In [BCD+24], Budroni et al. give an upper bound on the number of samples (i.e. fresh instances of the problem that share the same secret) IMCE (and MCE) can publish using the same secret. We will summarize this result later on in Section 4. The single sample version of IMCE remains unaffected by their attack.

The *Modified Inverse Matrix Code Equivalence* Problem (MIMCE) focuses on (anti)symmetric matrices and was introduced in [KLP25] to construct a post-quantum blind signature.

Problem 7 (Modified Inverse Matrix Code Equivalence (MIMCE))**.** *Given three matrix codes, $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$, such that $\mathcal{C}_1 = A\mathcal{C}_0B^T$ and $\mathcal{C}_2 = A^{-1}\mathcal{C}_0B^{-T}$, for $A \in \mathrm{GL}(m,q), B \in \mathrm{GL}(n,q)$ (anti)symmetric; find D, F (anti)symmetric such that $\mathcal{C}_2 = D\mathcal{C}_1F^T$.*

This problem looks very similar to IMCE under the condition that A, B are (anti)symmetric. The outputs, however, are slightly different. We formalize IMCE with (anti)symmetric keys in Problem 8.

Problem 8 (symIMCE)**.** *Given three matrix codes, $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$, such that $\mathcal{C}_1 = A\mathcal{C}_0B^T$ and $\mathcal{C}_2 = A^{-1}\mathcal{C}_0B^{-T}$, for $A \in \mathrm{GL}(m,q), B \in \mathrm{GL}(n,q)$ (anti)symmetric, find A, B up to equivalence.*

[2] A reader familiar with the MCE problem might expect the notation $G_1 = SG_0(A \otimes B)$, we follow instead the exposition of [KLP25] which is equivalent up to renaming of the matrices.

It was pointed out in [KLP25, Remark 2.21] that MIMCE and symIMCE should be equivalent, assuming the codes have a trivial automorphism group, i.e. are *rigid*.

Definition 4. A code $\mathcal{C}$ with automorphism group $Aut(\mathcal{C}) = \{\alpha I_m : \alpha \in \mathbb{F}_q\} \times \{\beta I_n : \beta \in \mathbb{F}_q\}$ is called *rigid*.

In the following lemma we formally prove the equivalence between MIMCE and symIMCE for rigid codes. This equivalence will be pertinent to our security analysis in Section 4.

Lemma 1. MIMCE *reduces to* symIMCE *in polynomial time. Assuming the relevant codes are rigid, then* symIMCE *reduces to* MIMCE *in polynomial time.*

Proof. (symIMCE **solves** MIMCE) Consider an adversary $\mathcal{A}$ against MIMCE that is given a MIMCE instance $(\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2)$, then this also forms a symIMCE instance. $\mathcal{A}$ calls the symIMCE solver on $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ to recover some matrices $\hat{A}, \hat{B}$. It then defines $D = \hat{A}^{-2}, F = \hat{B}^{-2}$ (which are symmetric since $\hat{A}, \hat{B}$ are) and returns D, F. Suppose that the symIMCE solver returns a correct output $\hat{A}, \hat{B}$, i.e. we have $\mathcal{C}_1 = \hat{A}\mathcal{C}_0\hat{B}$ and $\mathcal{C}_2 = \hat{A}^{-1}\mathcal{C}_2\hat{B}^{-1}$. Then $D\mathcal{C}_1F^T = D\mathcal{C}_1F = \hat{A}^{-2}\mathcal{C}_1\hat{B}^{-2} = \hat{A}^{-1}\mathcal{C}_0\hat{B}^{-1} = \mathcal{C}_2$, hence D, F is a correct solution to the MIMCE problem.

(MIMCE **solves** symIMCE) Consider an adversary $\mathcal{A}$ against symIMCE that is given a symIMCE instance $(\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2)$, then we show how to solve symIMCE given access to a MIMCE solver. Let $\mathcal{A}$ call the MIMCE solver on $(\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2)$ and obtain some (anti)symmetric matrices D, F such that $D^{-1}\mathcal{C}_2 = \mathcal{C}_1F$. Notice that in the symIMCE instance, substituting one equation into the other, we have that the secret A, B should be such that $A^2\mathcal{C}_2 = \mathcal{C}_1(B^{-1})^2$. In particular, this means that $\mathcal{C}_2 = D\mathcal{C}_1F$ but also $\mathcal{C}_2 = A^{-2}\mathcal{C}_1B^{-2}$. Hence we get that $D\mathcal{C}_1F = A^{-2}\mathcal{C}_1B^{-2}$, so $A^2D\mathcal{C}_1FB^2 = \mathcal{C}_1$. So the pair (A^2D, FB^2) is in the automorphism group of $\mathcal{C}_1$. Since we assumed the codes are rigid, this means that $A^2 = D^{-1}$ and $B^2 = F^{-1}$ up to multiplication by a scalar. To solve the symIMCE instance, it remains to compute two (anti)symmetric matrices A, B such that $(A^2, B^2) = (D^{-1}, F^{-1})$.

We start by computing A, a square root of D^{-1}. Following the approach from [BFP15, Sect. 6.3], we can first write D^{-1} as $D^{-1} = TJT^{-1}$, where J is in Jordan normal form and $T \in GL(n, q)$. Note that the diagonal entries of J will be the eigenvalues of D^{-1}, and thus it may only be defined over a field extension, of degree $O(n)$. Then the candidates for A will take the form $TJ^{1/2}T^{-1}$ (again, here we may need to double the field extension so that the values of $J^{1/2}$ are defined). To compute the matrix $J^{1/2}$, we first consider how to compute the square root of one Jordan block of the form

$$J_i = \begin{bmatrix} \lambda_i & 1 & \cdots & 0 \\ 0 & \lambda_i & \cdots & 0 \\ & & & \\ 0 & 0 & \ddots & 1 \\ 0 & 0 & \cdots & \lambda_i \end{bmatrix}.$$

The square root of this block will be an upper triangular matrix whose diagonal entries are the (same) square root of λ_i. Note that if any of the the square roots were to vary by sign, at least one of the entries at position $(i, i+1)$ would be a zero instead of a one, as required by the Jordan normal form. The remaining upper triangular values are then well-defined by this choice of square root. An explicit formula for these remaining values is given in [BFP15, Eq. 9]. Thus, every block has two choices of square root, and J has 2^k square roots, where k is the number of distinct eigenvalues (i.e. k is the number of blocks in J).

Among these square roots, we want to select one correct $J^{1/2}$ such that $A = TJ^{1/2}T^{-1}$ is (anti)symmetric. We first compute one possible solution for $J^{1/2}$. From here, we can

parameterize A using at most n variables $s_i \in \{\pm 1\}$ which account for the possible changes of sign in the blocks of $J^{1/2}$. In total, this creates a system of $n + \frac{n(n+1)}{2}$ variables and n^2 equations . Thus we may solve for A linearly. Then, with A, we may return to the original symIMCE instance and solve linearly for B. □

Remark 2. Note that [RST24] suggests, backed by experimental evidence and analysis, that a randomly chosen code is, in fact, rigid.

3 Reduction between LCE and DmILCE

The work of [DKQ+25] introduces a general framework for blind signatures from cryptographic group actions. They instantiate their framework using code-based group actions. In Versions 1 and 2 of their work, they presented the DmILCE problem (Problem 4).

In the following, we analyze this new assumption and show that it is in fact polynomial-time equivalent to the LCE problem, a well-studied code-based assumption. We recall further reductions from LCE known in the literature to consolidate the confidence in this assumption.

We will make use of the following lemma.

Lemma 2. *Let $D \in D(n,q)$ and $P \in S_n$ then $PDP^{-1} \in D(n,q)$.*

Proof. Let $\{d_{i,i}\}_{i=1}^n$ denote the diagonal entries of D. Then left multiplication by a permutation matrix $P \in S_n$ will permute the rows of D by some permutation function π, thus it sends $d_{i,i}$ to $d_{\pi(i),i}$. On the other hand, right multiplication by the same permutation matrix P permutes the columns of D by π^{-1}. Thus it sends $d_{i,i}$ to $d_{i,\pi^{-1}(i)}$.
Therefore left multiplication by P and right multiplication by P^{-1} sends $d_{i,i}$ to $d_{\pi(i),(\pi^{-1})^{-1}(i)} = d_{\pi(i),\pi(i)}$. This means that every diagonal entry in D remains on the diagonal. The other entries of D are identically zero, hence permuting them will not change the shape of the resulting matrix. Therefore, $PDP^{-1} \in D(n,q)$. □

We now give polynomial-time algorithms that can solve LCE given access to a DmILCE solver, and vice versa.

Theorem 1. LCE *and* DmILCE *are polynomial-time equivalent.*

Proof. (LCE **solves** DmILCE**:**)

Given the DmILCE instance (G_0, G_1, G_2), observe that the pair (G_0, G_1) forms an LCE instance. Thus we can use the LCE solver to recover $Q = D_0 P$ such that $G_1 = A_0 G_0 D_0 P$ for $A_0 \in GL(m,q), D_0 \in D(n,q)$ and a permutation matrix $P \in S_n$. With Q, we can easily read off P. This trivially solves DmILCE.

(DmILCE **solves** LCE**:**)

Let (G_0, G_1) be an instance of LCE. Thus $G_1 = A_0 G_0 D_0 P$ for some $A_0 \in GL(m,q), D_0 \in D(n,q)$ and a permutation matrix $P \in S_n$. We are asked to recover all of A_0, D_0, P.

Consider an adversary $\mathcal{B}$ against DmILCE. We want to build an adversary $\mathcal{A}$ that can solve LCE given access to $\mathcal{B}$.

The key insight of our reduction consists in observing that in the DmILCE instance, the codes corresponding to G_0 and G_2 are in the same equivalence class, i.e. $[G_0]_D = [G_2]_D$. Indeed, recall that $[G_0]_D := \{AG_0D : A \in GL(m,q), D \in D(n,q)\}$. We have that

$$\begin{aligned} G_2 &= A_1 G_1 D_1 P^{-1} \\ &= A_1 (A_0 G_0 D_0 P) D_1 P^{-1} \\ &= (A_1 A_0) G_0 (D_0 P D_1 P^{-1}). \end{aligned}$$

Note that $(A_1A_0) \in GL(m,q)$. Further, by Lemma 2 the matrix PD_1P^{-1} is diagonal, and thus so is $D_0PD_1P^{-1}$. This shows that $G_2 \in [G_0]_D$.

Using this insight, let us now describe the corresponding adversary, which is summarized in Algorithm 1.

Algorithm 1: $\mathcal{A}(G_0, G_1)$

$D \xleftarrow{\$} D(n,q)$
$A \xleftarrow{\$} GL(n,q)$
$G_2 \leftarrow AG_0D$
$\mathcal{B}(G_0, G_1, G_2) \to P$
Solve $G_0D_0PH_1^T = 0$ to recover some D_0
Solve $G_1 = A_0G_0D_0P$ to recover A_0
Return A_0, D_0, P

Given G_0, G_1 of rank m, $\mathcal{A}$ will start by selecting a random $D' \in D(n,q)$ and $A' \in GL(m,q)$. Then it computes G_2 in the following way:

$$\begin{aligned} G_2 &:= A'G_0D' \\ &= A'(A_0G_1P^{-1}D_0^{-1})D' \end{aligned}$$

Let $A_1 := A'A_0 \in GL(m,q)$. Since A' is sampled from a uniformly random distribution, then A_1 will be randomly distributed as well. Further, note that G_2 will always have rank m since all of A', G_0, and D' will have full rank by definition.

Define $D_1 := P^{-1}D_0^{-1}D'P$. Then, by Lemma 2, D_1 is diagonal and we can rewrite our equation as

$$G_2 = A_1G_1D_1P^{-1}.$$

As before, since D' is sampled uniformly within the set of $n \times n$ diagonal matrices over $\mathbb{F}_q$, it follows that D_1 will also be uniformly random within that same set. Hence the G_2 built here follows the same distribution as a DmILCE instance.

It follows that the tuple (G_0, G_1, G_2) constitutes a full instance of DmILCE. After passing it to the DmILCE solver $\mathcal{B}$, we are provided P.

From here, we may recover D_0 using the parity check matrix of G_1, denote it H_1. Observe that

$$\begin{aligned} & G_1H_1^T = 0 \\ \iff & A_0G_0D_0PH_1^T = 0 \\ \iff & G_0D_0PH_1^T = 0 \end{aligned}$$

This gives a system of mn linear equations and only n variables, hence we can recover D_0 using Gaussian elimination in $O(n^3)$ elementary operations.

From here, we can recover A_0 by solving the linear system $G_1 = A_0G_0D_0P$. This computation also requires $O(n^3)$ elementary operations.

Suppose that the adversary $\mathcal{B}$ solves the DmILCE problem in polynomial time, then by construction the adversary $\mathcal{A}$ will solve the LCE problem in polynomial time too. □

This reduction gives theoretical support for the hardness of DmILCE. Additionally, in [BW25, Corr. 5.2] Bennett and Win give a polynomial-time reduction from LCE (over a field whose order is polynomially-bounded) to the Lattice Isomorphism Problem (LIP). Therefore, [BW25] together with Theorem 1, proves the existence of a reduction from DmILCE to two well-studied problems: LCE itself and LIP.

Remark 3. Note that the 2-sample version of LCE is broken in polynomial time by the work of [BCD+24]. Thus our equivalence result suggests against using several-sample versions of DmILCE too.

4 Solving 2-(M)IMCE

We now consider a variant of (M)IMCE (Problem 7), where an adversary has access to more than one instance using the same secret matrices A, B. Such a formulation would be of interest in the context of multi-party protocols, such as threshold or ring signatures. First, we establish the state-of-the-art for this problem using the findings from Budroni et al. [BCD+24]. Later, we present our attack and show that actually, in the case of MIMCE, two samples are sufficient to recover A, B in polynomial time.

4.1 t-symIMCE

We follow the analysis of [BCD+24] to study the MIMCE variants with more samples, i.e. we look at the variants of Problems 5 and 6 when the matrices involved are symmetric and two samples are provided.

In Corollaries 1 and 2 of [BCD+24], the authors give an estimate on the number of samples needed to solve MCE, respectively IMCE, with non negligible probability in time $O((mn)^{2\omega})$, where ω is the linear algebra constant. We follow their modeling to deduce a similar bound in the context of the blind signature from [KLP25] whose hardness relies on MIMCE.

The IMCE problem can be expressed using the generator matrices of the underlying code. Given G_0, a generator matrix of $\mathcal{C}_0$, we get that $G_1 = SG_0Q$ is the generator matrix of $\mathcal{C}_1$, with $Q = A^T \otimes B^T$ in $GL(mn, q)$.

One can then use the equation that stems from [BCD+24, Prop. 2] to write

$$G_1 H_1^T = 0 \iff SG_0QH_1^T = 0 \iff G_0QH_1^T = 0 \iff (G_0 \otimes H_1^T)vec(Q) = 0$$

[BCD+24, Prop. 2] shows that this expression gives $k(mn-k)$ linear equations in $(mn)^2$ variables (the entries of $Q = A^T \otimes B^T$). However, in our case A, B are symmetric. That is, $A = A^T$ and $B = B^T$ so in particular $Q = A^T \otimes B^T = A \otimes B$ is also symmetric by the properties of the Kronecker product ($(A \otimes B)^T = (A^T \otimes B^T)$). So we can instead model Q using $mn(mn+1)/2$ variables. By following their reasoning, we can adapt their results to obtain the following two corollaries.

Corollary 1. *For $t \geq \lfloor \frac{mn(mn+1)/2}{k(mn-k)} \rfloor + 1$ samples, the t-symMCE problem is solvable with non-negligible probability in time $O((mn)^{2\omega})$.*

Corollary 2. *For $t \geq \lfloor \frac{mn(mn+1)/2}{2k(mn-k)} \rfloor + 1$ samples, the t-symIMCE problem is solvable with non-negligible probability in time $O((mn)^{2\omega})$.*

Following the NIST Level 1 MEDS parameters, we fix $n = m = k$. We compute in Table 2 the number of samples t needed according to the results of [BCD+24] and above.

Table 2: Parameters from MEDS spec and t-sample estimates necessary to recover the secret, derived from [BCD+24].

NIST level	n	t_{MCE}	t_{IMCE}	t_{symMCE}	t_{symIMCE}
1	14	15	8	8	4
3	22	23	12	12	6
5	30	31	16	16	8

This shows that the symmetric version of the MCE and IMCE does not reach the same security level against the t-sample attack as the original version. In fact, the number of samples needed to run the attack is divided by two for all parameter sets.

In [KLP25], they suggest increasing the MEDS parameters by a factor $\sqrt{2}$ to ensure security against the basic algebraic brute-force attack. In Table 3, we report the same metrics for these new parameters. It shows that this increase would not be enough to preserve the same security level against these t-sample attacks.

Table 3: Increased MEDS parameters as suggested by [KLP25] and corresponding t-sample estimates, derived from [BCD+24].

NIST level	n	t_{symMCE}	t_{symIMCE}
1	20	11	6
3	31	17	9
5	42	22	11

In the following section, we improve upon the results derived from [BCD+24] by recovering the secret inputs using only 2 samples instead of 6-11 samples, regardless of the dimension of the underlying matrix code.

4.2 Attack on 2-(M)IMCE

We state below the 2 sample version of the symIMCE problem in terms of generator matrices. Recall from Lemma 1, that this is equivalent to the 2 sample MIMCE problem.

Problem 9. *[2-symIMCE] Suppose we are given two tuples of three codes C_0, C_1, C_2 and C'_0, C'_1, C'_2 with associated generator matrices G_0, G_1, G_2 and F_0, F_1, F_2 forming two* symIMCE *instances.*
They verify

$$\begin{cases} G_1 = S_0 G_0 (A \otimes B) \\ G_2 = S_1 G_0 (A^{-1} \otimes B^{-1}) \\ F_1 = S_2 F_0 (A \otimes B) \\ F_2 = S_3 F_0 (A^{-1} \otimes B^{-1}) \end{cases} \tag{1}$$

with $A \in GL(m, q), B \in GL(n, q)$ (anti)symmetric and $S_i \in GL(k, q), i = 0, .., 3$. Recover A, B (up to equivalence).

Recall that applying the results from [BCD+24] would require extra samples in order to recover the secret inputs to 2-symIMCE. We show below that the 2 sample instance can in fact already be solved in polynomial time.

Theorem 2. *The 2-*symIMCE *problem can be solved in $O(k^{2\omega} + (m^2 n)^\omega + n^{2\omega})$ operations. When $n = m = k$, this gives a dominating complexity of $O(n^{3\omega})$ operations.*

Proof. In what follows we outline an attack on the 2-symIMCE problem when the secret matrices are symmetric. Note that the antisymmetric case follows analogously.

Recovering S_0. The first step consists in enumerating the possibilities for the isomorphism S_0. We will make use of the symmetry of the underlying matrices. Observe that $G_1 G_2^T = S_0 G_0 (A \otimes B)(A \otimes B)^{-T} G_0^T S_1^T$. Since A, B are symmetric, so is $A \otimes B$, we have that $(A \otimes B)^{-T} = (A \otimes B)^{-1}$. Thus the equation does not involve A or B, and we obtain $G_1 G_2^T = S_0 G_0 G_0^T S_1^T$. We can repeat this for our second instance, as well as for cross terms

to obtain the following set of equations.

$$\begin{cases} G_1 G_2^T S_1^{-T} = S_0 G_0 G_0^T \\ F_1 F_2^T S_3^{-T} = S_2 F_0 F_0^T \\ G_1 F_2^T S_3^{-T} = S_0 G_0 F_0^T \\ F_1 G_2^T S_1^{-T} = S_2 F_0 G_0^T \end{cases} \tag{2}$$

This gives us a system of $4k^2$ linear equations in $4k^2$ variables (the S_i matrices). However, solving naively gives a large number of solutions due to redundancies in the system. We provide an explicit description of the solution space of (2) in the following lemma.

Lemma 3. *Let $V := (G_0 G_0^T)(F_0 G_0^T)^{-1}(F_0 F_0^T)(G_0 F_0^T)^{-1}$. Assume that the minimal polynomial of V has degree n (i.e., it coincides with the characteristic polynomial of V). Let S_0' be a solution for S_0 in System (2). Then any other solution for S_0 in the system (2) will be of the form $S_0' \mathsf{P}_i$ where $\mathsf{P}_i := \sum_{j=0}^{k-1} \lambda_j^i V^j$ for some $\lambda_j^i \in \mathbb{F}_q$.*

Proof. Let $(S_i')_{i=0}^3$ be a fixed solution for (2), and let $(S_i)_{i=0}^3$ be another arbitrary solution. Since both solutions verify (2), we have

$$S_0 G_0 G_0^T S_1^T = G_1 G_2^T = S_0' G_0 G_0^T S_1'^T$$

In particular, this gives us

$$S_0^{-1} S_0' G_0 G_0^T S_1'^T S_1^{-T} = G_0 G_0^T$$

The other equations of the system in 2 follow similarly. Let us denote $P_i = S_i^{-1} S_i'$, then we obtain the following system.

$$\begin{cases} P_0^{-1} G_0 G_0^T P_1 & = G_0 G_0^T \\ P_2^{-1} F_0 F_0^T P_3 & = F_0 F_0^T \\ P_0^{-1} G_0 F_0^T P_3 & = G_0 F_0^T \\ P_2^{-1} F_0 G_0^T P_1 & = F_0 G_0^T \end{cases}$$

We would like to characterize the matrices P_i appearing in this system.

If we express P_1, P_2, P_3 in terms of P_0, we obtain the following equations:

$$\begin{cases} P_1 & = (G_0 G_0^T)^{-1} P_0 (G_0 G_0^T) \\ P_3 & = (F_0 F_0^T)^{-1} P_2 (F_0 F_0)^T \\ P_3 & = (G_0 F_0^T)^{-1} P_0 (G_0 F_0^T) \\ P_2 & = (F_0 G_0^T) P_1 (F_0 G_0^T)^{-1} \end{cases}$$

Replacing the expression for P_1 in P_2, we get $P_2 = (F_0 G_0^T)(G_0 G_0^T)^{-1} P_0 (G_0 G_0^T)(F_0 G_0^T)^{-1}$ Which we can now plug into the first expression for P_3 and get

$$P_3 = (F_0 F_0^T)^{-1}(F_0 G_0^T)(G_0 G_0^T)^{-1} P_0 (G_0 G_0^T)(F_0 G_0^T)^{-1}(F_0 F_0)^T$$

We now have two expressions for P_3 in terms of P_0 which we can set equal to obtain

$$P_0 = (G_0 F_0^T)(F_0 F_0^T)^{-1}(F_0 G_0^T)(G_0 G_0^T)^{-1} P_0 (G_0 G_0^T)(F_0 G_0^T)^{-1}(F_0 F_0)^T (G_0 F_0^T)^{-1}$$

Then, if we set $V := (G_0 G_0^T)(F_0 G_0^T)^{-1}(F_0 F_0^T)(G_0 F_0^T)^{-1}$, we observe that $P_0 = V^{-1} P_0 V$. In other words, P_0 and V are commuting matrices.

By our assumption that the minimal and characteristic polynomials of V coincide, we conclude that P_0 can be written as a polynomial in V [Cur84, Sect. 25, Exercise 5]. Thus we can write P_0 as a polynomial in V as desired and $S_0' P_0$ will be a solution by construction. □

Remark 4. For a random matrix over a finite field $\mathbb{F}_q$, [NP95] tells us that the minimal and characteristic polynomial will coincide with probability greater than 99% for our choices of q. This probability was corroborated experimentally for actual V matrices of the form described above. In fact, when checking over several hundred runs, we never encountered a failure case. This shows that our assumption on V is not restrictive in general.

Thus, we can solve System (2) while normalizing n entries to obtain a unique solution S_0', requiring $O(k^{2\omega})$ elementary operations, where ω is the linear algebra constant. This will work so long as the normalized entries do not correspond to zero entries in the original matrix S_0 that we are solving for. The probability of this occurring is $1-(\frac{q-1}{q})^n$, i.e. the complement probability to having all the entries be non-zero. In such a case, we can repeat by normalizing a different set of entries.

Lemma 3 allows us to characterize the possible solutions for S_0 based on S_0'.

Recovering A and B. Observe that

$$G_1 = S_0 G_0(A \otimes B) \iff G_1(I_m \otimes B^{-1}) = S_0 G_0(A \otimes I_n). \tag{3}$$

If we plug in the characterization of S_0 deduced from Lemma 3, we get the following:

$$G_1 = S_0'\mathsf{P}_0 G_0(A \otimes B) \iff G_1(I_m \otimes B^{-1}) = S_0'\mathsf{P}_0 G_0(A \otimes I_n). \tag{4}$$

This gives a linear expression for B depending bilinearly on A and P_0.

From here, we would like to recover all of A ($m(m+1)/2$ variables), B ($n(n+1)/2$ variables), and P_0 (k variables). We will use a similar approach to that of Section **??** to construct bilinear equations. First, write G_1 as n blocks of size $k \times m$. This gives

$$G_1 = [[G_1]_1, \cdots, [G_1]_n].$$

Then from Eq. 4 we have

$$G_1(I_m \otimes B^{-1}) = [[G_1]_1 B^{-1}, \ldots, [G_1]_n B^{-1}].$$

Similarly, we can write the rest of Eq. 4 in terms of blocks of the form

$$S_0'\mathsf{P}_0 G_0(A \otimes I_n) = [\mathcal{B}_0(A, \mathsf{P}_0) \cdots \mathcal{B}_n(A, \mathsf{P}_0)],$$

where the $\mathcal{B}_i(A, \mathsf{P}_0)$ are $k \times m$ blocks whose entries are bilinear equations depending on the entries of A and P_0.

Now, for any $i \in [n]$, we have that

$$[G_1]_i^{-1}[G_1]_i B^{-1} = [G_1]_i^{-1}\mathcal{B}_i(A, \mathsf{P}_0).$$

Remark 5. Throughout the attack, we will be decomposing G_0, G_1 into m blocks of size $k \times n$ and n blocks of size $k \times m$. Thus, in order for the notion of invertibility to be well-defined, we require $n = m = k$.

Remark 6. Note that sometimes the G_1^i are not invertible. The probability that any one block M has rank $n-d$ for some integer d goes to q^{-d^2} as q goes to infinity. This result is a direct observation from [FG15]. To ensure invertibility, we are interested in full rank matrices i.e. $d = 0$, hence as q grows, this probability tends to 1. For small values of q this could still be an issue. However, when a block is not full rank, one can add a linear combination of the other blocks to itself to make it invertible.

Now consider all pairs $(1, j)$ for $j \in [2, n]$, and subtract the corresponding equations:

$$0 = B^{-1} - B^{-1} = [G_1]_1^{-1}\mathcal{B}_0(A, \mathsf{P}_0) - [G_1]_j^{-1}\mathcal{B}_j(A, \mathsf{P}_0).$$

Table 4: Average runtime of our attack on 2-MIMCE taken from 10 runs. We use the MEDS parameters but with n scaled by a factor of $\sqrt{2}$, as suggested by [KLP25]. We run the attack using Magma [BCP97] on a laptop.

NIST level	n	q	Thm. 2 runtime
1	20	4093	35.5 s
3	31	4093	25.3 min
5	42	2039	6.58 hr

Any other relations derived in this way would be linear combinations of each other, so we only consider these $n-1$ relations.

In total this gives a system of $n-1$ bilinear matrix equations depending on the entries of A and the k coefficients in P_0. Each of these relations contain $m \cdot k$ homogenous equations. In total, there are $(n-1)mk$ equations, and $\frac{m(m+1)}{2} + k$ variables. Further, since the variables of A are being multiplied by those of P_0, we get $\frac{m(m+1)}{2}k$ monomials in our bilinear system. Experiments verify that these equations are indeed linearly independent from each other. This gives a system that can be efficiently linearized. With this number of monomials, we use direct linearization which should require roughly $O((m^2 n)^\omega)$ elementary operations, where ω is the linear algebra constant.

After having recovered A and P_0, we may solve for B directly using the relation $G_1 = S_0 G_0 (A \otimes B)$. This relation contains a total of kmn linear equations, and $\frac{n(n+1)}{2}$ unknowns. Gaussian elimination will suffice here, which uses $O(n^{2\omega})$ elementary operations. □

Experimental results. The authors of [KLP25] suggested using MEDS [CNP+23b] parameter sets, and scaling them up to a small factor (around $\sqrt{2}$). In particular, this means that $n = m = k$. The original parameters from MEDS propose vector dimensions and field size, (n, q), of $(14, 4093), (22, 4093), (30, 2039)$. We run our attack against the modified MEDS parameters, and report on the timings in Table 4. The code used to obtain these timings can be found at https://github.com/vgilchri/blind-sigs-mimce.

5 Conclusion

Code-based problems have emerged as candidates for post-quantum cryptography, and received particular attention as a new option for a cryptographic group action. While the core LCE and MCE problems have received (and continue to receive) extensive cryptanalysis, variants of these problems have begun to appear in the literature that also require analysis. These variants are often developed with the purpose of achieving some new functionalities and advanced protocols.

In this work, we study two variants of LCE and MCE that were introduced in the context of blind signatures. We show a polynomial-time equivalence between LCE and its DmILCE variant. This provides strong theoretical confidence in the hardness of this new problem, which was presented in a prior version of the blind signature scheme [DKQ+25]. With new confidence in the DmILCE problem, it is an interesting question to see which primitives could be built from it.

The multiple samples problems have been used in the past to build multi-user primitives such as [BBMP24] which builds threshold signatures from 2-LCE. We anticipate such future attempts and analyze the multiple sample version of the MIMCE problem. We show that two samples are enough to break the problem, regardless of the dimensions of the code, suggesting that this problem is not suitable for building multi-user primitives.

References

[BBD+25] Michele Battagliola, Giacomo Borin, Giovanni Di Crescenzo, Alessio Meneghetti, and Edoardo Persichetti. Enhancing threshold group action signature schemes: Adaptive security and scalability improvements. In Ruben Niederhagen and Markku-Juhani O. Saarinen, editors, *Post-Quantum Cryptography - 16th International Workshop, PQCrypto 2025, Part I*, pages 129–161. Springer, Cham, April 2025. doi:10.1007/978-3-031-86599-2_5.

[BBMP24] Michele Battagliola, Giacomo Borin, Alessio Meneghetti, and Edoardo Persichetti. Cutting the GRASS: threshold group action signature schemes. In Elisabeth Oswald, editor, *Topics in Cryptology - CT-RSA 2024 - Cryptographers' Track at the RSA Conference 2024, San Francisco, CA, USA, May 6-9, 2024, Proceedings*, volume 14643 of *Lecture Notes in Computer Science*, pages 460–489. Springer, 2024. doi:10.1007/978-3-031-58868-6_18.

[BBN+22] Alessandro Barenghi, Jean-François Biasse, Tran Ngo, Edoardo Persichetti, and Paolo Santini. Advanced signature functionalities from the code equivalence problem. *International Journal of Computer Mathematics: Computer Systems Theory*, 7(2):112–128, 2022.

[BBPS23] Alessandro Barenghi, Jean-François Biasse, Edoardo Persichetti, and Paolo Santini. On the computational hardness of the code equivalence problem in cryptography, 2023. URL: https://www.aimsciences.org/article/id/62fa202b4cedfd0007b8b288, doi:10.3934/amc.2022064.

[BCD+24] Alessandro Budroni, Jesús-Javier Chi-Domínguez, Giuseppe D'Alconzo, Antonio J. Di Scala, and Mukul Kulkarni. Don't use it twice! Solving relaxed linear equivalence problems. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part VIII*, volume 15491 of *LNCS*, pages 35–65. Springer, Singapore, December 2024. doi:10.1007/978-981-96-0944-4_2.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The magma algebra system i: The user language. *Journal of Symbolic Computation*, 24(3-4):235–265, 1997.

[BFP15] Jérémy Berthomieu, Jean-Charles Faugère, and Ludovic Perret. Polynomial-time algorithms for quadratic isomorphism of polynomials. *J. Complex.*, 31(4):590–616, August 2015. doi:10.1016/j.jco.2015.04.001.

[BMPS20] Jean-François Biasse, Giacomo Micheli, Edoardo Persichetti, and Paolo Santini. LESS is more: Code-based signatures without syndromes. In Abderrahmane Nitaj and Amr M. Youssef, editors, *AFRICACRYPT 20*, volume 12174 of *LNCS*, pages 45–65. Springer, Cham, July 2020. doi:10.1007/978-3-030-51938-4_3.

[BW25] Huck Bennett and Kaung Myat Htay Win. Relating code equivalence to other isomorphism problems. *Des. Codes Cryptogr.*, 93(3):701–723, 2025. URL: https://doi.org/10.1007/s10623-024-01542-3, doi:10.1007/S10623-024-01542-3.

[CNP+23a] Tung Chou, Ruben Niederhagen, Edoardo Persichetti, Lars Ran, Tovohery Hajatiana Randrianarisoa, Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Matrix equivalence digital signature, 2023. URL: https://www.meds-pqc.org/spec/MEDS-2023-07-26.pdf.

[CNP+23b] Tung Chou, Ruben Niederhagen, Edoardo Persichetti, Tovohery Hajatiana Randrianarisoa, Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Take your MEDS: Digital signatures from matrix code equivalence. In Nadia El Mrabet, Luca De Feo, and Sylvain Duquesne, editors, *AFRICACRYPT 23*, volume 14064 of *LNCS*, pages 28–52. Springer, Cham, July 2023. doi:10.1007/978-3-031-37679-5_2.

[Cur84] Charles W. Curtis. *Linear Algebra - An Introductory Approach*. Springer, 1984.

[DFG23] Giuseppe D'Alconzo, Andrea Flamini, and Andrea Gangemi. Non-interactive commitment from non-transitive group actions. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 222–252. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8739-9_8.

[DFMS24] Giuseppe D'Alconzo, Andrea Flamini, Alessio Meneghetti, and Edoardo Signorini. A framework for group action-based multi-signatures and applications to less, meds, and ALTEQ. *IACR Cryptol. ePrint Arch.*, page 1691, 2024. URL: https://eprint.iacr.org/2024/1691.

[DKQ+25] Dung Hoang Duong, Xuan Thanh Khuc, Youming Qiao, Willy Susilo, and Chuanqi Zhang. Blind signatures from cryptographic group actions. Cryptology ePrint Archive, Report 2025/397, 2025. URL: https://eprint.iacr.org/2025/397.

[FG15] Jason Fulman and Larry Goldstein. Stein's method and the rank distribution of random matrices over finite fields. *The Annals of Probability*, 43(3), May 2015. doi:10.1214/13-aop889.

[GMPT24] Valerie Gilchrist, Laurane Marco, Christophe Petit, and Gang Tang. Solving the tensor isomorphism problem for special orbits with low rank points: Cryptanalysis and repair of an asiacrypt 2023 commitment scheme. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 141–173. Springer, Cham, August 2024. doi:10.1007/978-3-031-68376-3_5.

[JWL+25] Kaijie Jiang, Anyu Wang, Hengyi Luo, Guoxiao Liu, Tang Gang, Yanbin Pan, and Xiaoyun Wang. Re-randomize and extract: A novel commitment construction framework based on group actions. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part II*, volume 15602 of *Lecture Notes in Computer Science*, pages 124–153. Springer, 2025. doi:10.1007/978-3-031-91124-8_5.

[KLP25] Veronika Kuchta, Jason T. LeGrow, and Edoardo Persichetti. Post-quantum blind signatures from matrix code equivalence. Cryptology ePrint Archive, Paper 2025/274, 2025. URL: https://eprint.iacr.org/2025/274.

[NP95] Peter M Neumann and Cheryl E Praeger. Cyclic matrices over finite fields. *Journal of the London Mathematical Society*, 52(2):263–284, 1995.

[RST24] Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Hardness estimates of the code equivalence problem in the rank metric. *Designs, Codes and Cryptography*, 92:1–30, 01 2024. doi:10.1007/s10623-023-01338-x.

[SS13] Nicolas Sendrier and Dimitris E. Simos. The hardness of code equivalence over and its application to code-based cryptography. In Philippe Gaborit, editor, *Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013*, pages 203–216. Springer, Berlin, Heidelberg, June 2013. doi:10.1007/978-3-642-38616-9_14.

5 The Tensor Isomorphism Problem for special orbits

Publication data. Valerie Gilchrist, Laurane Marco, Christophe Petit, Gang Tang. Solving the Tensor Isomorphism Problem for special orbits with low rank points: Cryptanalysis and repair of an Asiacrypt 2023 commitment scheme. CRYPTO 2024.

My Contribution. In this work all authors participated in the theoretical development. Myself and L. Marco split between us the majority of writing, coding, and experiments.

Solving the Tensor Isomorphism Problem for special orbits with low rank points: Cryptanalysis and repair of an Asiacrypt 2023 commitment scheme

Valerie Gilchrist[1], Laurane Marco[2], Christophe Petit[13], Gang Tang[43]

[1] Université Libre de Bruxelles, Brussels, Belgium
[2] EPFL, Lausanne, Switzerland
[3] University of Birmingham, Birmingham, United Kingdom
[4] University of Technology Sydney, NSW, Australia

Abstract. The Tensor Isomorphism Problem (TIP) has been shown equivalent to the matrix code equivalence problem, making it an interesting candidate on which to build post-quantum cryptographic primitives. These hard problems have already been used in protocol development. One of these, MEDS, is currently in Round 1 of NIST's call for additional post-quantum digital signatures.

In this work, we consider the TIP restricted to the orbits of a special class of tensors. The hardness of the decisional version of this problem is the foundation of a commitment scheme proposed by D'Alconzo, Flamini, and Gangemi (Asiacrypt 2023). We present polynomial-time algorithms for the decisional and computational versions of TIP for special orbits, which implies that the commitment scheme is not secure. The key observations of these algorithms are that these special tensors contain some low-rank points, and their stabilizer groups are not trivial.

With these new developments in the security of TIP in mind, we give a new commitment scheme based on the general TIP that is non-interactive, post-quantum, and statistically binding, making no new assumptions. Such a commitment scheme does not currently exist in the literature.

1 Introduction

Group actions have proven very useful in the transition to post-quantum cryptography. They provide a quantum-safe algebraic operation in certain instantiations, making it easy to use them as a replacement in previously classical schemes. Recently, isogenies have been the most well-studied example of a post-quantum

* Authors listed in alphabetical order: see https://www.ams.org/profession/leaders/CultureStatement04.pdf. Valerie Gilchrist is supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium; Christophe Petit is partly supported by EPSRC through grant number EP/V011324/1.

Date of this document: 2025-03-18.

group action [8, 13]. We can find, however, group actions from other hard problems, some of which have already been proposed for use in post-quantum cryptography. Such group actions include alternating trilinear forms [48], lattices [21], polynomials [41], linear codes [5, 16], and tensors [32].

Grochow and Qiao [27] defined the tensor isomorphism class TI, that is, if a problem is polynomially equivalent to tensor isomorphism, it is TI-complete. In particular, the alternating trilinear form equivalence problem is TI-complete [28]. Based on this problem, a signature scheme ALTEQ [9] has recently been submitted to NIST as a round 1 candidate. The work of [27] shows that three problems, the matrix code equivalence problem, the trilinear form equivalence problem, and the tensor isomorphism problem (TIP) are all equivalent. Particularly given a 3-tensor with dimensions l, m, n, we can naturally represent it as a $[l \times m, n]$-matrix code. Thus, though our focus in this work will be on TIP, we will be inherently studying the security of all three of these problems. Two recent independent works [16, 32] proposed using tensor isomorphism (or matrix code equivalence) for cryptographic purposes from the different algebraic structures of tensors and codes. Very recently, a digital signature scheme, MEDS [15], based on matrix code equivalence was submitted to the NIST standardization competition for post-quantum signatures as a round 1 candidate. The security of MEDS, that is, the algorithm of matrix code equivalence, has been studied in several works [15, 16, 18, 39, 44]. We proceed to briefly review the state-of-the-art algorithms for matrix code equivalence, and hence TIP.

Algorithms solving matrix code equivalence. There are three types of algorithms for matrix code equivalence. The first one use algebraic methods, whereby matrix code equivalence is translated into solving systems of polynomial equations. Gröbner basis techniques are often employed for this approach, albeit with efficiency contingent upon the values of l, m and n. Notably, efficiency diminishes significantly when these dimensions become slightly large. The second is the graph-theoretic algorithm [44], which involves transforming matrix codes into quadratic mappings, thereby facilitating the adaptation of algorithms designed for Quadratic Maps Linear Equivalence (QMLE) [10]. Nevertheless, this approach does not perform well when $l = m = n$. The last one is the Leon-like algorithm, which is an adaptation of the code equivalence problem algorithm on the Hamming metric. The basic idea comes from the observation that equivalence preserves the Hamming weight as well as the weight distribution of the codewords. Leon's [35] algorithm entails finding the set of codewords with minimal Hamming weight in two given codes, which can reveal enough information to recover the equivalence. Beullens [7] recently improved this algorithm by building two lists of codewords with a particular weight and then searching for collisions between them to recover the equivalence. This approach naturally translated to matrix code [16]: by creating two lists of matrices with low rank in the two given matrix codes one can then find the collision to recover the equivalence. Very recently, Narayanan, Qiao and Tang [39] further improved Beullens' algorithm and introduced a new invariant called "co-rank1 associated invariant" that avoids finding collisions by Gröbner basis.

In this work, we will consider TIP, but on special orbits. This problem was first introduced at ASIACRYPT 2023 by D'Alconzo, Flamini and Gangemi [20] in the context of a commitment scheme construction. While the general TIP considers tensors that are generated randomly, which we call *random tensors*, the work of [20] restricts to the use of very particular tensors with a lot of structure, which we will call *unit tensors*. Their commitment scheme uses the orbits of two of these unit tensors, which they conjecture is still secure.

1.1 Our contributions

In this paper, we propose polynomial-time algorithms for the decisional and computational versions of TIP on special orbits. This implies that the commitment scheme proposed by D'Alconzo, Flamini and Gangemi in Asiacrypt 2023 is broken. We also propose a countermeasure to fix the commitment scheme. We summarize our contributions in what follows.

Algorithms for tensor isomorphism on special orbits. We present a polynomial-time algorithm for the TIP on special orbits. One observation is that some low-rank points exist for the tensors on these special orbits. This allows us to distinguish tensors on two different orbits, which implies that the decisional TIP is broken. Another observation is that the automorphism groups of these unit tensors are not trivial, that is, they have stabilizers. Therefore, we can avoid finding collisions and quickly recover an equivalent isomorphism which solves the computational TIP.

Regarding the commitment scheme proposed by D'Alconzo, Flamini and Gangemi, its security is based on the hardness of decisional TIP on these unit tensors. We implement our algorithms and use them to break the hiding property of the scheme, as well as recover the random values used when creating the commitment. These algorithms render the scheme completely insecure; see more details in Section 3.

Repairing Asiacrypt 2023 scheme. We provide a countermeasure to repair the commitment scheme from [20] using *random* (instead of unit) tensors. Our new scheme fills the gap of a post-quantum, non-interactive commitment scheme from non-transitive group actions, which was the aim of the Asiacrypt 2023 commitment scheme [20], and which is still missing in the literature. We stress that our new scheme makes no new assumptions, but only depends on already existing hard problems from random tensors. The scheme we propose is statistically binding and computationally hiding. We conclude by proposing a zero-knowledge proof of opening for our new commitment scheme.

Organization. In Section 2 we summarize the relevant concepts pertaining to group actions, tensors, commitment schemes, and the protocol we will be attacking from [20]. In Sections 3.1 and 3.2 we discuss low rank points and our attack on the hiding property of [20], and in Sections 3.3, 3.4, and 3.5, we compute stabilizers and give an attack that recovers the full secret. Finally, in Section 4,

we give a repair on their commitment scheme that preserves the non-interactive structure and still has statistical binding, with a discussion on relevant zero-knowledge proofs in Section 4.2.

Acknowledgements. We thank the anonymous reviewers for their careful reading and several suggestions. We thank Youming Qiao for the helpful discussions and for referring us to [12]. Valerie Gilchrist is supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium; Christophe Petit is partly supported by EPSRC through grant number EP/V011324/1; Gang Tang is partly supported by the Australian Research Council Linkage Projects LP220100332, Sydney Quantum Academy and EPSRC through grant number EP/V011324/1.

2 Preliminaries

In the rest of this work, we denote by $\mathbf{1}_A(x)$ the function taking value 1 if $x \in A$ and 0 otherwise. We write PPT to stand for Probabilistic Polynomial Time. We call a function *negligible*, written $\mathrm{negl}(\lambda)$, if its absolute value is asymptotically dominated by $O(x^{-n})$ for all $n > 0$.

2.1 Cryptographic group actions

Group actions have been of interest in cryptography because they sometimes provide an easy way to transition classical cryptosystems to post-quantum ones. The first well-studied instance of a group action used in cryptography was in works from Couveignes [17], and Rostovstev and Stolbunov [45], where they developed a Diffie-Hellman-like key exchange scheme from the isogeny group action.

The seminal work from Alamati, De Feo, Montgomery, and Patranabis in [1] generalizes the hard problems taken from the isogeny group action for use with any group action. In doing so, they establish standard definitions and notation that have since been upheld in the field, providing a valuable common ground for proceeding works. To begin, we recall some of these definitions that will be relevant to our work.

Definition 1 (Group action). *We say that a group G acts on a set X if there exists a map $\star : G \times X \to X$ such that :*

- *Identity: If e is the identity element of G, then for any $x \in X$, $e \star x = x$;*
- *Compatibility: For any $g, h \in G$, and any $x \in X$, $(gh) \star x = g \star (h \star x)$.*

Such a group action can be denoted $(G, X, \star)$.

We call a group action *transitive* if for any two elements $x, y \in X$, there exists a group element $g \in G$ mapping x to y, i.e. $y = g \star x$. We say a group action is *free* if for each $g \in G$, g is the identity element if and only if there exists an element $x \in X$ such that $x = g \star x$. A group action is called *regular* if it is both transitive and free.

Definition 2 (Orbit). *The* orbit*, O_x, of an element $x \in X$, is all the $y \in X$ for which there exists an element mapping x to y i.e $O_x = \{y \in X | \exists g \in G, y = g \star x\}$.*

We can reformulate transitivity in terms of orbits : an action is transitive if for all $x, y \in X$, we get that $O_x = O_y$.

Since we are considering group actions in the context of cryptography, we will only be concerned with *effective group actions*.

Definition 3 (Effective group action (EGA)). *We call a group action $\star : G \times X \to X$* effective *if the following properties hold :*

- *G is finite and there exists a* PPT *algorithm for the following operations : group operation, computation of inverses, membership testing, equality testing and sampling.*
- *X is finite and there exists a* PPT *algorithm for membership testing and for computing a unique representation of elements in X.*
- *There exists a distinguished element $x_0 \in X$ such that its bit-string representation is known. We call this point the* origin*.*
- *$\star$ can be computed efficiently for any $g \in G, x \in X$.*

When creating cryptography from group actions, we will want to be able to actually evaluate the group action, hence we will limit ourselves to EGAs. In some instances where only a portion of the group or set can be evaluated efficiently, we can restrict to such a subset. This is called a *Restricted Effective Group Action (REGA)*.

We can impose some additional properties to be fulfilled by the group action in order to build specific cryprographic constructions. We recall a definition from [20] that will be useful later on in this work.

Definition 4 (Decisional Group Action Inverse Problem). *Let $(G, X, \star)$ be a group action, and $t_0, t_1 \in X$ lie in distinct orbits. The decisional Group Action Inverse Problem (dGA-IP) game for $(G, X, \star)$ is described in Figure 1. For an adversary $\mathcal{A}$ against the* dGA-IP *game, we define the advantage* Adv *as follows :*

$$\mathrm{Adv}(\mathcal{A}) = \Pr[1 \leftarrow \mathsf{dGA\text{-}IP}(\mathcal{A})] - \frac{1}{2}.$$

2.2 Tensors over finite fields

We recall some results on tensors, following the notations introduced in [20].

Fix a finite field $\mathbb{F}_q$ for a prime q, and ℓ, m, n positive integers. Then given bases $\{e_i\}_{i=1}^{l}$, $\{f_i\}_{i=1}^{m}$, $\{g_i\}_{i=1}^{n}$ of $\mathbb{F}_q^l$,$\mathbb{F}_q^m$,$\mathbb{F}_q^n$, respectively, a 3-tensor $v \in \mathbb{F}_q^l \otimes \mathbb{F}_q^m \otimes \mathbb{F}_q^n$ is defined as follows

$$v = \sum_{i=1}^{\ell} \sum_{j=1}^{m} \sum_{k=1}^{n} v(i,j,k) e_i \otimes f_j \otimes g_k.$$

$\underline{\mathsf{dGA\text{-}IP}(\mathcal{A})}$

1: $c, b \xleftarrow{\$} \{0,1\}$
2: $g, g' \xleftarrow{\$} G$
3: $s \leftarrow g \star t_c$
4: **if** $b = 1$ **then**
5: $\quad t \leftarrow g' \star s$
6: **end if**
7: **if** $b = 0$ **then**
8: $\quad t \leftarrow g' \star t_{1-c}$
9: **end if**
10: $b' \leftarrow \mathcal{A}(s, t)$
11: **return** $b' = b$

Fig. 1. The dGA-IP game

Alternatively a *3-tensor*, v, can be represented as a 3-way array of field elements

$$v = [[v(i,j,k)]]_{i,j,k=1}^{l,m,n} \in \mathbb{F}_q^{l \times m \times n}.$$

In the same way that matrices encode bilinear forms, 3-tensors encode trilinear forms. More precisely, given a 3-tensor v, one can associate the 3-linear form $L_v : \mathbb{F}_q^l \times \mathbb{F}_q^m \times \mathbb{F}_q^n \to \mathbb{F}_q$ defined as

$$L_v(x^1, x^2, x^3) = \sum_{i_1=1}^{l} \sum_{i_2=1}^{m} \sum_{i_3=1}^{n} v(i_1, i_2, i_3) x_{i_1}^1 x_{i_2}^2 x_{i_3}^3.$$

For the rest of this work, we restrict to the case where $l = m = n$. We will also only be concerned with 3-tensors, even if not explicitly stated, since the work from [27] gives an equivalence between isomorphism problems on 3-tensors and higher dimensional d-tensors. This makes our study of 3-tensors an optimal instance of these problems. Let $\mathbf{V}$ be the tensor space $\mathbf{V} = \mathbb{F}_q^n \otimes \mathbb{F}_q^n \otimes \mathbb{F}_q^n$, and we will use the canonical basis $\{e_i\}_{i=1}^n$ on $\mathbb{F}_q^n$ to construct tensors.

Tensor rank. An essential invariant of tensors that we will be considering is their *rank*.

Definition 5 (Rank 1 tensor). *A tensor, $v \in \mathbf{V}$, is said to have* rank 1 *if it can be written as $v = a \otimes b \otimes c$, for $a, b, c \in \mathbb{F}_q^n$.*

Definition 6 (Rank of a tensor). *The* rank *of a tensor, $v \in \mathbf{V}$, is the minimal r such that we can write v in the form $v = \sum_{i=1}^{r} w_i$, where $\{w_i\}_{i=1}^r \subset \mathbf{V}$ is a set of rank 1 tensors.*

Problem 1 (Computational Tensor Rank Problem). Given a tensor $v \in \mathbf{V}$, compute $\mathrm{rank}(v)$.

It has been shown that Problem 1 is NP-hard [29, 30, 46].

It is shown in [30] that other tensor related problems are NP-hard. These include the decisional and computational versions of problems investigating eigenvalues, singular values, and spectral norms of tensors. These problems, however, are beyond the scope of this paper.

Group action on tensors. We can obtain further hard problems from tensors by defining a group action on them.
More concretely, the tensors are being acted on by the group $G = \mathrm{GL}(n,q) \times \mathrm{GL}(n,q) \times \mathrm{GL}(n,q)$, where $\mathrm{GL}(n,q)$ denotes the general linear group of degree n over $\mathbb{F}_q$, in the following way:

$$\star : G \times \mathbf{V} \to \mathbf{V},$$

$$\Big((A,B,C), \sum_{i,j,k} v(i,j,k) e_i \otimes e_j \otimes e_k\Big) \mapsto \sum_{i,j,k} v(i,j,k) Ae_i \otimes Be_j \otimes Ce_k, \qquad (1)$$

for $v(i,j,k) \in \mathbb{F}_q$.

Lemma 1. *Let $(A,B,C) \in G$, and $v \in \mathbf{V}$. Then $rank((A,B,C)\star v) = rank(v)$. In other words, $(\star)$ preserves rank.*

Proof. We can decompose $w := (A,B,C)\star v$ into a sum as shown in Equation 1, giving us an upper bound on its rank. Thus $rank(w) \leq rank(v)$.

Now consider the tensor $(A^{-1}, B^{-1}, C^{-1}) \star w$, which again can be written as a sum whose index is bounded above by $rank(w)$. This gives us that $rank((A^{-1}, B^{-1}, C^{-1})\star w) \leq rank(w)$. We notice, however, that $(A^{-1}, B^{-1}, C^{-1})\star w = v$, thus we get that $rank(v) \leq rank(w)$ and from before, $rank(w) \leq rank(v)$. Hence $rank(v) = rank(w)$.

□

Given this group action on tensors, we can consider two additional potentially hard problems.

Problem 2 (Decisional Tensor Isomorphism Problem). Given two tensors $v_0, v_1 \in \mathbf{V}$, decide whether there exists $(A,B,C) \in G$ such that $(A,B,C) \star v_0 = v_1$.

Problem 3 (Computational Tensor Isomorphism Problem). Given two tensors $v_0, v_1 \in \mathbf{V}$, such that $(A,B,C) \star v_0 = v_1$ for some $(A,B,C) \in G$, compute (A,B,C).

We can rephrase these problems in terms of *orbits*, from Definition 2 : Problem 2 asks to determine whether two tensors belong to the same orbit; Problem 3 asks to determine the isomorphism mapping two elements from the same orbit.

2.3 Commitment schemes

Commitment schemes have many real-world use cases, finding applications in multiparty computation, zero-knowledge proofs, and coin tossing [24, 33, 36, 40, 43].

A commitment scheme is used when a sender wants to commit to some value, m, without initially revealing it to the receiver. Instead, the sender will commit to m by computing a commitment, c, that depends on m, and send it to the receiver. At a later time, the sender will reveal m, and the receiver should be able to verify that c was computed using m. The key security properties here are *hiding*, which means c should reveal nothing about m; and *binding*, which means no other value $m' \neq m$ should be able to open c. We proceed to give formal definitions for a commitment scheme and its security properties.

Definition 7 (Commitment scheme). *A commitment scheme is defined by a message space, $\mathcal{M}$, randomness space, $\mathcal{R}$, a commitment space, $\mathcal{C}$, and a tuple of algorithms,* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$*, defined in the following way :*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$*: This* PPT *algorithm takes as input a security parameter λ in unitary form and returns some public parameters* pp *which will be implicitly given as inputs to the proceeding two algorithms.*
- $\mathsf{Commit}(m, r) \to c$ *: This* PPT *algorithm takes as input a message $m \in \mathcal{M}$ and randomness $r \in \mathcal{R}$ and outputs a commitment value $c \in \mathcal{C}$.*
- $\mathsf{Verify}(c, (m, r)) \to 0/1$*: This is a deterministic algorithm which, given as input $c \in \mathcal{C}$, $(m, r) \in \mathcal{M} \times \mathcal{R}$ returns 1 if $\mathsf{Commit}(m, r) = c$ and 0 otherwise.*

In the case of a *bit commitment scheme*, the committed messages are restricted to two possible values, 0 or 1.

Definition 8 (Hiding Security). *A commitment scheme* $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Verify})$ *is* hiding *if for any adversary, $\mathcal{A}$, the advantage,* Adv*, is negligible, where* Adv *is defined as*

$$\mathrm{Adv} = |\Pr[1 \leftarrow \mathsf{Hiding}(\mathcal{A})] - 1/2|$$

and the Hiding *game is defined in Figure 2. In this figure, we take m_0, m_1, st to be two messages and a state chosen by the adversary $\mathcal{A}$ for use in the game.*

$\underline{\mathsf{Hiding}(\mathcal{A})}$

1: $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2: $m_0, m_1, st \leftarrow \mathcal{A}(\mathsf{pp})$
3: $b \xleftarrow{\$} \{0,1\}, r \xleftarrow{\$} \mathcal{R}$
4: $c \leftarrow \mathsf{Commit}(m_b, r)$
5: $b' \leftarrow \mathcal{A}(c, st)$
6: **return** $b' = b$

$\underline{\mathsf{Hiding}_{\mathsf{Bit}}(\mathcal{A})}$

1: $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2: $b \xleftarrow{\$} \{0,1\}, r \xleftarrow{\$} \mathcal{R}$
3: $c \leftarrow \mathsf{Commit}(b, r)$
4: $b' \leftarrow \mathcal{A}(c)$
5: **return** $b' = b$

Fig. 2. The Hiding and $\mathsf{Hiding}_{\mathsf{Bit}}$ game

Remark 1. Note that in the case of a bit commitment scheme, we have $\mathcal{M} = \{0,1\}$, and in the Hiding game the adversary does not choose m_0, m_1 and does not need to provide a state st, rather the bit committed is the one sampled by the game, i.e. one can remove Line 2 in Figure 2 and set $m_b = b$, as depicted on the right-hand side of Figure 2 by the $\mathsf{Hiding}_{\mathsf{Bit}}$ game.

Definition 9 (Binding Security). *A commitment scheme* (Setup, Commit, Verify) *is* binding *if for any adversary* $\mathcal{A}$ *the advantage* Adv *is negligible, where* Adv *is defined as*

$$\text{Adv} = \Pr[1 \leftarrow \mathsf{Binding}(\mathcal{A})]$$

and the Binding *game is as defined in Figure 3.*

$\underline{\mathsf{Binding}(\mathcal{A})}$
1: $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2: $m_0, m_1, r_0, r_1 \leftarrow \mathcal{A}(\mathsf{pp})$
3: **return** $(m_0 \neq m_1) \wedge \mathsf{Commit}(m_0; r_0) = \mathsf{Commit}(m_1; r_1)$

Fig. 3. The Binding game

Remark 2. We can make the hiding and binding definition more precise by quantifying the complexity of the adversary as well as its advantage. When $\mathcal{A}$ has unbounded complexity and $\text{Adv} = 0$, the commitment scheme is said to be *perfectly* binding/hiding. When $\mathcal{A}$ has unbounded complexity and $\text{Adv} = \text{negl}(\lambda)$, it is called *statistically* binding/hiding. When $\mathcal{A}$ is PPT , and $\text{Adv} = \text{negl}(\lambda)$ it is called *computationally* binding/hiding.

2.4 Asiacrypt 2023's commitment scheme from tensors

While there is already literature about commitment schemes [38,42], developing post-quantum equivalents is necessary in order to curb the risk of future quantum attacks. Many post-quantum commitments exist from lattice-based assumptions with recent constructions such as [2,6,22], code-based assumptions [31,37], isogeny-based assumptions [47] and *transitive* group actions [11, 14, 32]. *Non-transitive* group actions, however, are less restrictive and arise naturally, making them interesting for use in commitment schemes. In [20], at Asiacrypt 2023, D'Alconzo, Flamini, and Gangemi describe a general bit commitment framework from post-quantum non-transitive group actions that they call GACE (Group Action with Canonical Element), and then give an instance using tensors in Section 6.2 of their work.

In particular, they take inspiration from the fact that it is widely believed to be hard to compute the rank of a random tensor (see Section 2.2), and proven NP-hard [29,30,46]. In the paper they provide a method of constructing a distinguished element for any rank, such that given the rank and some additional data it is easy to verify the correctness of the original claim. Let $b \in \{0, \dots n-1\}$, then their distinguished element is

$$t_b := \sum_{i=1}^{n-b} e_i \otimes e_i \otimes e_i. \tag{2}$$

We will refer to the t_b as *unit tensors*, following the terminology from [12].

In their commitment scheme they use t_b to create either an n-rank tensor or an $(n-1)$-rank tensor, thereby encoding a bit $b \in \{0,1\}$. They then sample an element $(A, B, C) \in G = \mathrm{GL}(n,q) \times \mathrm{GL}(n,q) \times \mathrm{GL}(n,q)$, and compute the commitment $c = (A, B, C) \star t_b$ via the group action described in Equation 1. The sender sends the commitment c to the receiver. When the sender is ready, they will reveal $(A, B, C), b$. The receiver can then easily check that $c = (A, B, C) \star t_b$, and so has rank $n - b$.

Parameters. There are no parameters given for the commitment scheme from [20], and it does not rely on a pure form of the tensor isomorphism problem, so parameter choice for this scheme is not obvious. Though our attacks run in polynomial time complexity, we would like a concrete example instance of the commitment scheme on which to test the algorithms outlined in Section 3. Thus, we will consider the parameters outlined in MEDS [15], a post-quantum signature from matrix code equivalences, submitted to the NIST competition for post-quantum signatures. Their proposed choices of vector dimension and field size, (n, q), are $(14, 4093), (22, 4093), (30, 2039)$.

The matrix code equivalence problem is polynomially equivalent to the general trilinear form equivalence problem and the tensor isomorphism problem as noted in [27], meaning parameter sizes should be the same across all of these hard problems.

2.5 The MinRank problem

In Section 3 we will be exploring how low rank points can affect the security of some tensor-based hard problems. For this, we will need a way of solving a MinRank instance. While some more straightforward approaches to MinRank, such as [34], would suffice for computing low rank points, we choose to use a more general algorithm to eliminate the chance of using different rank tensors as a countermeasure to our attacks later on. Here we summarize the relevant results from [4], the current state-of-the-art for this problem. The authors of this work revise the algebraic approach to MinRank, and thus completely avoid the use of a Gröbner basis computation for some parameters, keeping all the equations linear. When applying this approach to three NIST candidate key exchange schemes, they were able to improve upon or match the state-of-the-art attacks.

The MinRank problem is as follows:

Problem 4 (MinRank problem). Given an integer $r \in \mathbb{N}$, and k matrices $M_1, \dots M_k \in \mathbb{F}_q^{m \times n}$, determine field elements $x_1, \dots x_k$ not all zero, such that

$$\mathrm{rank}\Big(\sum_{i=1}^{k} x_i M_i\Big) \leq r.$$

In previous works [3], the MinRank problem was solved by creating equations using *coefficient* variables and *support* variables depending on maximal minor equations. This system of equations would then be solved using a Gröbner basis algorithm. In [4], they instead define even more bilinear equations using the coefficient and support variables. So many, in fact, that they are sufficient for solving the entire problem, thereby avoiding the Gröbner basis computation in many cases.

The algorithm. We are given n matrices, $M_1, \dots M_n$, of size $n \times n$, and a target rank r. We want to find $\{x_i\}_{i=1}^n$ such that $\mathbf{M} := \sum x_i M_i$ has rank r. So the entries of $\mathbf{M}$ are linear expressions in the variables x_i. When $\mathbf{M}$ has rank r, we can write it as $\mathbf{M} = SR$, where columns of the $n \times r$ matrix S form a basis for the column-space of $\mathbf{M}$, and the $r \times n$ matrix R contains a basis for the row-space of $\mathbf{M}$. The entries in S and R are referred to as the *support* and *coefficient* variables, respectively. The matrix R will necessarily have full rank r. Hence, if we add a row from $\mathbf{M}$ to it, all of its maximal minors will be vanishing. These vanishing maximal minors give us algebraic equations depending on the x_i and the entries of R. We can do this for each row of $\mathbf{M}$, obtaining enough equations to linearize and solve for the x_i.

2.6 Sigma protocols

We give below an informal definition of a sigma protocol that will be of use in Section 4.2.

Definition 10 (Sigma protocol). *A sigma protocol Σ for a relation $\mathcal{R}$ is a protocol between a prover P and a verifier V. It has input $(x, w) \in \mathcal{R}$ where x, the statement, is a common input and w, the witness, is private to P, and it must satisfy the following :*

- *Σ is a three-move protocol, where P sends the first message a, V replies with some string e, P send a final message z upon which V wither accepts or rejects.*
- *It is complete i.e. if P, V follow the protocol on input (x, w) with $(x, w) \in \mathcal{R}$ then V always accepts.*
- *Special soundness: there exists a polynomially bounded algorithm $\mathcal{E}$ called* extractor *such that for any x, if (x, a, e, z) and (x, a, e', z) are two accepting views for $\mathcal{V}$ such that $e \neq e'$ then $\mathcal{E}(x, a, e, z, e', z')$ yields w such that $R(x, w)$.*
- *Special honest-verifier zero-knowledge : there exists a polynomially bounded algorithm Sim called* simulator *such that for any $x \in L$ and e, the transcript (a, e, z) of the interaction $\mathcal{P} \overset{x}{\leftrightarrow} \mathcal{V}$ conditioned to e has the same distribution as $\mathsf{Sim}(x, e)$.*

A sigma-protocol can be turned into a non-interactive zero-knowledge proof using the Fiat-Shamir transform [23].

3 Solving Tensor Isomorphism Problems in t_b orbits

In this section we focus on attacking the commitment scheme from [20], which uses special cases of tensor hard problems. We begin in Section 3.1 by establishing some background on the rank of points in tensors. This will help us in Section 3.2, where we focus on attacking a variant of the Decisional Tensor Isomorphism Problem (DTI, see Problem 2) present in their commitment scheme, and provide a polynomial time attack on it. We then describe some elements of the stabilizer group of the family of tensors $\{t_b : b \in \mathbb{Z}_n\}$ in Section 3.3. Lastly, in Sections 3.4 and 3.5, we use these findings to attack their variant of the Computational Tensor Isomorphism Problem (CTI, see Problem 3).

All the algorithms and experiments described in the proceeding sections were coded in Magma, and can be found at

https://github.com/vgilchri/tensor-group-action .

3.1 Computing the rank of points

In the case of tensors, we can first choose to view a 3-tensor, $g \in \mathbf{V}$, as a list of n matrices $G_1, \dots G_n \in \mathbb{F}_q^{n \times n}$. Given $u, v, w \in \mathbb{F}_q^n$, we represent $u \otimes v \otimes w$ as $[G_1, ..., G_n]$ where $G_i = u_i \cdot (v \cdot w^T)$. Note, there are two additional ways to do this, which essentially changes along which axis you are "slicing". While the choice of this axis is not important, choosing one and staying consistent is. We highlight our chosen representation in the following example.

Example 1. Suppose we are working over $\mathbb{F}_5^3$. We would like to compute the tensor $u \otimes v \otimes w$, where $u = [4, 1, 2], v = [3, 0, 1], w = [2, 4, 0]$.

We proceed by first expanding the matrix $v \cdot w^T$:

$$v \cdot w^T = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 0 & 0 \\ 2 & 4 & 0 \end{bmatrix}.$$

Now we multiply this matrix by each entry of u, storing them in a list as we go and we obtain the following :

$$\begin{bmatrix} 4 & 3 & 0 \\ 0 & 0 & 0 \\ 3 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 2 & 0 \\ 0 & 0 & 0 \\ 2 & 4 & 0 \end{bmatrix}, \begin{bmatrix} 2 & 4 & 0 \\ 0 & 0 & 0 \\ 4 & 3 & 0 \end{bmatrix}.$$

Using this representation of tensors, we can define the rank of a *point* in a tensor.

Definition 11 (Rank of a point). *Let $g \in \mathbf{V}$ be a tensor, written in the form $g = [G_1, \dots G_n]$. Then the rank of a point $u := [u_1, \dots u_n] \in \mathbb{F}_q^n$ in g is exactly the matrix rank of $u_1 G_1 + \dots u_n G_n$.*

Lemma 2. *The set of rank 0 points of a tensor, $v \in \mathbf{V}$, is a vector space.*

Proof. Associativity, commutativity, and distributivity are all inherited from $\mathbb{F}_q^n$. The zero-vector in $\mathbb{F}_q^n$ is trivially a rank 0 point in any tensor, and so serves as the additive identity element for the vector space. Furthermore, we have closure under additive inverses and scalar multiplication since multiplication by a non-zero scalar does not affect the rank of a matrix. Lastly, given two rank 0 points u and w, we get that $u + w$ is a rank 0 point, since the sum of rank 0 matrices will have rank 0 as well. □

Computing the rank of points in a tensor will be useful to us later on. In the rest of this paper, when talking about the rank of points, we will consider points up to scalar multiplication, unless stated otherwise.

Lemma 3. *Let $v \in \mathbf{V}$ be a tensor. Denote $L_k(v) := \{u \in \mathbb{F}_q^n : rank_v(u) = k\}$ to be the set of rank k points in v. Now let $(A, B, C) \in G$ be an isomorphism, and $w = (A, B, C) \star v$. Then $|L_k(v)| = |L_k(w)|$.*

In particular, we get a bijection that sends $u \in L_k(v)$ to $uA^{-1} \in L_k(w)$.

Proof. We observe that, given a tensor v, the action of B and C does not affect the rank of a point, meaning that a point u of rank r in v will also be of rank r for $(I, B, I) \star v$, and $(I, I, C) \star v$. This follows straightforwardly from the fact that given $v = [T_1, ..., T_n]$, we can write the action of (I, B, I) as $(I, B, I) \star v = [BT_1, ..., BT_n]$ and that of (I, C, I) as $(I, I, C) \star v = [T_1C^T, ..., T_nC^T]$ with ^{T} denoting the transpose.
We observe that only the action of A has some influence on the rank of points, and A sends a rank r point u in v to the rank r point $u' = uA^{-1}$ in $(A, I, I) \star v$, where $\star$ is as defined in Equation 3.3.
Combining these results, we obtain a one-to-one correspondence between the rank r points in v, and the rank r points in $w = (A, B, C) \star v$, with an explicit mapping, which proves the statement. □

When considering tensors, Lemma 3 tells us that the number of points of fixed rank in a tensor is preserved via the group action ($\star$) from Equation 3.3. Computing a group action in this case amounts to transforming the bases of $\mathbb{F}_q^n$ being used to compute the tensor, to new bases, via linear transformations (recall we require three bases of $\mathbb{F}_q^n$). Hence, the overall structure or rank of the tensor does not change.

Rank 0 points in tensors, when considering a tensor as a list of matrices, means that there exists some linear dependency between these matrices. Transforming the bases of $\mathbb{F}_q^n$ preserves this dependency, regardless of the exact changes of bases being used. Solving for these points can be done with Gaussian elimination, as we will see in Section 3.2. Solving for higher rank points will amount to a MinRank problem, described in Section 2.5.

3.2 Hiding attack

We now describe an attack that breaks the hiding property of the commitment scheme described in [20]. That is, given a committed value $c = \mathsf{Commit}(b, r)$ for

a given bit b, we recover b by solving a system of linear equations, therefore disproving the security claims made in [20].

Let us recall the problem we are attacking:

Problem 5. Given $c = (A, B, C) \star t_b$ with $t_b = \sum_{i=1}^{n-b} e_i \otimes e_i \otimes e_i$, $b \in \{0, \dots n-1\}$, $(A, B, C) \in G$ sampled randomly, and $\star$ is as defined in Equation 3.3, recover b.

Note, this problem is slightly more general than the commitment scheme from [20], since we are considering any $b \in \{0, 1, \dots n-1\}$. From Problem 5, we see that c is precisely the tensor t_b with an isomorphism acting on it, as described in Lemma 3.

Lemma 4. *Let $t_b = \sum_{i=1}^{n-b} e_i \otimes e_i \otimes e_i$, then the vector space of rank 0 points has dimension b.*

Proof. Lemma 2 immediately tells us that the set of rank 0 points is a vector space. It remains to prove the dimension.

Let us write t_b as an n list of $n \times n$ matrices, which we denote $T_b^1, \dots T_b^n$. These matrices can be described in the following way :

$$(T_b^k)_{ij} = \begin{cases} 1 \text{ if } i = j = k \text{ and } k \leq n - b \\ 0 \text{ otherwise} \end{cases}$$

We know that the rank of a point $u \in \mathbb{F}_q^n$ of t_b is the rank of the matrix $U_b = u_1 T_b^1 + .. + u_n T_b^n$.
Using the above definition of the T_b^i, $i = 1, .., n$ we observe that U_b will be a diagonal matrix with entries

$$(U_b)_{ii} = \begin{cases} u_i, \text{ for } i = 1, ..., n - b \\ 0 \text{ otherwise} \end{cases} .$$

Suppose $u \neq 0 \in \mathbb{F}_q^n$, then U_b will consist of a diagonal matrix of dimension $(n-b) \times (n-b)$ with the remaining b rows/columns being completely zeros. This upper bounds the rank to be at most $n - b$. For example, U_0 will have rank at least 1 when u is non-zero, and therefore no non-trivial rank 0 points and thus the subspace has dimension 0. Similarly U_1 will have one zero column so rank at most $n - 1$ and will always have a rank 0 point (up to scalar multiplication), in the form $u_1 = ... = u_{n-1} = 0$ and u_n any non-zero element of $\mathbb{F}_q$.

More generally, for t_b where $b > 0$, letting e_i denote the canonical basis element of $\mathbb{F}_q^n$, we can define the basis B generating the vector space of rank 0 points in t_b as

$$B := \{e_n, \dots e_{n-b+1}\}.$$

□

We focus for a moment on the case $b \in \{0, 1\}$. Using Lemma 4, we have that t_1 has exactly one rank 0 point, and that t_0 has no rank 0 points. Combining

this with Lemma 3 shows that we can use the existence of a rank 0 point as a distinguisher on c (as defined in Problem 5). We can find the rank 0 points of c by solving the system of n^2 linear equations

$$\alpha_1 G_1 + ... + \alpha_n G_n = 0, \tag{3}$$

for $\alpha_i \in \mathbb{F}_q$, $i = 1, ..., n$. If such a solution exists, then $b = 1$, else $b = 0$. For the more general case of $b \in \{0, 1, \dots n-1\}$, the dimension of the solution space will be exactly b.

While more complex algorithms could be used to solve this system, Gaussian elimination is sufficient for our purposes. The challenge here is selecting a set of n linearly independent equations from the n^2 possible choices. In the worst case this gives an upper bound on the number of operations of $O(n^4)$. This proves Theorem 1.

Theorem 1. *There exists an algorithm that solves Problem 5 using $O(n^4)$ operations.*

For all parameters highlighted in Section 2.4 the attack runs in under a second on a laptop. While running these experiments we found that sampling the first n equations was sufficient for correctly solving the system. While we would not expect this to be true for a random set of n^2 equations, this could be due to the structure of the tensor which is not random. So heuristically, we expect a complexity of $O(n^3)$. Note that Theorem 1 always returns a correct answer to Problem 5.

To this end, we give a second probabilistic algorithm that also requires $O(n^3)$ operations but uses a slightly different approach to random sampling.

Theorem 2. *There exists a probabilistic algorithm that solves Problem 5 using $O(n^3)$ operations.*

Proof. Recall from Problem 5, given $c = (A, B, C) \star t_b$, we would like to recover b.

We begin by restricting to the case $b \in \{0, 1\}$. Consider a random point $u \in \mathbb{F}_q^n$. Then u has rank n in t_0 and rank $n-1$ in t_1 with very high probability. Namely u will have rank n in t_0 with probability $(q-1)^n/q^n$ since there can be no zeros in this point. Similarly, in t_1, one of the entries can be free but the rest must be non-zero, so we get probability $(q-1)^{n-1}/q^{n-1}$. By Lemma 3, these probabilities will translate to c. Thus, it suffices to compute the rank of u in the commitment, c, to recover b with overwhelming probability, for large q. This approach generalizes to $b \in \{0, 1, \dots n-1\}$, in the obvious way.

This algorithm consists of computing a random linear combination of n square matrices, and computing the sum's rank over the finite field $\mathbb{F}_q$. Using Gaussian elimination, we can do this in $O(n^3)$ complexity. □

Remark 3. This attack method works because of how structured t_b is. For a random tensor we do not expect to have any rank 0 point (this will be shown in Lemma 12). For this reason, our distinguishing attack is not expected to have

any impact on the general case of Problem 2 (the Decisional Tensor Isomorphism Problem), which asks an attacker to determine whether two given tensors are isomorphic. For the same reason, we also do not claim any improvement on Problem 3 (the Computational Tensor Isomorphism Problem). To date, both of these problems are believed to be cryptographically hard in the general setting, and we will use them in Section 4 to build a new commitment scheme from random tensors.

3.3 Stabilizer subgroup of t_b

We now investigate some elements belonging to the stabilizer group of tensors, and construct the entire stabilizer group of t_0. These findings will be of use to us in an attack in later sections.

Stabilizers for all tensors. Recall, we define $v \in \mathbf{V}$ as

$$v := \sum_{i,j,k=1}^{n} v(i,j,k) e_i \otimes e_j \otimes e_k,$$

and we are considering isomorphisms $(A, B, C) \in G$ that act on v via the group action, $(\star)$, defined as

$$(A, B, C) \star \sum_{i,j,k} v(i,j,k) e_i \otimes e_j \otimes e_k = \sum_{i,j,k} v(i,j,k) Ae_i \otimes Be_j \otimes Ce_k.$$

Lemma 5. *Let $\lambda_a, \lambda_b, \lambda_c \in \mathbb{F}_q$ be such that $\lambda_a \lambda_b \lambda_c = 1$. Then for all $v \in \mathbf{V}$, we have that $(\lambda_a I_n, \lambda_b I_n, \lambda_c I_n) \star v = v$.*

The proof of this lemma can be done straightforwardly using the definition of the group action from Equation 3.3. We reserve the details for Appendix A.

Stabilizer subgroup of t_b. Now we focus our attention to t_b, which can be written as

$$t_b := \sum_{i=1}^{n-b} e_i \otimes e_i \otimes e_i.$$

In the following two lemmas we formalize how diagonal matrices and permutation matrices can be used to create stabilizing elements for t_b, and more specifically, for t_0.

Lemma 6. *Suppose $\Lambda_A, \Lambda_B, \Lambda_C$ are three diagonal $n \times n$ matrices over $\mathbb{F}_q$ such that $\Lambda_A \Lambda_B \Lambda_C = I_n$. Then $(\Lambda_A, \Lambda_B, \Lambda_C) \star t_b = t_b$, i.e. $(\Lambda_A, \Lambda_B, \Lambda_C)$ is in the stabilizer subgroup of t_b.*

Proof. We claim that $(\Lambda_A, \Lambda_B, \Lambda_C)$ is an element of the stabilizing group of t_b. We can see this explicitly as

$$\begin{aligned}(\Lambda_A, \Lambda_B, \Lambda_C) \star t_b &= \sum_{i,j,k=1}^{n-b} \sum_{s=1}^{n-b} \Lambda^A_{is}\Lambda^B_{js}\Lambda^C_{ks} e_i \otimes e_j \otimes e_k \text{ (the matrices are diagonal)} \\ &= \sum_{i=1}^{n-b} \Lambda^A_{ii}\Lambda^B_{ii}\Lambda^C_{ii} e_i \otimes e_i \otimes e_i \\ &= \sum_{i=1}^{n-b} e_i \otimes e_i \otimes e_i = t_b.\end{aligned}$$

□

Additional stabilizer elements for t_0. To explore how permutation matrices can be used to build stabilizing elements, we first focus on the case of $b = 0$. In what follows, we use S_n to denote the symmetric group of integers $\{1, \dots n\}$.

Lemma 7. *Let $\sigma \in S_n$, and let P_σ be the associated permutation matrix given by $P_\sigma = (p_{ij})_{i,j=1}^n$ and $p_{ij} = \mathbf{1}_{i=\sigma(j)}$. Then $(P_\sigma, P_\sigma, P_\sigma) \star t_0 = t_0$, i.e. $(P_\sigma, P_\sigma, P_\sigma)$ is in the stabilizer subgroup of t_0.*

Proof. Consider any permutation matrix P_σ. When applied consistently to the bases being used to construct $t_0 = \sum_{s=1}^n e_s \otimes e_s \otimes e_s$, it simply permutes the order in which the summation is performed, resulting in the same tensor t_0. We reserve the explicit calculations showing this equality for Appendix A. □

In [12, Proposition 4.1], they state that the entire stabilizer group of t_0, defined over the complex numbers, is exactly the semi-direct product of the diagonal matrices from Lemma 6 and the set of permutation matrices P_σ for $\sigma \in S_n$. This is consistent with our findings over finite fields, and was further corroborated via our experiments.

Additional stabilizer elements for t_1. We now turn to the study of the stabilizer elements specific to t_1.

Lemma 8. *Let $\sigma \in S_{n-1}$, and let P_σ be the associated permutation matrix given by $P_\sigma = (p_{ij})_{i,j=1}^{n-1}$, $p_{ij} = \mathbf{1}_{i=\sigma(j)}$ and M_σ be the $n \times n$ matrix obtained by adding an additional zero row and zero column to P_σ, in the following way :*

$$M_\sigma := \begin{pmatrix} P_\sigma & \mathbf{0}_v \\ \mathbf{0}_h & 0 \end{pmatrix}$$

where 0_v and 0_h are zero column and zero row vectors respectively.
Then $(M_\sigma, M_\sigma, M_\sigma) \star t_1 = t_1$, i.e. $(M_\sigma, M_\sigma, M_\sigma)$ is in the stabilizer subgroup of t_1.

Proof. As in Lemma 7, the action of the permutation matrices is permuting the order of the bases being used to construct $t_1 = \sum_{i=1}^{n-1} e_i \otimes e_i \otimes e_i$. In this case, however, we risk mapping one of the basis vectors $\{e_1, \dots e_{n-1}\}$ to e_n, which should not be included in our summation. Hence it is crucial to exclude the permutations that do so by forcing the final row/column to be zeros. This leaves us with the permutations from S_{n-1}. □

Lemma 9. *Let I be the $(n-1) \times (n-1)$ identity matrix, $\mathbf{v} = (v_1, ..., v_{n-1}) \in \mathbb{F}_q^{n-1}, v_n \in \mathbb{F}_q$ and let M be a matrix of the following form:*

$$M := \begin{pmatrix} I & \mathbf{v} \\ \mathbf{0}_h & v_n \end{pmatrix}$$

where 0_h is a zero row vector. Then $(M, I, I) \star t_1 = t_1$, $(I, M, I) \star t_1 = t_1$, and $(I, I, M) \star t_1 = t_1$. i.e. these three families of isomorphisms are in the stabilizer subgroup of t_1.

Proof. Recall, applying an isomorphism (A, B, C) to t_1 means

$$(A, B, C) \star t_1 = \sum_{i=1}^{n-1} Ae_i \otimes Be_i \otimes Ce_i.$$

Since we are restricted to $i < n$, every vector contains a zero in the n^{th} component. Thus the n^{th} columns of A, B, C will always be multiplied by zeros. Therefore the families of isomorphisms $\{(M, I, I), (I, M, I), (I, I, M)\}$ are contained in the stabilizer subgroup of t_1. □

Equivalence classes on G. The study of stabilizer groups leads us to the definition of an equivalence relation on $G = \mathrm{GL}(n, q) \times \mathrm{GL}(n, q) \times \mathrm{GL}(n, q)$. This observation will allow us to add more constraints to the solutions we are looking for in CTI (Computational Tensor Isomorphism Problem), and thus accelerate our attack in the proceeding section.

Definition 12. *Let $(A_1, B_1, C_1), (A_2, B_2, C_2)$ be two elements from G, and $t \in \mathbf{V}$ a tensor. We say that (A_1, B_1, C_1) and (A_2, B_2, C_2) belong to the same* equivalence class via t*, denoted $(A_1, B_1, C_1) \equiv_t (A_2, B_2, C_2)$, if and only if they are equal up to right-multiplication by an element in the stabilizer subgroup of t.*

3.4 Solving CTI in orbits of t_0

In this section we focus on the $b = 0$ case from the commitment scheme, and outline an approach to recovering an element from the isomorphism class via t_0 of the secret (A, B, C). Recall our problem is as follows:

Problem 6. Let $t_0, g \in \mathbf{V}$ be two 3-tensors such that

$$t_0 := \sum_{i=1}^{n} e_i \otimes e_i \otimes e_i, \text{ and } g := \sum_{i,j,k=1}^{n} g_{i,j,k} e_i \otimes e_j \otimes e_k$$

for $\{g_{i,j,k}\} \subset \mathbb{F}_q$. Assuming it exists, find an isomorphism $(A, B, C) \in G$ such that

$$(A, B, C) \star t_0 = g, \tag{4}$$

where $\star$ is as defined in Equation 3.3.

We begin by noticing that the tensor t_0 has exactly n rank 1 points (up to scalars), which we will prove in Lemma 10. Then from Lemma 3 this tells us that g will also have n rank 1 points. Denote a set of rank 1 points of t_0 as $\{e_1, \dots e_n\}$. Suppose we could find the rank 1 points in g, denoted $\{a_1, \dots a_n\}$. Then, we can try to match the e_i to the a_i to recover part of the isomorphism, namely A^{-1}. We formalize this idea in Lemma 11.

Lemma 10. *Let $t_0 = \sum_{i=1}^{n} e_i \otimes e_i \otimes e_i$ then t_0 has exactly n rank 1 points (up to scalars).*

Proof. Let $u := [u_1, \dots u_n]$ be a rank 1 point in t_0, define matrices $T_0^1, \dots T_0^n$ as

$$(T_b^k)_{ij} = \begin{cases} 1 \text{ if } i = j = k \text{ and } k \leq n - b \\ 0 \text{ otherwise} \end{cases}.$$

Since u has rank 1 in t_0, we expect the following diagonal matrix to have rank 1 (by definition of rank of a point, see Definition 11).

$$U_0 := \begin{pmatrix} u_1 & & 0 \\ & \ddots & \\ 0 & & u_n \end{pmatrix}$$

Thus, for U_0 to have rank 1, we see that u can have only one non-zero entry. This gives n distinct points up to scalars. □

Lemma 11. *Consider Problem 6. Let $\{a_1, \dots, a_n\}$ be the set of rank 1 points of g, and $\{e_1, \dots, e_n\}$ be the set of rank 1 points in t_0 (up to scalar multiplication). Then there exists an ordering, σ, of $\{a_1, ..., a_n\}$ and a matrix A such that for each $i \in [1, \dots n]$, $a_{\sigma(i)} = e_i A^{-1}$.*

Proof. As a consequence of Lemma 3, we see that a rank one point in t_0 is sent to a rank one point in g through A^{-1}. This will fix the ordering σ.

□

Finding the rank 1 points $\{a_1, \dots a_n\}$ is an instance of MinRank, so we can use the algorithm given in [4] that is summarized in Section 2.5. In this particular context, we want to find n matrices with target rank 1. This means that our coefficient variable matrix, R, is simply a row matrix. Denote the entries in R as $r_1, \dots r_n$. This leaves us with a bilinear system in the target variables, $\{x_i\}_{i=1}^n$, and the coefficient variables, $\{r_i\}_{i=1}^n$, where we will have $n^2(n-1)/2$ equations and $n(n-1)$ monomials. If the equations are not related by linear dependencies,

we can solve the system by direct linearization. In our experiments, this was indeed the case.

Going forward, Lemma 11 shows that we can recover some matrix A_0^{-1} whose rows are given by $\{a_i\}_{i=1}^n$. Our findings from Section 3.3 tell us that neither the permutation of these rows, nor the scalar multiples of them, will affect our search for a suitable isomorphism satisfying Equation 4. In fact, fixing these rows and scalar multiples is a good idea as it reduces the number of solutions and variables in our search.

Once we have recovered some A_0^{-1}, we can then solve for B_0 and C_0 to complete the isomorphism by considering the system of n^3 linear equations in $2n^2$ variables given by $(I, B_0, I)\star t_0 = (A_0^{-1}, I, C_0^{-1})\star g$. The system, as is, is very under-determined because of the size of the equivalence class of (A, B, C) via t_0. To slim down the solution space, thanks to Lemma 6, we can similarly normalize the first row of B_0, by setting the entries of this row to arbitrary values of $\mathbb{F}_q^*$.

There is a chance that this will not give a solution, in case one or more of the entries in the first row of B were 0. We argue in the proof of Theorem 3 that when q is large enough compared to n, the probability of this failure is very low. In the small chance of failure, however, we can apply a new known isomorphism, $(I, B_1, I) \in G$, and now consider $(I, B_1, I)\star((A, B, C)\star t_0) = (I, B_1, I)\star g$ instead. This essentially "rerandomizes" our instance, in the hopes that the solution space for $B_1 \cdot B$ will contain an entry whose first row has no zeros. Alternatively, we can also normalize a randomly chosen element in each column of B.

These steps are summarized in Figure 4.

1: **Given**: $g \in \mathbf{V}$, isomorphic to t_0.
2: $a_1, \ldots a_n \leftarrow$ `MinRank`$(g, \text{target rank} = 1)$
3: $A_0^{-1} \leftarrow$ `Matrix`$(a_1, \ldots a_n)$
4: $B_0, C_0 \leftarrow$ `Solutions`$((I, B_0, I)\star t_0 = (A_0^{-1}, I, C_0^{-1})\star g) \cap$ `Solutions`$((B_0)_{1,i} = 1, i = 1, \ldots n)$
5: **return** (A_0, B_0, C_0)

Fig. 4. Solving Problem 6.

Theorem 3. *There exists a probabilistic algorithm that solves Problem 6 using $O(n^6)$ operations.*

Proof. We first show that the probability of failure of Algorithm 4 is negligible for large q. When q is small, in case of failure one can just apply a new isomorphism $(I, B_1, I) \in G$ to g, and repeat the algorithm with this new instance as described above.

We have a chance of failure when we normalize the first row of B_0. Since we have already fixed A_0, this fixes the permutation of columns of B_0 as well (see Lemma 7). Hence, we risk failure when one of these normalized columns necessarily has a leading zero. The chance of this happening is the complement

probability to the columns having no leading zero. This gives us a total probability of failure of

$$P(\text{Alg } 4 \text{ fails}) = 1 - \frac{(q-1)^n}{q^n},$$

which is negligible for large q.

We now prove the complexity of this algorithm. The dominating subroutines in Algorithm 4 include the MinRank computation, the tensor group action arithmetic, and solving the final system of equations (which can be done using Gaussian elimination). While the tensor arithmetic could be done in time $O(n^4)$, as noted in Section 6.2 of [48], the other two subroutines require $O(n^6)$ operations. □

Remark 4. An alternate approach to proving Theorem 3 could be to apply Lemma 11 over all three axis, allowing us to recover some A, B, C, each up to some stabilizer elements. The issue with this approach, however, is that those stabilizer elements will a priori not be the same for all three matrices, so at this point we can write a system of equations to recover these stabilizer elements and finish the attack. This approach, while correct, adds an extra step, making it neither conceptually simpler nor more efficient than what is outlined above. Additionally, the above approach has the advantage of directly using the partial knowledge we have about A when recovering B and C.

We test the attack from Theorem 3 on the parameters from Section 2.4 and give the results in seconds in Table 1. We consider some additional parameters in Figure 5 to further support our complexity claims.

n	q	time (s)
14	4093	9.3
22	4093	141.6
30	2039	858.9

Table 1. Timings in seconds for solving the CTI variant from Problem 6.

3.5 Solving CTI in orbits of t_1

The approach of computing the set of rank 1 points via the MinRank algorithm from [4] will only work in the case the tensor has rank n (i.e. $b = 0$ in the commitment scheme). Any smaller rank would result in too many rank 1 points to compute. In the $b = 1$ case, however, we will see that we can edit MinRank to further filter distinct rank 1 points using our knowledge of the stabilizer group of t_1.

We begin by computing the one rank 0 point in g (which is unique up to scalars). This point is easy to compute, as seen in Section 3.2. Denote it P_0.

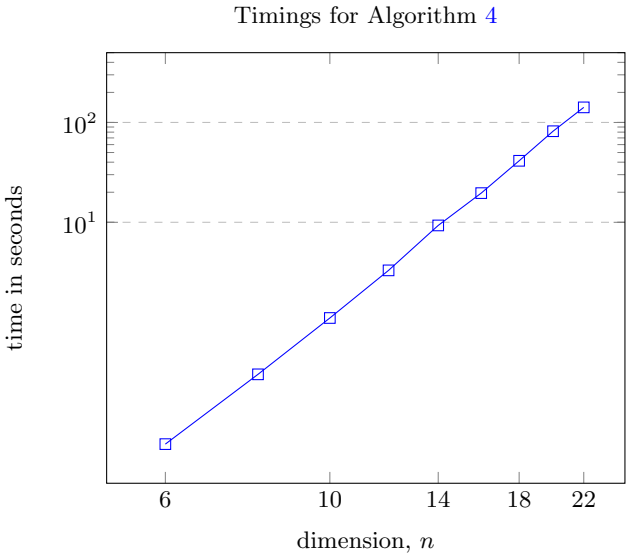


Fig. 5. We show some timings of Algorithm 4 for $q = 4093$ and varying values of n on a log-log scale. This corroborates our claim that Algorithm 4 has complexity $O(n^6)$, since the curve follows very closely to a line of slope 6.

Now, we know that any rank 1 point could have a rank 0 point added to it, and still remain a rank 1 point. In other words, we get an equivalence class on the set of rank 1 points where we say two rank 1 points, R_i, R_j, are equivalent if $R_i - R_j = \lambda P_0$ for some scalar λ. This means, for each $i \in [1, \dots n-1]$, the equivalence class can be written as the subspace $\langle P_i, P_0 \rangle$, for some rank 1 point P_i, and the rank 0 point P_0. Hence, we can obtain a representative from each equivalence class, P_i, by appropriately filtering the results returned from MinRank. More precisely, we normalize a component of the rank 1 points, thus filtering much of the equivalence class. The index of the component to be normalized corresponds to the index of the first non-zero entry of P_0.

After these computations, we will be left with one rank 0 point and $n-1$ distinct rank 1 points. We are assured to get exactly $n-1$ linearly independent rank 1 points since each equivalence class, $\langle P_i, P_0 \rangle$, is the image under an isomorphism of the subspace $\langle e_i, e_n \rangle$. We can then follow a similar attack as that described in Section 3.4, Figure 4, to reconstruct an isomorphism $(A_0, B_0, C_0) \equiv_{t_1} (A, B, C)$.

First we need to be careful to place the rank 0 point in the final row of A_0^{-1} due to the restriction on permutation matrices in the stabilizer group of t_1, as shown in Lemma 8. We can also use Lemma 9 to further filter out options by fixing the final columns of B_0, C_0 to random elements. To do so we slightly modify the equations we are solving to avoid high degree equations involving the inverse of C_0. We consider the new equation $(I, B_0, C_0) \star t_1 = (A_0^{-1}, I, I) \star g$. This new equation is no longer linear since the left hand side will have quadratic

terms in the variables of B_0 and C_0. This means we cannot solve the system of equations linearly, but instead will require a Gröbner basis. Experiments show the run times of this attack to be very close to those of the computational attack on t_0, which we give in Table 2. Thus, we also estimate the complexity of the overall attack to be polynomial.

n	q	time (s)
14	4093	8.7
22	4093	158.1
30	2039	1235.9

Table 2. Timings in seconds for solving CTI for t_1.

4 Repairing the Asiacrypt 2023 scheme

In this section, we propose a new construction of a commitment scheme and include proofs of its security. This commitment is the first non-interactive commitment scheme from non-transitive group actions proposed; such group actions are less restrictive, and arise naturally. In particular, tensor-based group actions are such an example. At the moment isogenies have been among the only group actions receiving attention. By studying tensors in this application, and showing how they can be used in protocols, our hope is to contribute to the assumption diversification of the field.

The key components that we aim to preserve from [20] are the following: a commitment scheme based on a non-transitive group action for which the group action should be uncertified[5], and the commitment scheme should be non-interactive.

We therefore propose a non-interactive bit commitment scheme from non-transitive uncertified group actions, described in Figure 6. It is statistically binding (Theorem 4) and computationally hiding (Theorem 5). We improve on [32]'s construction, which is also statistically binding and computationally hiding but requires interaction. We lose the perfect binding property of [20], however, this allows us to have a concrete instantiation that relies on standard problems. In practice, replacing perfect binding by statistical binding does not affect the security of more advanced constructions that could be built from bit commitments. Note that we do not keep the framework of Problem 5, but rather introduce a slightly different and simpler one, which still does not introduce any new assumptions.

[5] An *uncertified* group action is a group action for which checking that two elements are in the same orbit is hard.

Definition 13 (Decisional group action framework). *Consider a group action $\star : G \times V \to V$ that has the following properties :*

1. *It is an Effective Group Action.*
2. *Given two elements v_0, v_1 such that $v_1 = g \star v_0$ for some $g \in G$, computing g is hard.*
3. *The probability, p, that two random elements are in different orbits is $p = 1 - negl(\lambda)$.*
4. *The Decisional Group Action Inversion Problem for $\star$ is hard (recall Definition 4).*

Note that the first three properties are described in [32].

$\underline{\mathsf{Setup}(1^\lambda)}$
1: $v_0, v_1 \xleftarrow{\$} V$
2: **return** v_0, v_1

$\underline{\mathsf{Commit}(b)}$
1: $g \xleftarrow{\$} G$
2: $c \leftarrow g \star v_b$
3: **return** c

$\underline{\mathsf{Verify}(c, (b, g))}$
1: **return** $(c = g \star v_b)$

Fig. 6. Commitment scheme

We can instantiate this framework using tensors through the group action described in Equation 3.3. The crucial difference with the construction of [20] is that v_0, v_1 are now two **randomly** generated tensors and they do not possess any special structure. In particular, a random tensor will only have low rank points with negligible probability, as shown in the Lemma 12.

Remark 5. One can extend the commitment scheme to accommodate a polynomial number of messages say $0, \dots N$ by slightly modifying the Setup and Commit algorithms of Figure 6. Indeed, during Setup one can instead randomly sample N elements $v_0, \dots v_N$ and output these. Then during Commit, to commit to a message $k \in \{0, \dots N\}$, one samples $g \in G$ and computes $g \star v_k$.

Remark 6. Note that v_0, v_1 must be sampled randomly, in practice this can be done by sampling them as the output of a pseudo-random generator or a cryptographic hash function. What matters for our proofs is that they are indeed indistinguishable from elements sampled from a uniformly random distribution.

Lemma 12. *Given a random tensor $t \in \mathbf{V}$, the average number of rank $n - d$ points (up to scalar multiplication) is q^{-d^2+n-1} as q goes to infinity.*

Proof. Consider our tensor t as a list of n random $n \times n$ matrices, over $\mathbb{F}_q$, say $M_1, \dots M_n$. A rank r point in t is $\lambda = (\lambda_1, \dots, \lambda_n) \in \mathbb{F}_q^n$, $\lambda \neq 0$, such that $rank(\lambda_1 M_1 + \dots + \lambda_n M_n) = r$.
First, we have that the probability that a random matrix M has rank $n - d$ for some integer d goes to q^{-d^2} as q goes to infinity. This result is a direct observation from [25]. Then we observe that there are about q^{n-1} possibilities projectively for λ, hence we get that the average number of rank $n - d$ points is q^{-d^2+n-1}. □

This means that our distinguishing attack from Section 3.2 is not applicable on a randomly generated instance. Indeed, this lemma shows that it is unlikely for a random tensor to have points of small rank. For points of larger rank, the MinRank attack described in Section 2.5 becomes impractical.

Let us now check that all the desired properties from Definition 13 are satisfied: Property 1 has been studied in [32] and Property 2 corresponds to the Computational Tensor Isomorphism Problem (Problem 3). Property 4 is introduced in [20] and they discuss its applicability to tensors. While we show it is broken for their instantiation using unit tensors, t_b, it is still believed to be hard in the case of random tensors, which is precisely our case. Regarding Property 3, we must compute the probability, p, of two random tensors being in different orbits under $\star$. Note that [32] assumes that this probability is high but they do not give any estimates.

Lemma 13. *The probability, p, of two random tensors being in distinct orbits under $\star$ is $p \geq 1 - \frac{1}{q^{n^3-3n^2}}$.*

Proof. Let $t \in \mathbf{V}$ be any tensor, then trivially

$$\#Orb(t) \leq \#G = (\#\,\mathrm{GL}_n(q))^3 = O(q^{3n^2}).$$

Furthermore, we have $\#\mathbf{V} = q^{n^3}$. So the total number of orbits is at least $\#\mathbf{V}/(\#\,\mathrm{GL}_n(q))^3 = O(q^{n^3-3n^2})$. The probability of two tensors being in the same orbit is therefore bounded by $\frac{1}{q^{n^3-3n^2}}$, hence $p \geq 1 - \frac{1}{q^{n^3-3n^2}}$. □

We have shown that all the properties from Definition 13 are satisfied. Now it remains to show that our proposed commitment scheme from Figure 6 is both hiding and binding.

Theorem 4 (Binding). *The commitment scheme described in Figure 6 is statistically binding.*

Proof. Consider G to be the binding security game from Section 2.3 applied to our scheme. Taking the following probabilities over the randomness of the game we have :

$$\begin{aligned}\Pr[\mathsf{G}(\mathcal{A}) \to 1] &= \Pr[\mathsf{G}(\mathcal{A}) \to 1 \wedge v_0, v_1 \text{ are in the same orbit}] \\ &\quad + \Pr[\mathsf{G}(\mathcal{A}) \to 1 \wedge v_0, v_1 \text{ are in different orbits}] \\ &\leq \Pr[v_0, v_1 \text{ are in the same orbit}] \\ &\quad + \Pr[\mathsf{G}(\mathcal{A}) \to 1 | v_0, v_1 \text{ are in different orbits}] \cdot p,\end{aligned}$$

where p is the probability defined by Property 3, and computed in Lemma 13. We have $\Pr[v_0, v_1 \text{ are in the same orbit}] = 1 - p$ by definition of p. Furthermore, $\Pr[\mathsf{G}(\mathcal{A}) \to 1 | v_0, v_1 \text{ are in different orbits}] = 0$ since otherwise if $\mathcal{A}$ wins G in that case, this means they return (A_0, B_0, C_0) and (A_1, B_1, C_1) such that

$$(A_0, B_0, C_0) \star v_0 = (A_1, B_1, C_1) \star v_1.$$

In particular this would mean that $(A_1^{-1}A_0, B_1^{-1}B_0, C_1^{-1}C_0) \star v_0 = v_1$ which is impossible since v_0, v_1 are in different orbits and thus no such isomorphism exists.

Overall we get

$$\Pr[\mathsf{G}(\mathcal{A}) \to 1] \leq (1-p) + 0 \cdot p = (1-p) = \mathrm{negl}(\lambda).$$

□

Remark 7. Notice that in the case where v_0, v_1 are indeed in the same orbit (which happens only with negligible probability), breaking the Binding problem amounts to solving the Computational Tensor Isomorphism problem (Problem 3) in a *random* orbit, hence remains plausibly hard.

Theorem 5 (Hiding). *Assuming the hardness of the dGA-IP problem (Definition 4), our commitment scheme is hiding.*

Proof. Consider G to be the hiding game from Section 2.3. Taking the following probabilities over the randomness of the game, we have :

$$\begin{aligned}\Pr[\mathsf{G}(\mathcal{A}) \to 1] &= \Pr[\mathsf{G}(\mathcal{A}) \to 1 \wedge v_0, v_1 \text{ are in the same orbit}] \\ &\quad + \Pr[\mathsf{G}(\mathcal{A}) \to 1 \wedge v_0, v_1 \text{ are in different orbits}] \\ &\leq \Pr[v_0, v_1 \text{ are in the same orbit}] \\ &\quad + \Pr[\mathsf{G}(\mathcal{A}) \to 1 | v_0, v_1 \text{ are in different orbits}] \cdot p,\end{aligned}$$

where p is the probability defined by Property 3. We have $\Pr[v_0, v_1 \text{ are in the same orbit}] = 1 - p = \mathrm{negl}(\lambda)$.
In the case where v_0, v_1 are in different orbits, one can refer to [20], Theorem 2. Roughly, given an adversary $\mathcal{B}$ against dGA-IP, that receives as input s, t, they instantiate two Hiding games with adversaries $\mathcal{A}_1, \mathcal{A}_2$ to which they respectively give s or t as input, and $\mathcal{B}$ returns 1 if the outputs of $\mathcal{A}_1$ and $\mathcal{A}_2$ are equal and 0 otherwise. They get that

$$\Pr[\mathsf{G}(\mathcal{A}) \to 1 | v_0, v_1 \text{ are in different orbits}] \leq \frac{1}{2} + 2\epsilon(\lambda)^2$$

where $\epsilon(\lambda)$ is the advantage in the dGA-IP game.

So overall we have

$$\begin{aligned}\Pr[\mathsf{G}(\mathcal{A}) \to 1] &\leq (1-p) + p \cdot (\frac{1}{2} + 2\epsilon(\lambda)^2) \\ &= \frac{1}{2}p + (1-p) + 2p\epsilon(\lambda)^2\end{aligned}$$

and by assumption both $\epsilon(\lambda)$ and $1-p$ are negligible. □

Remark 8. Consider the extension of our commitment scheme to a message space of $\{0, \dots N\}$ as described in Remark 5. Then the hiding and binding properties are derived in a straightforward way since $v_0, \dots v_N$ are in different orbits with probability at least $1 - \frac{N}{q^{n^3-3n^2}}$.

4.1 Cost comparison

The cost of generating the commitment in [20] is $O(n^3)$ multiplications and additions over $\mathbb{F}_q$ (the cost of matrix multiplication, ignoring asymptotic improvements). Due to our use of generic tensors, we have to perform $3n$ matrix multiplications to compute our commitment, bringing the cost to $O(n^4)$. The cost of multiplications over $\mathbb{F}_q$ is $O(\log q \log\log q)$. While this provides a slowdown compared to [20], it is still reasonable when compared to other cryptographic operations. In particular, we benefit from the fact that matrix multiplication already has existing low-level code and algorithms, including parallel algorithms. It is also worth noting that our loss in efficiency only occurs during the actual computation of the commitment. For any operation on the commitment itself or involving it (e.g zero-knowledge proofs) the cost of manipulating it would be the same for both our construction and that of [20] since they are both comprised of matrix multiplication on a (seemingly) random tensor.

MEDS [16], a NIST signature candidate, only uses two matrices A, B, but introduces *systematic form computation* to recover a third matrix, C. This allows a trade-off between computation time and size. Therefore, the cost of a single group action on a matrix code is the cost of matrix multiplication plus the systematic form computation. This trade-off could also be applied to the case of our commitment scheme, along with their other (future) group action optimizations.

4.2 On proofs of knowledge

To use commitment schemes within more advanced protocols such as verifiable secret sharing or zero-knowledge proofs for generic statements, it is useful to have non-interactive zero-knowledge proofs of knowledge of commitments. More precisely, one wishes to prove that they know the value to which a commitment c corresponds without giving away any information about it.

Let us consider the commitment scheme of Figure 6, extended to the message space $\{0, \dots N\}$. Given public parameters $\mathsf{pp} = \{t_0, \dots t_N\}$ consisting of $N+1$ random tensors from $\mathbf{V}$, we consider the relation

$$\mathcal{R} = \{(c, (i, (A, B, C))) \mid c = (A, B, C) \star t_i\},$$

where c is the statement and $(i, (A, B, C))$ is the witness. This means given c we wish to prove knowledge of the message i and randomness (A, B, C) to which c corresponds, without revealing anything about them. We briefly expose two ways of obtaining such a proof.

Using a variant of OR-proofs. A proof for $\mathcal{R}$ can be obtained by combining a variant of an OR-proof [19] and a proof for $\mathcal{R}_i = \{((c, t_i), (A, B, C))) \mid c = (A, B, C) \star t_i\}$, to show that one knows an isomorphism mapping c to a particular choice of one of the t_i. A proof for $\mathcal{R}$ can be obtained by adapting, for example, the graph isomorphism proof [26]. However, using OR-proofs increases the communication cost in the answer of the prover, and we will therefore prefer to build a direct proof for the full relation $\mathcal{R}$.

A direct proof for $\mathcal{R}$. We give in Figure 7 a sigma protocol that directly gives a proof for $\mathcal{R}$.

$\mathsf{Prover}_1(\mathsf{pp}, i, c, (A, B, C))$

$A', B', C' \xleftarrow{\$} \mathrm{GL}_n(q)^3$
$\sigma \xleftarrow{\$} S_n$
for $j = 1, \dots N$ **do**
    $d_j \leftarrow (A', B', C') \star t_{\sigma(j)}$
end for
$st \leftarrow (\mathsf{pp}, (A, B, C), (A', B', C'), \sigma)$
Send $(d_j)_{j=1}^N$ to Verifier
return st

$\mathsf{Prover}_2(st, \mathsf{ch})$

if $\mathsf{ch} = 0$ **then**
    $\mathsf{resp} \leftarrow ((A', B', C'), \sigma)$
    return resp
end if
$\mathsf{resp} \leftarrow (A(A')^{-1}, B(B')^{-1}, C(C')^{-1}), \sigma(i))$
return resp

$\mathsf{Verifier}_1(\mathsf{pp}, c, (d_j)_{j=1}^N)$

$\mathsf{ch} \xleftarrow{\$} \{0, 1\}$
$st \leftarrow (\mathsf{pp}, \mathsf{ch}, c, (d_j)_{j=1}^N), t_i)$
Send ch to Prover
return st

$\mathsf{Verifier}_2(st, \mathsf{resp})$

if $\mathsf{ch} = 0$ **then**
    $\mathsf{resp} \to ((\tilde{A}, \tilde{B}, \tilde{C}), \sigma)$
    return $\forall i, (\tilde{A}, \tilde{B}, \tilde{C}) \star t_{\sigma(i)} = d_i\big)$
end if
$\mathsf{resp} \to (\tilde{A}, \tilde{B}, \tilde{C}), i')$
return $\big((\tilde{A}, \tilde{B}, \tilde{C}) \star d_{i'} = c\big)$

Fig. 7. Proof system for $\mathcal{R}$

We sketch the extractor for special soundness and simulator for honest-verifier zero-knowledge below.

Special-soundness. Suppose we have two accepting transcripts with different challenges, $((d_j)_{j=1}^N, \mathsf{ch}_0, \mathsf{resp}_0)$ and $((d_j)_{j=1}^N, \mathsf{ch}_1, \mathsf{resp}_1)$. Without loss of generality, we assume $\mathsf{ch}_0 = 0$. Then we can build an extractor $\mathcal{E}$ which, given these transcripts as input, does the following : Recover (A', B', C') from resp_0 and $(A(A')^{-1}, B(B')^{-1}, C(C')^{-1})$ from resp_1. Compute (A, B, C) by multiplying the latter by the former. Recover i' from resp_0 and recover the correct $i = \sigma^{-1}(i')$. Then $(i, (A, B, C))$ is the witness.

Honest-verifier zero-knowledge. We now sketch the simulator. In the case $\mathsf{ch} = 0$, the simulator just follows the protocol honestly and will be producing an accepting transcript. In the $\mathsf{ch} = 1$ case, the simulator samples (A', B', C') and σ as in the protocol, then picks some index j and sets $d_j = ((A')^{-1}, (B')^{-1}, (C')^{-1}) \star c$. The other d_i for $i \neq j$ are computed according to the protocol. The distribution will remain computationally indistinguishable since (A', B', C') is random. The answer then consists of j and (A', B', C').

5 Conclusion

In this work, we study low rank points and stabilizers on unit tensors. We show how this information gives us the tools to break the hiding property of the bit

commitment scheme proposed in [20] at Asiacrypt 2023, that uses these unit tensors. Our attacks further allow to recover all the secret information used during the creation of the commitment. All of these algorithms run in polynomial time, which leaves the framework introduced in [20] with no concrete instantiation. Prior to this work, there was no evidence in the literature suggesting that unit tensors were not secure for use in cryptography.

With these attacks in mind, we propose a slightly different framework as well as a construction that makes use of *random* tensors. This allows us to build a statistically binding, computationally hiding commitment from non-transitive group actions. Finally, we complete this work by proposing a zero-knowledge proof of opening for our new commitment scheme.

At the moment, as pointed out by [20], the field of commitment schemes from non-transitive group actions is lacking protocols that are both post-quantum and non-interactive. We hope our new construction helps to fill this gap, and that our attacks can help to inform future cryptography designers on some of the limits of working with unit tensors. We leave any further investigation into a commitment scheme using the framework proposed in [20] as future work.

Supplementary Material

A Proofs

The proofs included here are done via a very computational approach, using core definitions. We encourage the reader who wants to familiarize themselves with these definitions to do them as exercise.

Lemma 5. *Let $\lambda_a, \lambda_b, \lambda_c \in \mathbb{F}_q$ be such that $\lambda_a\lambda_b\lambda_c = 1$. Then for all $v \in \mathbf{V}$, we have that $(\lambda_a I_n, \lambda_b I_n, \lambda_c I_n) \star v = v$.*

Proof. Let us write $v = \sum_{i,j,k=1}^{n} v(i,j,k) e_i \otimes e_j \otimes e_k$. We have

$$\begin{aligned}
(\lambda_a I_n, \lambda_b I_n, \lambda_c I_n) \star v &= \sum_{i,j,k=1}^{n} v(i,j,k)(\lambda_a I_n)e_i \otimes (\lambda_b I_n)e_j \otimes (\lambda_c I_n)e_k \\
&= \sum_{i,j,k=1}^{n} v(i,j,k)\lambda_a\lambda_b\lambda_c e_i \otimes e_j \otimes e_k \\
&= \sum_{i,j,k=1}^{n} v(i,j,k) e_i \otimes e_j \otimes e_k \text{ since } \lambda_a\lambda_b\lambda_c = 1 \\
&= v
\end{aligned}$$

□

Lemma 7. *Let $\sigma \in S_n$, then there exists P_σ such that $(P_\sigma, P_\sigma, P_\sigma) \star t_0 = t_0$, given by $P_\sigma = (p_{ij})_{i,j=1}^{n}$ and $p_{ij} = \mathbf{1}_{i=\sigma(j)}$.*

Proof. Let $t_0 = \sum_{s=1}^{n} e_s \otimes e_s \otimes e_s$, then we can express the action $(A, B, C) \star t_0$ (from (3.3)) as $(A, B, C) * (\sum_{s=1}^{n} e_s \otimes e_s \otimes e_s) = \sum_{i,j,k}^{n} \sum_{s=1}^{n} A_{is} B_{js} C_{ks} e_i \otimes e_j \otimes e_k$.

$$\begin{aligned}
(P_\sigma, P_\sigma, P_\sigma) \star t_0 &= \sum_{i,j,k=1}^{n} \sum_{s=1}^{n} (P_\sigma)_{is}(P_\sigma)_{js}(P_\sigma)_{ks} e_i \otimes e_j \otimes e_k \\
&= \sum_{i,j,k=1}^{n} \sum_{s=1}^{n} \mathbf{1}_{i=\sigma(s)} \mathbf{1}_{j=\sigma(s)} \mathbf{1}_{k=\sigma(s)} e_i \otimes e_j \otimes e_k \\
&= \sum_{i,j,k=1}^{n} \sum_{s=1}^{n} \mathbf{1}_{i=j=k} \mathbf{1}_{i=\sigma(s)} e_i \otimes e_j \otimes e_k \\
&= \sum_{m=1}^{n} \sum_{s=1}^{n} \mathbf{1}_{m=\sigma(s)} e_m \otimes e_m \otimes e_m \\
&= \sum_{m=1}^{n} e_m \otimes e_m \otimes e_m = t_0
\end{aligned}$$

since σ is a bijection. □

References

1. Alamati, N., Feo, L.D., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 411–439. Springer (2020)
2. Albrecht, M.R., Fenzi, G., Lapiha, O., Nguyen, N.K.: SLAP: succinct lattice-based polynomial commitments from standard assumptions. Cryptology ePrint Archive, Paper 2023/1469 (2023), https://eprint.iacr.org/2023/1469
3. Aragon, N., Gaborit, P., Hauteville, A., Tillich, J.: A new algorithm for solving the rank syndrome decoding problem. In: 2018 IEEE International Symposium on Information Theory, ISIT 2018, Vail, CO, USA, June 17-22, 2018. pp. 2421–2425. IEEE (2018). https://doi.org/10.1109/ISIT.2018.8437464
4. Bardet, M., Bros, M., Cabarcas, D., Gaborit, P., Perlner, R.A., Smith-Tone, D., Tillich, J., Verbel, J.A.: Improvements of algebraic attacks for solving the rank decoding and MinRank problems. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12491, pp. 507–536. Springer (2020). https://doi.org/10.1007/978-3-030-64837-4_17
5. Barenghi, A., Biasse, J., Persichetti, E., Santini, P.: On the computational hardness of the code equivalence problem in cryptography. Adv. Math. Commun. **17**(1), 23–55 (2023). https://doi.org/10.3934/AMC.2022064
6. Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S., Peikert, C.: More efficient commitments from structured lattice assumptions. In: Catalano, D., De Prisco, R. (eds.) Security and Cryptography for Networks. pp. 368–385. Springer International Publishing, Cham (2018)
7. Beullens, W.: Not enough LESS: An improved algorithm for solving code equivalence problems over Fq. In: International Conference on Selected Areas in Cryptography. pp. 387–403. Springer (2020)
8. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. Advances in Cryptology – ASIACRYPT 2019 pp. 227–247 (2019)
9. Bläser, M., Duong, D.H., Narayanan, A.K., Plantard, T., Qiao, Y., Sipasseuth, A., Tang, G.: The alteq signature scheme: Algorithm specifications and supporting documentation (2023), https://pqcalteq.github.io/ALTEQ_spec_2024.03.05.pdf
10. Bouillaguet, C., Fouque, P., Véber, A.: Graph-theoretic algorithms for the "isomorphism of polynomials" problem. In: Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7881, pp. 211–227. Springer (2013). https://doi.org/10.1007/978-3-642-38348-9_13, https://doi.org/10.1007/978-3-642-38348-9_13
11. Brassard, G., Yung, M.: One-way group actions. In: Proceedings of the 10th Annual International Cryptology Conference on Advances in Cryptology. p. 94–107. CRYPTO '90, Springer-Verlag, Berlin, Heidelberg (1990)
12. Bürgisser, P., Ikenmeyer, C.: Geometric complexity theory and tensor rank. CoRR **abs/1011.1350** (2010), http://arxiv.org/abs/1011.1350

13. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S.D. (eds.) Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III. Lecture Notes in Computer Science, vol. 11274, pp. 395–427. Springer (2018). https://doi.org/10.1007/978-3-030-03332-3_15
14. Chen, M., Lai, Y., Laval, A., Marco, L., Petit, C.: Malleable commitments from group actions and zero-knowledge proofs for circuits based on isogenies. In: Chattopadhyay, A., Bhasin, S., Picek, S., Rebeiro, C. (eds.) Progress in Cryptology - INDOCRYPT 2023 - 24th International Conference on Cryptology in India, Goa, India, December 10-13, 2023, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14459, pp. 221–243. Springer (2023). https://doi.org/10.1007/978-3-031-56232-7_11, https://doi.org/10.1007/978-3-031-56232-7_11
15. Chou, T., Niederhagen, R., Persichetti, E., Ran, L., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Matrix equivalence digital signature (2023), https://www.meds-pqc.org/spec/MEDS-2023-07-26.pdf
16. Chou, T., Niederhagen, R., Persichetti, E., Randrianarisoa, T.H., Reijnders, K., Samardjiska, S., Trimoska, M.: Take your MEDS: digital signatures from matrix code equivalence. In: Mrabet, N.E., Feo, L.D., Duquesne, S. (eds.) Progress in Cryptology - AFRICACRYPT 2023 - 14th International Conference on Cryptology in Africa, Sousse, Tunisia, July 19-21, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14064, pp. 28–52. Springer (2023). https://doi.org/10.1007/978-3-031-37679-5_2
17. Couveignes, J.M.: Hard homogeneous spaces. Cryptology ePrint Archive, Paper 2006/291 (2006), https://eprint.iacr.org/2006/291
18. Couvreur, A., Debris-Alazard, T., Gaborit, P.: On the hardness of code equivalence problems in rank metric. CoRR **abs/2011.04611** (2020), https://arxiv.org/abs/2011.04611
19. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) Advances in Cryptology — CRYPTO '94. pp. 174–187. Springer Berlin Heidelberg, Berlin, Heidelberg (1994)
20. D'Alconzo, G., Flamini, A., Gangemi, A.: Non-interactive commitment from non-transitive group actions. Advances in Cryptology – ASIACRYPT 2023 p. 723 (2023)
21. Ducas, L., van Woerden, W.P.J.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 643–673. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_23
22. Fenzi, G., Moghaddas, H., Nguyen, N.K.: Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. Cryptology ePrint Archive, Paper 2023/846 (2023), https://eprint.iacr.org/2023/846
23. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings. Lecture Notes in Com-

puter Science, vol. 263, pp. 186–194. Springer (1986). https://doi.org/10.1007/3-540-47721-7_12, https://doi.org/10.1007/3-540-47721-7_12
24. Frederiksen, T.K., Pinkas, B., Yanai, A.: Committed MPC - maliciously secure multiparty computation from homomorphic commitments. In: Abdalla, M., Dahab, R. (eds.) Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10769, pp. 587–619. Springer (2018). https://doi.org/10.1007/978-3-319-76578-5_20
25. Fulman, J., Goldstein, L.: Stein's method and the rank distribution of random matrices over finite fields. The Annals of Probability **43**(3) (May 2015). https://doi.org/10.1214/13-aop889
26. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. J. ACM **38**(3), 690–728 (jul 1991). https://doi.org/10.1145/116825.116852
27. Grochow, J.A., Qiao, Y.: On the complexity of isomorphism problems for tensors, groups, and polynomials I: tensor isomorphism-completeness. In: Lee, J.R. (ed.) 12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference. LIPIcs, vol. 185, pp. 31:1–31:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPICS.ITCS.2021.31, https://doi.org/10.4230/LIPIcs.ITCS.2021.31
28. Grochow, J.A., Qiao, Y., Tang, G.: Average-case algorithms for testing isomorphism of polynomials, algebras, and multilinear forms. In: Bläser, M., Monmege, B. (eds.) 38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference). LIPIcs, vol. 187, pp. 38:1–38:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPICS.STACS.2021.38, https://doi.org/10.4230/LIPIcs.STACS.2021.38
29. Håstad, J.: Tensor rank is NP-complete. J. Algorithms **11**(4), 644–654 (1990). https://doi.org/10.1016/0196-6774(90)90014-6
30. Hillar, C.J., Lim, L.: Most tensor problems are NP-hard. J. ACM **60**(6), 45:1–45:39 (2013). https://doi.org/10.1145/2512329
31. Jain, A., Krenn, S., Pietrzak, K., Tentes, A.: Commitments and efficient zero-knowledge proofs from learning parity with noise. In: Wang, X., Sako, K. (eds.) Advances in Cryptology – ASIACRYPT 2012. pp. 663–680. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
32. Ji, Z., Qiao, Y., Song, F., Yun, A.: General linear group action on tensors: A candidate for post-quantum cryptography. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part I. Lecture Notes in Computer Science, vol. 11891, pp. 251–281. Springer (2019). https://doi.org/10.1007/978-3-030-36030-6_11
33. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures (extended abstract). In: Jr., B.S.K. (ed.) Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings. Lecture Notes in Computer Science, vol. 1294, pp. 150–164. Springer (1997). https://doi.org/10.1007/BFB0052233
34. Kipnis, A., Shamir, A.: Cryptanalysis of the oil & vinegar signature scheme. In: Krawczyk, H. (ed.) Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings. Lecture Notes in Computer Science, vol. 1462, pp. 257–266.

Springer (1998). https://doi.org/10.1007/BFB0055733, https://doi.org/10.1007/BFb0055733
35. Leon, J.: Computing automorphism groups of error-correcting codes. IEEE Transactions on Information Theory **28**(3), 496–511 (1982)
36. Lyubashevsky, V., Nguyen, N.K., Seiler, G.: Shorter lattice-based zero-knowledge proofs via one-time commitments. In: Garay, J.A. (ed.) Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12710, pp. 215–241. Springer (2021). https://doi.org/10.1007/978-3-030-75245-3_9
37. Morozov, K., Roy, P.S., Sakurai, K.: On unconditionally binding code-based commitment schemes. In: Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication. IMCOM '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3022227.3022327, https://doi.org/10.1145/3022227.3022327
38. Naor, M.: Bit commitment using pseudorandomness. J. Cryptol. **4**(2), 151–158 (1991). https://doi.org/10.1007/BF00196774
39. Narayanan, A.K., Qiao, Y., Tang, G.: Algorithms for matrix code and alternating trilinear form equivalences via new isomorphism invariants. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14653, pp. 160–187. Springer (2024). https://doi.org/10.1007/978-3-031-58734-4_6, https://doi.org/10.1007/978-3-031-58734-4_6
40. Ostrovsky, R., Persiano, G., Visconti, I.: Simulation-based concurrent non-malleable commitments and decommitments. In: Reingold, O. (ed.) Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5444, pp. 91–108. Springer (2009). https://doi.org/10.1007/978-3-642-00457-5_7
41. Patarin, J.: Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In: Maurer, U.M. (ed.) Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding. Lecture Notes in Computer Science, vol. 1070, pp. 33–48. Springer (1996). https://doi.org/10.1007/3-540-68339-9_4
42. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings. Lecture Notes in Computer Science, vol. 576, pp. 129–140. Springer (1991). https://doi.org/10.1007/3-540-46766-1_9
43. Poelstra, A., Back, A., Friedenbach, M., Maxwell, G., Wuille, P.: Confidential assets. In: Zohar, A., Eyal, I., Teague, V., Clark, J., Bracciali, A., Pintore, F., Sala, M. (eds.) Financial Cryptography and Data Security - FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, March 2, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10958, pp. 43–63. Springer (2018). https://doi.org/10.1007/978-3-662-58820-8_4
44. Reijnders, K., Samardjiska, S., Trimoska, M.: Hardness estimates of the code equivalence problem in the rank metric. IACR Cryptol. ePrint Arch. p. 276 (2022), https://eprint.iacr.org/2022/276

45. Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Paper 2006/145 (2006), https://eprint.iacr.org/2006/145
46. Schaefer, M., Stefankovic, D.: The complexity of tensor rank. Theory Comput. Syst. **62**(5), 1161–1174 (2018). https://doi.org/10.1007/S00224-017-9800-Y
47. Sterner, B.: Commitment schemes from supersingular elliptic curve isogeny graphs. Journal of Mathematical Cryptology (2021)
48. Tang, G., Duong, D.H., Joux, A., Plantard, T., Qiao, Y., Susilo, W.: Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 582–612. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_21

6 The Vectorization Problem with shifted inputs

Publication data. Paul Frixons, Valerie Gilchrist, Péter Kutas, Simon-Philipp Merz, Christophe Petit, Lam Pham. Another Look at the Quantum Security of the Vectorization Problem with Shifted Inputs. In submission.

My Contribution. All six authors contributed equally to the theoretical development and writing.

Another Look at the Quantum Security of the Vectorization Problem with Shifted Inputs

Paul Frixons[1], Valerie Gilchrist[1], Péter Kutas[2,3], Simon-Philipp Merz[4], Christophe Petit[1,3], Lam L. Pham[5]

[1] Université Libre de Bruxelles, Belgium
[2] Eötvös Loránd University, Hungary
[3] University of Birmingham, United Kingdom
[4] ETH Zürich, Switzerland
[5] Ghent University, Belgium

Abstract. Cryptographic group actions provide a basis for simple post-quantum generalizations of many cryptographic protocols based on the discrete logarithm problem (DLP). However, many advanced group action-based protocols do not solely rely on the core group action problem (the so-called vectorization problem), but also on *variants* of this problem, to either improve efficiency or enable new functionalities. For example, the security of the CSI-SharK threshold signature protocol relies on the hardness of the *Vectorization Problem with Shifted Inputs* where (in DLP formalism) the adversary not only receives g and g^x, but also g^{x^c} for multiple known values of c. A natural open question is whether the additional data allows adversaries to solve the underlying problem more efficiently.

We revisit the concrete quantum security of this problem. We start from a quantum *multiple* hidden shift algorithm of Childs and van Dam, which to the best of our knowledge was never applied in cryptography before. We describe and analyze a variant of this algorithm, and we specify and analyze all its subroutines to provide concrete complexity estimates.

We then apply our analysis to the CSI-SharK protocol. In prior analyses based on Kuperberg's algorithms, group action evaluations contributed to a significant part of the overall T-gate cost. For CSI-SharK's suggested parameters, our new approach requires significantly fewer calls to

* Authors listed in alphabetical order: see https://www.ams.org/profession/leaders/CultureStatement04.pdf. Valerie Gilchrist is supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium; Simon-Philipp Merz is supported by DFG through a Walter Benjamin Fellowship and the Zurich Information Security and Privacy Center (ZISC); Christophe Petit and Péter Kutas are partly supported by EPSRC through grant number EP/V011324/1. Péter Kutas was supported by the Ministry of Culture and Innovation and the National Research, Development, and Innovation Office within the Quantum Information National Laboratory of Hungary (Grant No. 2022-2.1.1-NL-2022-00004) and by the grant "EXCELLENCE-151343". Péter Kutas is also supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. Christophe Petit is also supported by FRS-FNRS grant number T.0238.25. Lam Pham is supported by the FWO and the F.R.S.-FNRS under the Excellence Of Science (EOS) programme (project ID 40007542).

Date of this document: 2025-11-03.

the group action evaluation subroutine, leading to significant complexity improvements overall. We describe two instances of our approach, one minimizing the T-gate complexity, and the other one keeping qubit requirements small, both of them resulting in significant complexity improvements over previous works. More generally, we quantify the quantum security degradation resulting from additional published data in the CSI-SharK protocol.

1 Introduction

The looming threat of large-scale error-tolerant quantum computers motivates the development of new forms of cryptography based on alternative "hard problems" with the potential to also withstand adversaries with such machines. Following a massive effort from the cryptography community, we now have good candidates for basic primitives such as signatures and key encapsulation mechanisms. However, realizing advanced protocols based on many of the new problems remains a daunting task.

The inversion of cryptographic group actions [5], the so-called *vectorization problem*, provides a natural generalization to the classical discrete logarithm problem (DLP) which in turns leads to relatively easy adaptations of many DLP-based protocols. Currently, the most promising candidates for post-quantum group actions replace the classical DLP in the Diffie-Hellman protocol with an isogeny computation, e.g. [28, 33, 40]. One of these candidates, CSIDH [28], has given rise to several other protocols including signatures [14], ring signatures [47], threshold signatures [7, 41], identification protocols [8], and many more. These more advanced protocols can also be adapted to use other group actions.

Protocols based on isogeny group actions benefit from relatively small key sizes. However, before they can replace their classical counterparts, they must withstand quantum cryptanalysis, and their concrete quantum security needs to be understood better. So far, the main quantum attacks on CSIDH and its variants are based on (variations of) Kuperberg's algorithm [13, 23, 52, 54, 63]. In particular, Peikert provided quantum complexity estimates to break CSIDH [63].

While our understanding of the quantum security of the vectorization problem for CSIDH has greatly improved recently, several advanced protocols based on CSIDH rely on *variants* of this problem. Indeed, these variants can provide protocols with crucial additional functionality and enhanced efficiency. We note that this approach is reminiscent of a prior similar trend with DLP-based protocols (i.e. 35 "DLP variants" are listed in [11]). As was the case with some DLP variants [50], one may wonder whether all vectorization problem variants used in CSIDH-based cryptographic protocols are equally hard. For example, the Diffie-Hellman Inversion problem asks to compute g^{-x} given access to g, g^x.

While this problem remains hard in some settings, the CSIDH group action is an example where this problem becomes trivial by computing the quadratic twist.

In this paper, we focus on the *Vectorization Problem with Shifted Inputs* underlying the security of the CSI-SharK and BCP protocols [7, 8]. In DLP formalism, this variant provides an adversary not only with two group elements g and g^x, but also with several other pairs (c, g^{x^c}). An algorithm to solve this problem for c in an interval $[0, M]$ would also solve the M-Diffie-Hellman inversion problem and the M-Strong Diffie-Hellman problem, which underlie the security of various DLP-based protocols [17–20, 36]. If the interval is punctured once, then it solves the $(M/2)$-Diffie-Hellman Exponent problem, used in protocols such as [21, 42, 51]. The group action version of this problem is also considered in [38], where it is proven secure in the general group action model.

So far, the only cryptanalytic work leveraging the extra information provided in the Vectorization Problem with Shifted Inputs is the classical attack of Kim [49], which adapts Cheon's algorithm [29] to the isogeny group action setting. This attack affects instances where a shift g^{x^c} is published such that c divides the order of the class group, though such instances are specifically avoided in the CSI-SharK and BCP key generation algorithms. Apart from this restriction, the security analysis and parameter selection in [7, 8] are solely based on CSIDH. In particular, they ignore additional information provided in the Vectorization Problem with Shifted Inputs, and the protocols' quantum security is assumed to be equivalent to CSIDH's quantum security.

Contributions. In this work, we introduce a variant of a quantum *multiple* hidden shift algorithm from Childs and van Dam (CvD) [31] to the Vectorization Problem with Shifted Inputs. We carefully select subroutines to be used in the algorithm, and we discuss different trade-offs between them. Among these subroutines is a knapsack problem, which we reduce to a variant of the Closest Vector Problem (CVP) in the ℓ^∞ norm. The literature on CVP algorithms for the ℓ^∞ norm is much sparser than for the Euclidean norm. We consider two approaches to solve this problem, respectively based on enumeration and sieving. We propose concrete algorithms taking inspiration from existing ones, but with crucial optimizations taking into account the context of our application. We also provide concrete (not asymptotic) complexity and memory estimates for all of these subroutines, filling a gap in the literature.

The original Childs–van Dam algorithm required to find all solutions to the knapsack subroutine; our variant relaxes this requirement to accommodate a larger class of CVP solvers, and is of independent interest.

Our results can be applied to any group action, but we particularly focus on the isogeny-based group action underlying CSIDH since it is currently the most widely studied example of a post-quantum group action in cryptography.

As a concrete application, we use our analysis to reevaluate the security of the CSI-SharK and BCP protocols [7, 8], and we show that they fall short of the claimed security level. For CSIDH-512 parameters and the suggested 2^{12} curves per public key, we estimate that CSI-SharK and BCP can be broken

using between $2^{50.8}$ and $2^{56.7}$ T-gates depending on the subroutines used and whether [13] or [23] is used as the cost model for the group action evaluation[6].

In comparison, Peikert estimated the cost for the cryptanalysis of CSIDH-512 (i.e. for the single curve per public key instance) to require significantly more resources, i.e. between 2^{59} to 2^{72} T-gates. More generally, we quantify how the quantum security of the Vectorization Problem with Shifted Inputs degrades when the number of curves in the public key increases, via concrete estimates for CSIDH-512. A summary of our results for different parameters is shown in Tab. 1.

group action cost estimate		Peikert Alg. [63, Sect. 4.1]		M	k	Alg. 2 + sieving		Alg. 2 + enum.	
T-gates	ancillas	T-gates	QRACM			T-gates	ancillas	T-gates	ancillas
(from [13])				2^8	29	$\mathbf{2^{51.7}}$	$2^{48.4}$	$2^{72.9}$	$2^{17.7}$
$2^{43.8}$	2^{40}	$2^{59.8}$ to $2^{62.8}$	2^{40} to 2^{32}	2^{12}	20	$\mathbf{2^{50.8}}$	2^{40}	$\mathbf{2^{54.9}}$	$2^{16.7}$
				2^{16}	16	$\mathbf{2^{50.5}}$	2^{40}	$\mathbf{2^{51.7}}$	$2^{16.1}$
				2^{20}	13	$\mathbf{2^{50.2}}$	2^{40}	$\mathbf{2^{47.5}}$	$2^{15.5}$
(from [23])				2^8	29	$\mathbf{2^{60}}$	$2^{48.4}$	$2^{72.9}$	$2^{17.7}$
$2^{52.4}$	$2^{15.3}$	$2^{68.4}$ to $2^{71.4}$	2^{40} to 2^{32}	2^{12}	20	$\mathbf{2^{59.4}}$	$2^{38.9}$	$\mathbf{2^{56.7}}$	$2^{16.7}$
				2^{16}	16	$\mathbf{2^{59.1}}$	2^{35}	$\mathbf{2^{56.4}}$	$2^{16.1}$
				2^{20}	13	$\mathbf{2^{58.8}}$	$2^{33.0}$	$\mathbf{2^{56.1}}$	$2^{15.5}$

Table 1. Using CSIDH-512 parameter sets, we list complexities in number of T-gates and memory required of the Childs–van Dam algorithm when given access to shifts of the form $g^{c \cdot z}$ for $c \in [-M, M]$. We compare against Peikert's algorithm, based on Kuperberg's collimation sieve, whose (oracle calls, QRACM) usage ranges from $(2^{16}, 2^{40})$ to $(2^{19}, 2^{32})$. Improved T-gate counts are bolded.

Note that the best T-gate complexities are obtained with the sieving approach for lower values of M. On the other hand, as M increases, the enumeration approach becomes the better choice to minimize the gate cost. Neither approach requires any QRAQM or QRACM, and using enumeration requires significantly fewer qubits to implement. Overall, the quantum security of the schemes clearly degrades with increasing M.

Discussion and further work. The complexity improvements we obtain for CSI-SharK and BCP are interesting on their own, but our concrete analysis of Childs–van Dam's algorithm may also find further applications in cryptography, including, but not limited to isogeny-based cryptography.

Our complexity estimates heavily depend on the subroutines selected, in particular the ones for solving the knapsack problem. There is of course considerable literature on knapsack problems, but we found that "off-the-shelf" algorithms

[6] Note that [13] only targets T-gate counts whereas [23] (in its latest eprint version) also attempts to limit the number of qubits.

were not suited to our particular setting. We did our best to select and adapt existing algorithms, but we still consider it likely that these adaptations remain suboptimal. Our complexity estimates, while already improving on the state-of-the-art, should therefore still be considered as upper bounds.

We conclude the paper with potential improvements and extensions of this analysis. Various hard problems similar to the Vectorization Problem with Shifted Inputs have appeared in the literature; as evidenced by our work (and previous work on DLP variants), their actual hardness may not follow from the hardness of CSIDH. We encourage the community to study their concrete quantum security beyond a mere application of Kuperberg's algorithms and their variants.

Outline. In Sect. 2, we recall details of CSIDH, its state-of-the-art quantum cryptanalysis, as well as the CSI-SharK signature scheme and its hardness assumption. We describe and analyze our variant of Childs–van Dam's quantum algorithm in Sect. 3. In Sect. 4, we show how a knapsack problem appearing in the Childs–van Dam algorithm can be formulated as a closest vector problem in the ℓ^∞ norm. We solve this problem using enumeration and sieving in Sects. 5 and 6, respectively. Finally, we give concrete complexity estimates for running the attack on CSI-SharK parameters and investigate how the attack improves for larger number of shifts in Sect. 7. We conclude the paper and discuss several avenues of future work in Sect. 8.

2 Preliminaries

In this section, we recall the CSIDH group action, known quantum cryptanalysis approaches against it and the CSI-SharK protocol.

2.1 CSIDH group action

We briefly outline some key concepts about CSIDH and its underlying group action. For a more in-depth exposition of these topics, we refer to either CSIDH [28] or the textbook from Cox [32, Sect. 7].

Consider the set of supersingular elliptic curves defined over $\mathbb{F}_p$ whose $\mathbb{F}_p$-rational endomorphism ring is a fixed order $\mathcal{O}$ in the quadratic imaginary field $K = \mathbb{Q}(\sqrt{\Delta})$. Here Δ means the discriminant of the Frobenius characteristic polynomial. We denote this set of elliptic curves by $\mathcal{E}\ell\ell_p(\mathcal{O})$. Then, the *ideal class group* of $\mathcal{O}$ is the quotient of invertible fractional ideals, $I(\mathcal{O})$, and principal fractional ideals, $P(\mathcal{O})$, which we denote by $\mathrm{cl}(\mathcal{O}) = I(\mathcal{O})/P(\mathcal{O})$. Let $[\mathfrak{a}] \in \mathrm{cl}(\mathcal{O})$ be an ideal class. Then $[\mathfrak{a}]$ acts on any elliptic curve $E \in \mathcal{E}\ell\ell_p(\mathcal{O})$ via an isogeny

$$\varphi_{\mathfrak{a}} : E \to E/\mathfrak{a},$$

where $\ker(\varphi_{\mathfrak{a}}) = \bigcap_{\alpha \in \mathfrak{a}} \ker(\alpha)$. This gives rise to the following free and transitive group action

$$\mathrm{cl}(\mathcal{O}) \times \mathcal{E}\ell\ell_p(\mathcal{O}) \to \mathcal{E}\ell\ell_p(\mathcal{O}),$$

$$[\mathfrak{a}] \star E \mapsto E/\mathfrak{a}.$$

For brevity, we drop the $\star$ going forward and simply denote the group action as $[\mathfrak{a}]E$. Equipped with this isogeny group action, the CSIDH protocol naturally follows the structure of a Diffie-Hellman key exchange [28].

2.2 Quantum cryptanalysis of CSIDH

The hidden shift problem. The works of Childs, Jao, and Soukharev [30] and Biasse, Jao, and Sankar [15] showed how to frame the hard problem underlying CSIDH as a hidden shift problem. Recall that given two curves $E, E' \in \mathcal{E}\ell\ell_p(\mathcal{O})$ we would like to find an ideal class $[\mathfrak{a}] \in \mathrm{cl}(\mathcal{O})$ such that $E' = [\mathfrak{a}]E$. Then we can define two functions f_0, f_1 as $f_0 : [\mathfrak{b}] \mapsto [\mathfrak{b}]E$ and $f_1 : [\mathfrak{b}] \mapsto [\mathfrak{b}]([\mathfrak{a}]E)$. Notice that f_0 is injective and $f_1(\mathfrak{b}) = f_0(\mathfrak{b}\mathfrak{a})$, i.e. the functions f_0 and f_1 are equal up to a shift $\mathfrak{a}$. This means we can apply quantum hidden shift algorithms, like that of Kuperberg [52], to recover the secret $[\mathfrak{a}]$ in subexponential time.

Kuperberg's algorithm uses a complexity of $2^{O(\sqrt{\log N})}$ where N is the size of the group, and works in any finite abelian group. An algorithm from Regev [64] gives a space improvement in terms of both qubits and QRAM, at the cost of a slower run time, but restricts to the $\mathbb{Z}_{2^n}$ case. Later, Kuperberg released a second algorithm in [54] that also benefited from a space improvement but this time in the general setting, called the *collimation sieve*. Peikert [63] built an attack on the CSIDH-512 parameter set using the collimation sieve. He estimates that his attack requires between 2^{38} and 2^{47} T-gates, and an additional 2^{14} to 2^{19} calls to an oracle computing the group action. Note that any of these algorithms require access to the group action oracle, but the exact cost of such an oracle depends on memory constraints.

Quantum costs of computing the group action. With the ongoing debate on the cost of QRAM, estimating the concrete cost of quantum algorithms can be challenging: indeed it is not yet clear which cost metric should be adopted. As a result, the literature includes several estimates for group action evaluation targeting different cost metrics.

Bernstein, Lange, Martindale, and Panny [13] focus on giving in-depth analyses of the isogeny-related computations such as finding a point of particular order on an elliptic curve and computing ℓ-isogenous curves, using classical circuits. They choose to minimize the number of non-linear bit operations (thus minimizing Toffoli and T-gates) at the cost of more qubits. The analysis from Bonnetain and Schrottenloher [23, Tab. 3] focuses on reducing the number of qubits necessary in their algorithm. They present several trade-offs between three quantum abelian hidden shift algorithms, and focus on tweaking these quantum algorithms to gain improvements (as opposed to tweaking the isogeny algorithms as in [13]). We summarize the cost estimates for a single CSIDH-512 group action evaluation in Tab. 2. Note that [13] uses non-linear bit operations as their cost metric, so we convert this to Toffoli gates and ancilla qubits using the Bennett conversion [12], as was suggested by the authors [13, App. A.4].

Group action cost estimate	Toffoli gates	T-gates	ancilla qubits
[13, Alg. 7.1]	2^{41}	$2^{43.8}$	2^{40}
[23, Table 3]	$2^{49.6}$	$2^{52.4}$	$2^{15.3}$

Table 2. Resource estimates for running one CSIDH group action evaluation on the CSIDH-512 parameter set.

Remark 2.1. The estimates from Tab. 2 were published in 2019 and 2020. Since then, several new tools in isogeny-based cryptography have been developed that modify the cost of isogeny computations and (classical) group action evaluation. These developments would likely improve on some of these resource estimates. Assessing the exact improvement is highly non-trivial and remains outside the scope of this work. Instead, we purposely conduct our complexity analyses modularly so that new resource estimates may be easily updated.

QRAM costs. Quantum random-access memory (QRAM) is a circuit that allows a quantum algorithm to access some data stored in memory where the address is itself a quantum state. The data being stored could itself be classical, in which case we refer to it as quantum random-access *classical* memory (QRACM), or the data could be quantum, in which case it is called quantum random-access *quantum* memory (QRAQM). Kuperberg [53] suggests that QRAQM is strictly more resource intensive to implement because of the need to maintain quantum states in both memory and access processes. The exact difference in resource use is difficult to quantify concretely. In [45], Jaques and Rattew give a survey of the current state-of-the-art for QRACM and QRAQM constructions. They suggest using circuit QRAM for both, which means using something like the bucket brigade circuit from [60].

In this work, we choose to construct circuits that avoid the use of either QRACM or QRAQM, using only ancilla qubits, albeit sometimes at a small cost to the T-gate complexity (e.g. an overhead factor of around $2^{2.7}$). The main point of comparison to our algorithm will be the analysis from Peikert [63] that gives a concrete complexity analysis of Kuperberg's algorithm when used to attack an instance of CSIDH-512. In [63], QRACM is central to the manipulation of vectors and avoiding QRACM would mean having to change every computation for every element of every list, which seems to add an overhead factor of at least 2^{32} to the time complexity. Reconstructing the algorithm to avoid QRACM is thus a highly non-trivial task, and outside the scope of this work.

2.3 CSI-SharK

CSI-SharK [7] is an isogeny-based threshold signature scheme that was adapted from CSI-FiSh [14] and the protocol from Baghery, Cozzo, and Pedersen (BCP) [8], i.e. multiple curves are constructed from a single secret. Here, we first briefly recall the CSI-FiSh sigma protocol, and then explain how it was used to construct CSI-SharK.

CSI-FiSh sigma protocol. In [14], Beullens, Kleinjung, and Vercauteren build upon the ideas of Stolbunov to obtain a signature scheme for the CSIDH group action. The scheme starts from a standard proof of knowledge where one begins by fixing the starting curve to $E_0 : y^2 = x^3 + x$. Then the prover samples their secret, $\mathfrak{a} \in \mathrm{cl}(\mathcal{O})$, and computes their public key $E_A = [\mathfrak{a}]E_0$. To prove knowledge of their secret key to a verifier, they first send a commitment $E_1 = [\mathfrak{b}]E_0$. The verifier replies with a challenge $c \in \{0, 1\}$. If $c = 0$ then the prover sends $r = \mathfrak{b}$, and the verifier checks that $E_1 = [r]E_0$. Otherwise when $c = 1$, the prover sends $r = [\mathfrak{b}\mathfrak{a}^{-1}]$, and the verifier checks that $E_1 = [r]E_A$. The isogeny diagram for this scheme is outlined in Fig. 1.

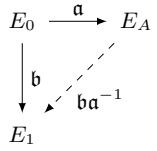


Fig. 1. The isogeny diagram related to the CSI-FiSh sigma protocol.

Let $\mathfrak{g}$ be an element of a class group generating a sufficiently large subgroup of cardinality N. Then CSI-SharK uses public keys of the form

$$\{([\mathfrak{g}^{c_i \cdot z}]E_0,\ c_i)\}_{i=0}^{M},$$

where the $\{c_i\}_{i=0}^{M} \subset \mathbb{Z}_N$ are known, and $z \in \mathbb{Z}_N$ is the secret key. The set $\{c_i\}_{i=0}^{M}$ is required to be a (super)*exceptional set*. That is, their pairwise differences (and sums) must be invertible modulo N. This property is essential for proving soundness. In particular, since c_0 is always chosen to be 0, this means that no c_i divides N. This prevents attackers from using an algorithm like Cheon's [29] to improve a baby-step giant-step approach to recover the secret.

The choice of $\{c_i\}_{i=0}^{M}$ is thus important. In BCP, the c_i are chosen to be necessarily smaller than the smallest prime factor of N. If M is larger than the smallest factor, they suggest restricting to a subgroup $\mathbb{Z}_{N'}$ whose smallest prime factor is larger than M. The most natural choice, as proposed by the authors in BCP [8, Footnote 1], is the consecutive list of integers $\{0, \ldots, M\}$. They list parameter sets for $M \in \{2^1, 2^2, 2^5, 2^8, 2^{10}, 2^{12}, 2^{15}, 2^{18}\}$ [8, Tab. 1]. CSI-SharK [7, Tab. 2] suggests taking $M \in \{2^4, 2^8, 2^{12}\}$.

Both CSI-SharK and BCP rely on the hardness of the following problem.

Problem 2.2 (Vectorization Problem with Shifted Inputs). Let $E \in \mathcal{E}\ell\ell_p(\mathcal{O})$. Given the pairs $(c_i, [\mathfrak{g}^{c_i z}]E)_{i=0}^{M}$, where $\mathfrak{g}$ is fixed and $\mathbb{C}_M = \{c_0 = 0, c_1 = 1, c_2, \ldots, c_M\}$ is an exceptional set, find $z \in \mathbb{Z}_N$.

Whenever the starting curve E_0 has j-invariant 1728, we can compute the curve $E_i^t = [\mathfrak{g}^{-c_i z}]E_0$ from $E_i = [\mathfrak{g}^{c_i z}]E_0$ using the quadratic twist. Though this

starting curve is not explicitly stated in [7], several of their optimisations take advantage of this easy twisting operation. As such, we may assume this starting curve implicitly. Note this gives us information about $2M+1$ curves, instead of the $M+1$ curves in the public key. For example, one of the proposed parameter sets in CSI-SharK uses $M = 2^{12}$. Thus, if the $\{c_i\}_{i=0}^{M}$ are consecutive integers starting at 0, including the twists, it means we have access to the following $2^{13}+1$ curves

$$\{[\mathfrak{g}^{cz}]E_0 : c \in [-2^{12}, 2^{12}]\}.$$

Problem 2.2 was also considered in [38], where the authors standardize some notions and problems related to group actions and provide reductions between some of them. Currently, the state-of-the-art security analysis on Problem 2.2 is to use Kuperberg's algorithm on one single instance. The best known classical attack, described in CSIDH, is a meet-in-the-middle key search, also run on one single instance. These approaches fail to make use of any of the additional information provided.

3 A modified version of the Childs–van Dam algorithm

In this section we recall the Childs–van Dam algorithm [31] and describe a modified version which will be the main quantum tool used in our attack.

3.1 The Algorithm

The algorithm from Childs and van Dam [31] aims at solving an instance of the *generalized hidden shift* problem.

Definition 3.1 (Generalized Hidden Shift). *For integers M, N, and a finite set S, let $f : \{-M, \dots, M\} \times \mathbb{Z}_N \to S$ be a function satisfying the following:*
(a) for fixed b, the map $f(b, \cdot) : \mathbb{Z}_N \to S$ is injective;
(b) $f(b, x) = f(b+1, x+z)$ for some fixed $z \in \mathbb{Z}_N$, all $b \in [-M, \dots, M-1]$ and all x.
The generalized hidden shift problem *asks to recover z given oracle access to f.*

In particular, (a) and (b) of Def. 3.1 imply that $f(b, x) = f(0, x - bz)$.

Note that we took $\{-M, \dots, M\} \times \mathbb{Z}_N$ as the domain of f, whereas related literature often defines the problem on the domain $\{0, \dots, M'-1\} \times \mathbb{Z}_N$ for some M'. These are clearly equivalent problems after relabeling the interval.

When $M' = 2$, Def. 3.1 is the standard *hidden shift* problem, and when $M' = N$ it is the *hidden subgroup* problem trying to find $\ker(f) = \langle (1, z) \rangle \subset \mathbb{Z}_N^2$.

Retrieving the secret key z in CSI-SharK can be seen as solving a generalized hidden shift problem for the function f sending $(b, x) \in \{-M, \dots, M\} \times \mathbb{Z}_N$ to $f(b, x) = [\mathfrak{g}^x]E_{-b}$. Indeed we have

$$f(b, x) = [\mathfrak{g}^x]E_{-b} = [\mathfrak{g}^x][\mathfrak{g}^{(-b)z}]E_0 = [\mathfrak{g}^{x-bz}]E_0 = f(0, x - bz),$$

hence $f(b,x) = f(b+1, x+z)$ as required.

We adapt Childs and van Dam's algorithm, outline our approach in Alg. 2, and provide further explanation in what follows. Throughout, we denote the logarithm in base 2 by "log" and the natural logarithm by "ln".

Setup. For the rest of this section, we denote by O_f the operator defined by $O_f(|b,x,0\rangle) = |b,x,f(b,x)\rangle$. Let $k \in \mathbb{N}_{\geq 1}$. Given $\boldsymbol{y} \in \mathbb{Z}_N^k$, define

$$\Lambda(\boldsymbol{y}) = \{\boldsymbol{x} \in \mathbb{Z}^k : \langle \boldsymbol{x}, \tilde{\boldsymbol{y}} \rangle \equiv 0 \pmod N\}, \tag{1}$$

where $\tilde{\boldsymbol{y}} \in \mathbb{Z}^k$ is any lift of $\boldsymbol{y}$ to $\mathbb{Z}^k$ and where $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^k x_i y_i$; we may sometimes abuse notation by writing $\boldsymbol{y}$ instead of $\tilde{\boldsymbol{y}}$ but this should cause no confusion. The set $\Lambda(\boldsymbol{y}) \subset \mathbb{Z}^k$ is an integral N-ary lattice of covolume $\det(\Lambda(\boldsymbol{y})) = N$ when N is prime, and of covolume $\det(\Lambda(\boldsymbol{y})) = N/d$, where $d = \gcd(y_1, \ldots, y_k, N)$ in general (this lattice is often denoted $\Lambda_N^\perp(\boldsymbol{y}^T)$ in the literature on lattices), and will play a central role in what follows. Given $\alpha \in \mathbb{Z}_N$, we also define

$$\Lambda_\alpha(\boldsymbol{y}) = \{\boldsymbol{x} \in \mathbb{Z}^k : \langle \boldsymbol{x}, \tilde{\boldsymbol{y}} \rangle \equiv \alpha \pmod N\}. \tag{2}$$

When $\alpha = 0$, $\Lambda_0(\boldsymbol{y}) = \Lambda(\boldsymbol{y})$, and in general, $\Lambda_\alpha(\boldsymbol{y}) = \boldsymbol{z}_\alpha + \Lambda(\boldsymbol{y})$ is a coset of $\Lambda(\boldsymbol{y})$, for some (hence any) solution $\boldsymbol{z}_\alpha \in \Lambda_\alpha(\boldsymbol{y})$ whenever $\Lambda_\alpha(\boldsymbol{y}) \neq \emptyset$.

For a positive integer M, we define $B_M = [-M, M]^k \cap \mathbb{Z}^k$, and we define

$$S_\alpha^{\boldsymbol{y}} = B_M \cap \Lambda_\alpha(\boldsymbol{y}). \tag{3}$$

That is, $S_\alpha^{\boldsymbol{y}}$ contains all solutions $\boldsymbol{i} \in B_M$ to the knapsack problem $\langle \boldsymbol{i}, \boldsymbol{y} \rangle = \alpha \pmod N$.

We define an operator O_C as follows:

$$O_C : |\boldsymbol{0}, \alpha, \boldsymbol{y}\rangle \mapsto |C_{\boldsymbol{y}}(\alpha), \alpha, \boldsymbol{y}\rangle\,, \tag{4}$$

where $C_{\boldsymbol{y}}(\alpha)$ is in $S_\alpha^{\boldsymbol{y}}$ with some probability which we shall quantify and analyze in detail in Sect. 3.4. We will build the operator O_C as a reversible circuit C that returns a single tuple $C_{\boldsymbol{y}}(\alpha)$ from α and $\boldsymbol{y}$.

We now present a subroutine (Alg. 1) to be used in our modified Childs–van Dam algorithm, followed by the algorithm itself (Alg. 2). Throughout the paper, the *quantum Fourier transform* (QFT) over $\mathbb{Z}_N$ is simply the discrete Fourier transform on $\mathbb{Z}_N$. We denote it by $\mathrm{QFT}_{\mathbb{Z}_N}$; it acts on a basis vector $|y\rangle$ as

$$\mathrm{QFT}_{\mathbb{Z}_N} |y\rangle = \frac{1}{\sqrt{N}} \sum_{z=0}^{N-1} \omega^{y\xi} |\xi\rangle\,, \quad \text{where } \omega = e^{2\pi\sqrt{-1}/N}. \tag{QFT}$$

Description of Alg. 1. In BuildRotatedIndex (Alg. 1), we build a group of registers with similar patterns to Kuperberg's algorithm. We start by building the uniform superposition of inputs

$$\frac{1}{\sqrt{N(2M+1)}} \sum_{i=-M}^{M} \sum_{x \in \mathbb{Z}_N} |i, x\rangle\,. \tag{A1.3}$$

Algorithm 1 BuildRotatedIndex

Input: N, M and *superposition* oracle access to f
Output: y, $(2M+1)^{-1/2}\sum_{i\in[-M,M]}\omega^{yiz}|i\rangle$

1: Start with three registers of the form $|0,0,0\rangle$ where the three registers will respectively contain elements in $\{-M,\ldots,M\}$, $\mathbb{Z}_N$ and S.
2: Apply the Quantum Fourier Transform (QFT) over $\mathbb{Z}_{2M+1}$ to the first register
3: Apply the QFT over $\mathbb{Z}_N$ to the second register $\rightsquigarrow$ (A1.3).
4: Apply the oracle O_f on the registers $\rightsquigarrow$ (A1.4).
5: Measure the last register and get $c = f(0, x_0)$ for an unknown x_0 $\rightsquigarrow$ (A1.5)
6: Apply the QFT over $\mathbb{Z}_N$ to the second register $\rightsquigarrow$ (A1.6).
7: Measure the last register and get y (uniformly) $\rightsquigarrow$ (A1.7).

Algorithm 2 Quantum algorithm

Input: N, M and *superposition* oracle access to f
Output: the hidden period z

1: Set $k = \lceil \log(N)/\log(2M+1)\rceil$.
2: Apply BuildRotatedIndex (Alg. 1) to k groups of registers $\rightsquigarrow$ (A2.2).
3: Compute $\alpha = \sum_{j=1}^{k} i_j y_j \bmod N$ in a new register $\rightsquigarrow$ (A2.3).
4: Apply O_C^{-1}, the reverse of the operator O_C to approximate (A2.4).
5: Apply $\mathrm{Id}\otimes \mathrm{QFT}_{\mathbb{Z}_N}^{-1}$ to approximate $|\mathbf{0}, z\rangle$ $\rightsquigarrow$ (A2.5).
6: Return the state (A2.5) and verify that it is $|\mathbf{0}, z\rangle$ by checking the equality $f(0,0) = f(1,z)$ and retry if it is not.

Observe that this can be done by applying the quantum Fourier transform (QFT) over $\mathbb{Z}_{2M+1}\times\mathbb{Z}_N$ to $|0,0\rangle$ (Steps 1 and 2 in Alg. 1), so (A1.3) $= (\mathrm{QFT}_{\mathbb{Z}_{2M+1}}\otimes\mathrm{QFT}_{\mathbb{Z}_N})|0,0\rangle$. This vector represents a uniform superposition of all basis states. We add a zero and then apply the function f (i.e., the operator O_f), and build the associated output

$$\frac{1}{\sqrt{N(2M+1)}}\sum_{i=-M}^{M}\sum_{x\in\mathbb{Z}_N}|i,x,f(i,x)\rangle\,. \tag{A1.4}$$

We then measure the output $c = f(0,x_0)$ and by the periodicity property of f, we get

$$\frac{1}{\sqrt{2M+1}}\sum_{i=-M}^{M}|i,x_0+iz\rangle\,. \tag{A1.5}$$

We apply the QFT to the second register to get

$$\frac{1}{\sqrt{N(2M+1)}}\sum_{i=-M}^{M}\sum_{x\in\mathbb{Z}_N}\omega^{(x_0+iz)x}|i,x\rangle\,. \tag{A1.6}$$

Since the outcome (say y) is uniformly distributed over $\mathbb{Z}_N$, when we measure it, we get

$$\frac{1}{\sqrt{2M+1}}\sum_{i=-M}^{M}\omega^{yiz}|i\rangle\,. \tag{A1.7}$$

Analysis of Alg. 2. After calling BuildRotatedIndex k times, the state is

$$\frac{1}{(2M+1)^{k/2}} \sum_{(i_1,\dots,i_k)\in B_M} \omega^{z\left(\sum_{j=1}^{k} i_j y_j\right)} |i_1,\dots,i_k\rangle\,. \tag{A2.2}$$

We compute $\alpha = \sum_{j=1}^{k} i_j y_j \bmod N$ in a new register. We put $\boldsymbol{i} = (i_1,\dots,i_k)$, and the global state becomes

$$\frac{1}{(2M+1)^{k/2}} \sum_{\boldsymbol{i}\in B_M} \omega^{z\alpha} |\boldsymbol{i},\alpha\rangle = \frac{1}{(2M+1)^{k/2}} \sum_{\alpha\in\mathbb{Z}_N} \omega^{z\alpha} \sum_{\boldsymbol{i}\in S_\alpha^{\boldsymbol{y}}} |\boldsymbol{i},\alpha\rangle\,. \tag{A2.3}$$

If we could erase $\boldsymbol{i}$ in the first k registers, i.e., $|\boldsymbol{i},\alpha\rangle \mapsto |\mathbf{0},\alpha\rangle$, we would get $(2M+1)^{-k/2}\sum_{\alpha\in\mathbb{Z}_N} |S_\alpha^{\boldsymbol{y}}|\cdot\omega^{z\alpha}|\mathbf{0},\alpha\rangle$. Heuristically, were we to approximate $|S_\alpha^{\boldsymbol{y}}|$ by its average $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|] \approx (2M+1)^k/N$ (see Sect. 3.3), we would get $\frac{1}{N}(2M+1)^{k/2}\sum_{\alpha\in\mathbb{Z}_N}\omega^{z\alpha}|\mathbf{0},\alpha\rangle$. So if we could erase the registers by renormalizing according to the partition of the box B_M by residue, i.e., $|\boldsymbol{i},\alpha\rangle \mapsto |S_\alpha^{\boldsymbol{y}}|^{-1/2}|\mathbf{0},\alpha\rangle$, then approximating $|S_\alpha^{\boldsymbol{y}}|^{1/2}$ by $(\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|])^{1/2}$ would yield

$$\frac{1}{N^{1/2}} \sum_{\alpha\in\mathbb{Z}_N} \omega^{z\alpha}|\mathbf{0},\alpha\rangle = \mathrm{QFT}_{\mathbb{Z}_N}(|z\rangle), \tag{A2.4}$$

and we could recover the hidden shift. Thus, the desired "uncompute" operation is $\sum_{\boldsymbol{i}\in S_\alpha^{\boldsymbol{y}}}|\boldsymbol{i},\alpha\rangle \mapsto |S_\alpha^{\boldsymbol{y}}|^{1/2}|\mathbf{0},\alpha\rangle$, which is the inverse of the operator $|\mathbf{0},\alpha\rangle \mapsto |S_\alpha^{\boldsymbol{y}}|^{-1/2}\sum_{\boldsymbol{i}\in S_\alpha^{\boldsymbol{y}}}|\boldsymbol{i},\alpha\rangle$. This quantum uniform sampling approach is adopted in the original quantum algorithm of [31]. However, due to unsolvable instances of the knapsack problem, this operator cannot be implemented in practice and hence has an intrinsic probability of success. Instead, we use in our algorithm the operator $O_C : |\mathbf{0},\alpha,\boldsymbol{y}\rangle \mapsto |C_{\boldsymbol{y}}(\alpha),\alpha,\boldsymbol{y}\rangle$ previously introduced in (4), and its inverse O_C^{-1}. The output is

$$|\psi_{\mathrm{out}}\rangle = (\mathrm{Id}\otimes\mathrm{QFT}_{\mathbb{Z}_N}^{-1})\circ O_C^{-1}\left(\frac{1}{(2M+1)^{k/2}}\sum_{\alpha\in\mathbb{Z}_N}\omega^{z\alpha}\sum_{\boldsymbol{i}\in S_\alpha^{\boldsymbol{y}}}|\boldsymbol{i},\alpha\rangle\right). \tag{A2.5}$$

3.2 A knapsack problem

To successfully implement our algorithm we need to describe how to implement the operator O_C. Clearly, this amounts to solving (for a superposition of α values) the knapsack problem given by $S_\alpha^{\boldsymbol{y}}$. Childs and van Dam use Lenstra's integer programming algorithm [57] to solve this problem in time $2^{O(k^3)}$. They also suggest to reduce this complexity to $2^{O(k\log k)}$ using Kannan's algorithm instead [46].

These asymptotic estimations for solving the integer programming problem at hand (implementing O_C) are useful for studying the asymptotic behavior of the Childs–van Dam algorithm, but they provide little insight on the actual cost of an attack against a concrete system such as CSI-SharK. For this reason we will reframe the knapsack problem as lattice problems in Sects. 5 and 6, and evaluate explicit costs for running the overall attack.

3.3 Analysis of the N-ary lattice

In this subsection, we analyze the knapsack problem from a theoretical standpoint via the lattice $\Lambda(\boldsymbol{y})$ and its cosets $\Lambda_\alpha(\boldsymbol{y})$ defined in Eqs. (1) and (2). This is necessary to understand our quantum algorithm and the various probabilistic assumptions.

If $\boldsymbol{y} \in \mathbb{Z}_N^k$ or $\boldsymbol{y} \in \mathbb{Z}^k$, we write $\gcd(\boldsymbol{y}, N) = \gcd(y_1, \ldots, y_k, N)$ – this should cause no confusion. Observe that $\Lambda_\alpha(\boldsymbol{y}) \neq \emptyset$ if $\gcd(\boldsymbol{y}, N) = 1$ which, for N prime, holds unless $\boldsymbol{y}$ is the zero vector mod N. More generally, if instead of $\Lambda_\alpha(\boldsymbol{y}) \subset \mathbb{Z}^k$, we consider its projection to $\mathbb{Z}_N^k$, then $|\Lambda_\alpha(\boldsymbol{y}) \pmod N| = \gcd(\boldsymbol{y}, N) \cdot N^{k-1}$ if $\gcd(\boldsymbol{y}, N) \mid \alpha$, and 0 otherwise. Thus, when N is prime, $\mathbb{Z}^k/\Lambda(\boldsymbol{y}) \cong \mathbb{Z}_N^k$, the cosets $\Lambda_\alpha(\boldsymbol{y})$ partition $\mathbb{Z}^k$ and each has the same cardinality when projected to $\mathbb{Z}_N^k$. For our algorithm, given M, we need to understand $|S_\alpha^{\boldsymbol{y}}| = |\Lambda_\alpha(\boldsymbol{y}) \cap B_M|$.

Distribution of residues. Recall that in Alg. 2, α is built from $\boldsymbol{y}$ and $\boldsymbol{i}$, and one of the main approximations that justify the design of our algorithm is $|S_\alpha^{\boldsymbol{y}}| \approx \mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|] \approx (2M+1)^k/N$. The following lemma justifies this: we show that the distribution of the residues α is indeed *almost uniformly random* over $\mathbb{Z}_N$ and compute $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|]$ for a uniform distribution. For the proof, see Sect. A.

Lemma 3.2. *Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_N^k$ uniformly random,*

$$\Pr_{\boldsymbol{x},\boldsymbol{y}}\big(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \equiv \alpha \pmod N\big) = \begin{cases} N^{-1} - N^{-(k+1)} & \text{if } \alpha \not\equiv 0 \pmod N, \\ N^{-1} + N^{-k} - N^{-(k+1)} & \text{if } \alpha \equiv 0 \pmod N. \end{cases}$$

Let $\alpha \in \mathbb{Z}_N$ be uniformly random. Then, $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|] = (2M+1)^k/N$.

Distribution of the number of solutions to the knapsack problem. We assume that $M < N$ and that N is prime. The following lemma summarizes the statistics of $|S_\alpha^{\boldsymbol{y}}|$ when $\boldsymbol{y}$ and α are both uniformly random.

In light of the following lemma, we set

$$k = \left\lceil \frac{\log(N)}{\log(2M+1)} \right\rceil.$$

Lemma 3.3. *If $\boldsymbol{y}$ and α are uniformly random over $\mathbb{Z}_N^k$, $\mathbb{Z}_N$ respectively, then*

$$\mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|] = \frac{(2M+1)^k}{N}, \quad \mathbf{Var}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|] = \left(1 - \frac{1}{N}\right) \mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|].$$

Further, if $1 - N^{-1} \leq \mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|]$, then $\Pr_{\alpha,\boldsymbol{y}}(|S_\alpha^{\boldsymbol{y}}| \geq 1) \geq 1/2$.

We also include the proof of this lemma in Sect. A.

3.4 Success probability

We now explain how to obtain a lower bound on the probability of success of our quantum algorithm and compare it to the original quantum algorithm of Childs and van Dam [31]. We also define precisely the success probability of the operator O_C defined in Eq. (4).

For a given $\boldsymbol{y} \in \mathbb{Z}_N^k$, we define the *probability of success of* O_C

$$P^C_{\text{succ}} = \mathbf{E}_{\boldsymbol{y}}[p_{\boldsymbol{y}}], \quad \text{where } p_{\boldsymbol{y}} = \Pr_{\alpha}\big(C_{\boldsymbol{y}}(\alpha) \in S^{\boldsymbol{y}}_{\alpha} \mid \boldsymbol{y}\big). \tag{5}$$

We also introduce the *probability of success on feasible instances* defined by

$$P^C_{\text{succ}} = \Pr_{\alpha,\boldsymbol{y}}(|S^{\boldsymbol{y}}_{\alpha}| \geq 1) \cdot P^C_{\text{succ,feas}}. \tag{6}$$

Here, $\boldsymbol{y}$ is uniformly random over $\mathbb{Z}_N^k$, as can be seen from our algorithm.

Proposition 3.4. *The probability of success of our quantum algorithm admits the lower bound:*

$$\Pr(\text{success}) \geq \frac{N}{(2M+1)^k} \cdot \left[\Pr_{\alpha,\boldsymbol{y}}(|S^{\boldsymbol{y}}_{\alpha}| \geq 1)\right]^2 \cdot (P^C_{\text{succ,feas}})^2.$$

In particular, if $1 - N^{-1} \leq \mathbf{E}_{\boldsymbol{y}}[|S^{\boldsymbol{y}}_{\alpha}|] \leq 1$, *then* $\Pr(\text{success}) \geq \frac{1}{4} \cdot \big(P^C_{\text{succ,feas}}\big)^2$.

The proof is given in Sect. B.

In the original quantum algorithm of [31], the operator O_C is replaced by quantum uniform sampling. In this case, the probability of success of their quantum algorithm admits the same lower bound without the factor $(P^C_{\text{succ,feas}})^2$. On the other hand, our algorithm is practical and concrete – unlike the uniform quantum sampling operator – and moreover allows to quantify success even with the existence of randomness in the solver used for the knapsack problem, leading to concrete complexity estimates.

3.5 Complexity analysis

The asymptotic complexity of Childs-van Dam algorithm is dominated by solving the knapsack problem. Using Kannan's algorithm this complexity is $2^{O(k \log k)}$, where $k \approx \log N / \log(2M+1)$, so the algorithm improves on Kuperberg's variants (running in time $2^{O(\sqrt{N})}$) when $M \in \Omega(2^{\sqrt{\log N}})$.

As we are interested in concrete complexity estimations for our variant of this algorithm, we analyze its three main subroutines : the quantum subroutine that performs the (isogeny) group action in superposition, Quantum Fourier Transforms, and solving the knapsack problem from Step 4.

Group action cost. Existing resource estimates for running a group action evaluation for CSIDH-512 parameters are listed in Tab. 2. Recall that multiple estimations exist, aiming to optimize either T-gate count or qubit count. We will keep this portion of the complexity analysis modular so that it is easy to modify according to which estimate is being considered, as well as any future (improved) estimates.

Quantum Fourier Transforms. In [3], Ahokas, Cleve, and Hales give an upper bound on the number of quantum gates necessary to compute a Quantum Fourier Transform modulo 2^n. They improved upon the previous upper bound of $O(n \log n)$, to $O(n(\log\log n)^2 \log\log\log n)$. This cost, however, is negligible in Childs–van Dam's algorithm, so we use the "naive" estimate of $n(n+1)/2$ gates.

Quantum complexity analysis of Childs–van Dam's algorithm. Recall from Alg. 2 that $k := \lceil \log(N)/\log(2M+1) \rceil$. The algorithm uses 3 QFTs in each call to BuildRotatedIndex (Alg. 1). This subroutine also includes one call to the group action oracle in Step 4. Step 3 of Alg. 2 consists of some computations which are negligible compared to the rest of the computations. Finally, Step 4 describes a knapsack problem and in Step 5 we have one final (inverse) QFT to do. Thus, at a high level, the complexity of the algorithm is

$$(3k+1) \text{ QFTs } + k \text{ group action evaluations } + \text{ knapsack problem.}$$

We recall that uncomputing the knapsack problem is applying the circuit that solves the knapsack problem and reversing it, thus the uncomputation has the same time and space complexity as the solving operation.

Note that k is relatively small in our attack (for CSI-SharK parameters, we may have $k = 20$). Thus, this algorithm differs from prior cryptanalysis in that the number of group action evaluations is small. For example, the algorithm from Peikert's quantum analysis of CSIDH [63, Fig. 2] for the same parameter sizes requires between 2^{14} to 2^{20} group action evaluations, depending on the length of the phase vector. As group action evaluation is quite expensive in practice, the reduced number of evaluations can significantly lower the overall cost.

In the following sections, we analyze the cost of solving the knapsack problem, before concluding the paper with concrete complexity estimates of our attack.

4 Solving the knapsack problem

Here we show how the knapsack problem in the Childs–van Dam algorithm can be formulated as an instance of a Closest Vector Problem (CVP) in $\Lambda(\boldsymbol{y})$. Our instance will use the ℓ^∞ norm instead of the usual ℓ^2 (Euclidean) norm. Further, we will show that we can assume an average-case hardness because of the ability to resample the lattice by rerunning Step 2 of Alg. 2. We describe how to actually solve the resulting CVP instance in the subsequent sections.

4.1 CVP formulation

Recall that to complete our attack we must find an element of $S_\alpha^{\boldsymbol{y}} = \Lambda_\alpha(\boldsymbol{y}) \cap B_M$. Recall also that $\{-M, \ldots, M\}$ is the set of available shifts we have access to, and k is the dimension we choose such that $|B_M| = (2M+1)^k \approx N$. Thus $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|] = \mathbf{E}_{\alpha,\boldsymbol{y}}[|S_\alpha^{\boldsymbol{y}}|] \approx 1$. As mentioned, $\boldsymbol{y}$ is fixed at the beginning of Step 2,

but can be re-randomized by rerunning BuildRotatedIndex (Alg. 1). Further, α takes all possible values in superposition.

In [31], the set $S_\alpha^{\boldsymbol{y}}$ is computed using a classical integer programming algorithm due to Lenstra [57]. The best complexity estimates are obtained when the interval is sufficiently large, while in cryptanalysis settings it will be fixed to a concrete size. Furthermore, the integer programming computation that was originally proposed is aimed at solving the worst case instance and does not offer concrete complexity estimates, making it difficult to measure the overall security of target systems. For these reasons we proceed with a reduction to an equivalent lattice problem.

We work with the N-ary lattice $\Lambda(\boldsymbol{y})$ defined in (1) and its cosets defined in (2). The equivalent lattice problem consists in finding vectors which are *close* to representatives of the coset $\Lambda_\alpha(\boldsymbol{y})$ (and thus, *short* when $\alpha = 0$) in the sense that the distance is less than M. As a shorthand, we shall use "CVP" and "SVP" to describe these problems, even though for our purposes it is not necessary that the vectors found be *the closest* or *the shortest*.

Let $\boldsymbol{z}_\alpha \in \Lambda_\alpha(\boldsymbol{y})$ be any solution; in order to find a "short" $\boldsymbol{x} \in \Lambda_\alpha(\boldsymbol{y}) \cap B_M$, we claim that it is enough to find a lattice vector in $\Lambda(\boldsymbol{y})$ which is "close" to $\boldsymbol{z}_\alpha$. Indeed, consider the shifted box $\boldsymbol{z}_\alpha + B_M$. If $\boldsymbol{v} \in (\boldsymbol{z}_\alpha + B_M) \cap \Lambda(\boldsymbol{y})$, that is, $\boldsymbol{v}$ is a lattice vector close to $\boldsymbol{z}_\alpha$ in the sense that $\|\boldsymbol{v} - \boldsymbol{z}_\alpha\|_\infty \leq M$, then we can write $\boldsymbol{v} = \boldsymbol{z}_\alpha + \epsilon$, where $\boldsymbol{v} \in \Lambda(\boldsymbol{y})$ and $\epsilon \in B_M$. But then, $(-\epsilon) = \boldsymbol{z}_\alpha - \boldsymbol{v} \in (\boldsymbol{z}_\alpha + \Lambda(\boldsymbol{y})) \cap B_M = \Lambda_\alpha(\boldsymbol{y}) \cap B_M$ as desired. This reduces the problem to finding elements of $(\boldsymbol{z}_\alpha + B_M) \cap \Lambda(\boldsymbol{y})$, as claimed.

4.2 A standard choice of basis of $\Lambda(\boldsymbol{y})$

We need a target vector $\boldsymbol{z}_\alpha$ and a matrix A whose columns form a basis of $\Lambda(\boldsymbol{y})$, i.e., $\Lambda(\boldsymbol{y}) = A\mathbb{Z}^k$. Assuming that $\boldsymbol{y} \neq \boldsymbol{0}$ in $\mathbb{Z}_N^k$, at least one coordinate is invertible mod N, say y_j, so without loss of generality, we may assume that $y_1 \neq 0$. Let $\alpha' = \alpha/y_1$ and let $\boldsymbol{y}' = \boldsymbol{y}/y_1 = (1, y_2', \ldots, y_k')$. Then, it is clear that $\Lambda_\alpha(\boldsymbol{y}) = \Lambda_{\alpha'}(\boldsymbol{y}')$, and in particular, $\Lambda(\boldsymbol{y}) = \Lambda(\boldsymbol{y}')$.[7] We may pick $\boldsymbol{z}_{\alpha'} = (\alpha', 0, \ldots, 0)$ and define

$$A = \begin{bmatrix} N & -y_2' & -y_3' & \cdots & -y_k' \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}. \tag{7}$$

It is easy to see that the columns of A form a basis for the lattice $\Lambda(\boldsymbol{y}')$.

In the rest of the paper, we will use α or α' interchangeably, but this should cause no confusion.

[7] Since $\boldsymbol{y} = \boldsymbol{0} \pmod N$ occurs only with probability $1/N$, the distributions of the lattices $\Lambda(\boldsymbol{y}')$ and $\Lambda(\boldsymbol{y})$ are almost the same, the former being conditioned on $\boldsymbol{y} \neq \boldsymbol{0} \pmod N$.

4.3 CVP specificities

The "CVP" instances we need to solve are in ℓ^∞ norm instead of Euclidean norm. Further, we only need one close vector within a box, as opposed to the closest vector.

The CVP algorithm must be run on a quantum superposition of α values. However, the lattice itself is independent of α and only depends on the elements $\{y_i\}$; one could therefore perform some (classical or quantum) preprocessing, typically to compute a reduced basis of $\Lambda(\boldsymbol{y})$, after the elements $\{y_i\}$ are obtained and before solving the CVP problems for a superposition of α values.

The elements $\{y_i\}$ are randomly chosen through measurement. One can tweak Chils-van Dam's algorithm and repeat this process until a good lattice, or a good enough precomputed basis for that lattice, is obtained. For our analysis this means that we assume both the lattice reduction and the CVP are average case instances instead of worst case instances. In fact, we can even assume to be in a somewhat "favorable case" instance, at the cost of repeating the process sufficiently many times. All this is carefully justified by a thorough discussion on random N-ary lattices of type $\Lambda(\boldsymbol{y})$. This will be beneficial, e.g. in the enumeration of Sect. 5, where the cost is highly dependent on the lattice basis.

4.4 Selecting CVP solvers

Our main task going forward will be to solve an average case variant of CVP in the ℓ^∞ norm. Current CVP algorithms broadly fall into three categories: enumeration [44, 46], sieving [1, 4], and Voronoi sets algorithms [39, 61].

Most of the literature describing algorithms for solving CVP problems targets the ℓ^2 norm, so a first idea may be to compute a ball of vectors that have small ℓ^2 norm, and check for suitably close vectors in infinity norm among them. However, a ball that is sufficiently large to contain all candidates for suitably close vectors for the infinity norm could also contain many more short vectors in the Euclidean norm, so the approach would be inefficient. For this reason we will be considering algorithms that directly use the ℓ^∞ norm.

Kannan [46] gives an enumeration algorithm for solving integer programming problems that splits the problem into several smaller ones. The time complexity of the algorithm is $2^{O(n \log n)}$. It is also deterministic and there is no space complexity. In Sect. 5 we describe a simple variant that is tailored to the setting of our problem. It requires little memory and is easy to implement, but the asymptotic time complexity will be worse than other approaches, meaning it may not scale so well to larger parameter sets.

Later on, in Sect. 6 we focus on a sieving approach. In [4], Ajtai, Kumar, and Sivakumar give a randomized algorithm based on sampling and sieving. The work of Aggarwal and Mukhopadhyay [1] follows the strategy from [4] at a high level, but focuses on the special case of using the ℓ^∞ norm. This approach offers a better time complexity than the enumeration approach, but will also require exponential size quantum memory.

We do not provide an approach using Voronoi sets, in part because the single exponential time complexity estimates computed in [61] are only for the Euclidean norm. In [16], the authors show that for non-Euclidean norms, it is unlikely to find algorithms of single exponential space and time that can solve CVP. They show that for some other norms the case is *similar* to the Euclidean one, and so may have single exponential algorithms. It is not clear how the ℓ^∞ norm plays into this, so instead we leave comparing such an approach (and any others) to future work.

In the following two sections, we assess the cost of the knapsack problem when implemented on a quantum computer with our specific choices of CVP solvers. It is of course entirely possible that better subroutines exist, hence our complexity estimates for our version of Childs–van Dam's algorithm should be considered as upper bounds.

5 Instantiating the attack with enumeration

As described in Sect. 4.1, the problem which remains to be solved to complete our attack using our quantum algorithm can be seen as a CVP problem in the ℓ^∞ norm. More precisely, given $\boldsymbol{y} \in \mathbb{Z}_N^k$, the elements of $S_\alpha^{\boldsymbol{y}} = \Lambda_\alpha(\boldsymbol{y}) \cap B_M$ correspond to lattice points in $\Lambda(\boldsymbol{y}) \cap (\boldsymbol{z}_\alpha + B_M)$ which are sufficiently close to the target vector $\boldsymbol{z}_\alpha$.

In this section, we describe a simple method which solves this CVP problem using enumeration. We start with an overview of the algorithm in Sect. 5.1 and describe how finding a better basis of the lattice allows us to lower the complexity of the enumeration. In Sect. 5.2, we provide experimental results on these improvements which are the basis of our security estimates later on. Finally, we prove rigorous upper bounds for the cost of this approach in the context of our attack with this quantum algorithm in Sect. 5.3.

5.1 Algorithm overview

Given a fixed lattice $\Lambda(\boldsymbol{y})$ with basis given by the matrix A and a target vector $\boldsymbol{z}_\alpha$ defined using α (see Sect. 4.1), we want to find a point in $\Lambda(\boldsymbol{y}) \cap (\boldsymbol{z}_\alpha + B_M)$. Recall from Sect. 3 that k is chosen so that for $\boldsymbol{y}$ and α uniformly random, there is on average only one such point[8].

Let $\boldsymbol{z}_\alpha$ be given as above, and let $\boldsymbol{x}_0 := \lfloor A^{-1}\boldsymbol{z}_\alpha \rceil$ be the approximate solution with its coordinates rounded to the nearest integer, i.e. $\|A^{-1}\boldsymbol{z}_\alpha - \boldsymbol{x}_0\|_\infty \leq \frac{1}{2}$. Assume now that $\boldsymbol{x}$ is a vector that yields a solution to our CVP problem at hand, i.e. we have $\|A\boldsymbol{x} - \boldsymbol{z}_\alpha\|_\infty \leq M$. While $\boldsymbol{x}_0$ will in general not yield a solution to the CVP problem itself, it gives an approximate solution, since

$$\|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \leq \|\boldsymbol{x} - A^{-1}\boldsymbol{z}_\alpha\|_\infty + \|A^{-1}\boldsymbol{z}_\alpha - \boldsymbol{x}_0\|_\infty$$

[8] In fact, it is not hard to show that if α is fixed and $\boldsymbol{y}$ is random, we also have $\mathbf{E}_{\boldsymbol{y}}[|S_\alpha^{\boldsymbol{y}}|] \approx 1$ for $\alpha \neq 0$ and $\mathbf{E}_{\boldsymbol{y}}[|S_\alpha^{\boldsymbol{y}}|] \approx 2$. The main difference occurs for $\alpha = 0$.

$$\leq \|A^{-1}\|_\infty \cdot \|A\boldsymbol{x} - \boldsymbol{z}_\alpha\|_\infty + \|A^{-1}\boldsymbol{z}_\alpha - \boldsymbol{x}_0\|_\infty \leq \|A^{-1}\|_\infty \cdot M + \frac{1}{2}.$$

This observation allows us to find a solution in the lattice $\Lambda(\boldsymbol{y})$ by enumerating through all $\boldsymbol{x}$ that lie at distance at most $\|A^{-1}\|_\infty \cdot M + \frac{1}{2}$ from $\boldsymbol{x}_0$. This gives at most $2\|A^{-1}\|_\infty \cdot M + 2$ options for each coordinate of $\boldsymbol{x}$ (from the positive and negative directions and 0), hence at most $(2\|A^{-1}\|_\infty \cdot M + 2)^k$ choices in total. For each candidate $\boldsymbol{x}$, we can verify whether it is a solution by checking whether $\|A\boldsymbol{x} - \boldsymbol{z}_\alpha\|_\infty$ is at most M. This check costs at most one matrix vector multiplication and one vector addition.

Algorithm 3 Algorithm using enumeration to solve CVP in ℓ^∞

Input: Basis matrix A of $\Lambda(\boldsymbol{y})$, target vector $\boldsymbol{z}_\alpha$.
Output: Lattice vectors with distance at most M from $\boldsymbol{z}_\alpha$ with respect to ℓ^∞ norm.

```
Compute x_0 := ⌊A^{-1} z_α⌉.
for each x with ‖x − x_0‖_∞ ≤ ‖A^{-1}‖_∞ · M + 1/2
    if ‖Ax − z_α‖_∞ ≤ M
        Return x.
    end if
end for
```

Lemma 5.1. *Alg. 3 finds a vector in $\Lambda(\boldsymbol{y}) \cap (\boldsymbol{z}_\alpha + B_M)$ after enumerating at most $(2\|A^{-1}\|_\infty \cdot M + 2)^k$ vectors.*

Amplitude amplification. We briefly discuss how to use Grover's search to accelerate the enumeration process of Alg. 3. Consider amplitude amplification introduced by Brassard, Hoyer, Mosca and Tapp [25], which extends Grover's algorithm [43] to any search space. Alg. 4 shows how Alg. 3 can be modified using this amplitude amplification. Note that Grover's search will require $\frac{\pi}{4}\sqrt{(2\|A^{-1}\|_\infty \cdot M + 2)^k / T}$ iterations, where $T \approx 1$ denotes the number of solutions.

Algorithm 4 Algorithm using enumeration to solve CVP in ℓ^∞

Input: Basis A of full-rank k-dimensional lattice, target vector $\boldsymbol{z}_\alpha$.
Output: Lattice vector with distance at most M from $\boldsymbol{z}_\alpha$ with respect to infinity norm.

```
Compute x_0 := ⌊A^{-1} z_α⌉.
Grover search on x with ‖x‖_∞ ≤ ‖A^{-1}‖_∞ · M + 1/2 with (π/4)(2‖A^{-1}‖_∞ · M + 2)^{k/2}
iterations using the following oracle:
    Check if ‖A(x + x_0) − z_α‖_∞ ≤ M
EndGrover
Return close vectors found.
```

The classical enumeration cost of $(2\|A^{-1}\|_\infty \cdot M + 2)^k$ vectors becomes $\frac{\pi}{4}(2\|A^{-1}\|_\infty \cdot M + 2)^{k/2}$ steps of amplification using [25, Thm. 4]. In each step

of the amplification, we have to generate the superposition of $\boldsymbol{x}$, which can be done by generating all of its k components individually using a QFT of size $(2\|A^{-1}\|_\infty \cdot M + 2)$ and checking if $\|A(\boldsymbol{x} + \boldsymbol{x}_0) - b\|_\infty \leq M$. The latter requires at most k^2 multiplications in $\mathbb{Z}_N$.

Minimizing $\|A^{-1}\|_\infty$. Given the dependence of the enumeration's complexity on $\|A^{-1}\|_\infty$, there are two easy ways of optimizing the approach:

1. Preprocess $\Lambda(\boldsymbol{y})$ to get a basis A which lowers $\|A^{-1}\|_\infty$.
2. Repeat Alg. 2 up to Step 3 until the measured lattice basis A is good, i.e. $\|A^{-1}\|_\infty$ is "sufficiently low".

Both steps together make our strategy. We repeat the initial quantum computation to get new random lattices, and we reduce their bases in order to minimize $\|A^{-1}\|_\infty$. If the resulting basis is sufficiently good, i.e. $\|A^{-1}\|_\infty$ of the resulting basis A is smaller than a desired threshold, we move forward and run the enumeration, otherwise we sample another lattice.

Note that each time we repeat the initial steps of the quantum computation, we obtain a new random lattice. Alg. 5 in summarizes the resulting strategy.

Algorithm 5 Using enumeration with preprocessing to find close vectors in Childs-van Dam algorithm.

Input: Class number N, M, superpos. oracle access to f with hidden period z and threshold t for $\|A^{-1}\|_\infty$
Output: The hidden period z

1: Let $c_A > t$.
2: **while** $c_A > t$
3: Run Alg. 2 up to Step 3 to get lattice $\Lambda(\boldsymbol{y})$ and target vector $\boldsymbol{z}_\alpha$ in superpos.
4: Preprocess basis A of $\Lambda(\boldsymbol{y})$ (either classically or with a quantum computer) to minimize $\|A^{-1}\|_\infty$, and compute resulting c_A.
5: **end while**
6: Use Grover's alg. for enumeration, Alg. 4, to get vectors in $\Lambda(\boldsymbol{y})$ at ℓ^∞ distance at most M from $\boldsymbol{z}_\alpha$.
7: Finish variant of Childs–van Dam as in Alg. 2.

Let $c_A := \|A^{-1}\|_\infty \cdot N^{1/k}$, where A corresponds to a basis of a k-dimensional lattice $\Lambda(\boldsymbol{y})$. For a basis with lengths close to the successive minima, we obtain the following theoretical upper bound for c_A.

Proposition 5.2. *Let* $0 < \gamma < 1$ *and* $\epsilon \geq 1/M$. *Define*

$$C(k) := \sqrt{\frac{6}{\pi \ln k}} + \frac{2\sqrt{3}}{\pi}.$$

Then,

$$1 \leq c_A \leq (2+\epsilon)\, C(k)\, \gamma^{-1} \cdot (k \ln k)^{1/2} \quad \textit{with probability } \geq 1 - \gamma^k,$$
$$1 \leq \mathbf{E}_{\boldsymbol{y}}[c_A] \leq \frac{k+1}{k} (2+\epsilon)\, C(k)\, \gamma^{-1} \cdot (k \ln k)^{1/2}.$$

We give some details for the proof of Prop. 5.2 in Sect. 5.3.

Using c_A, we can compute the overall complexity of the enumeration subroutine given by the following lemma.

Lemma 5.3. *Grover's search to find vectors in $\Lambda(\boldsymbol{y})$ at ℓ^∞ distance at most M from $\boldsymbol{z}_\alpha$ shown in Alg. 4 requires*

$$\log N \cdot \log\log N \cdot \log M \cdot k^2 \cdot \frac{\pi}{4}\left(2 \cdot c_A \cdot M \cdot N^{-1/k} + 2\right)^{k/2}$$

T-gates. The number of qubits necessary is $\log(M)nk^2 + kn$.

Proof (Lem. 5.3). We first compute the number of multiplications over $\mathbb{Z}_N$ used, and then show how to optimize the complexity by using additions instead.

By Lem. 5.1, there are at most $(2 \cdot c_A \cdot M \cdot N^{-1/k} + 2)^k$ vectors that need to be enumerated. Using amplitude amplification as described in Sect. 5.1, we can reduce this to $\frac{\pi}{4}(2c_A \cdot M \cdot N^{-1/k} + 2)^{k/2}$ steps of amplification. Each such step is dominated by checking if $\|A(\boldsymbol{x} + \boldsymbol{x}_0) - \boldsymbol{z}_\alpha\|_\infty \leq M$ which requires k^2 multiplications in $\mathbb{Z}_N$. The quantum complexity of the enumeration at hand thus becomes

$$k^2 \cdot \frac{\pi}{4}\left(2 \cdot c_A \cdot M \cdot N^{-1/k} + 2\right)^{k/2}$$

multiplications in $\mathbb{Z}_N$.

These multiplications in $\mathbb{Z}_N$, according to the estimate from Pavlidis and Gizopoulos [62, Tab. 4], would require a circuit using $2^{9.6}n^2 + 2^{9.3}n$ gates (for $n = \log N$) per multiplication. From Draper [37] the quantum cost of integer addition is around $n \log n$ gates. Thus we can try to save complexity by changing these multiplications to additions. Recall from Alg. 4, Step 3, that the computation being repeated is to check if $\|A(\boldsymbol{x}+\boldsymbol{x}_0)-b\|_\infty \leq M$. In each of the iterations, the variables $A, \boldsymbol{x}_0, \boldsymbol{z}_\alpha, M$ are all fixed so the values of $A\boldsymbol{x}_0 - \boldsymbol{z}_\alpha$ can be precomputed. Since $-M \leq x_j \leq M$, we can write $x_j = -M + \sum_{k=0}^{\lceil \log_2 M \rceil + 1} x_{jk} 2^k$ with $x_{jk} \in \{0, 1\}$. We then have $(A\boldsymbol{x})_i = -M \sum_j A_{ij} + \sum_{jk} A_{ij} x_{jk} 2^k$, where the values $-M \sum_j A_{ij}$ and $(A_{ij} 2^k \mod N)$ can be precomputed. This requires $k^2 \log M$ *additions*. In total, in each iteration this replaces k^2 multiplications with $k^2 \log M$ additions. This yields the claimed number of quantum T-gates.

Throughout this computation, the memory cost is dominated by storing a $k \times k$ matrix A^{-1} and a $k \times 1$ vector $\boldsymbol{z}_\alpha$. As each value in these structures requires at most n qubits to represent, this gives a total memory requirement of $(\log M)nk^2 + kn$ in the approach which trades multiplications for additions. □

To improve our cost estimates, we complement these theoretical bounds with experiments in the following subsection.

5.2 Experimental evaluation of c_A

The preceding subsections raised the question how to best preprocess a basis of a lattice to minimize c_A and how tight our theoretical upper bounds for c_A are for randomly generated lattices.

Note that after measuring the lattice basis, any preprocessing of the lattice can be done entirely using classical operations. Independent of whether this may or may not be implemented on a quantum computer in the end, we can implement and test this part experimentally on a classical computer to estimate both classical and quantum costs.

For our experiments, we implemented the following method to preprocess the lattice. We repeatedly chose a random $\boldsymbol{y}$ which gave rise to a random lattice $\Lambda(\boldsymbol{y})$. Then, we computed an approximation of a good basis by reducing the basis of the dual lattice using LLL in the ℓ^2 norm – partially motivated by the fast implementations of LLL available – and reducing the resulting lattice with respect to the infinity norm using an algorithm due to Lovasz and Scarf [58] before measuring the resulting c_A. Note that the final use of Lovasz–Scarf can only improve upon the result after the LLL reduction, and it is expected to give an improvement since in general a small linear combination of a good ℓ^2 basis can also reduce the ℓ^∞ norm.

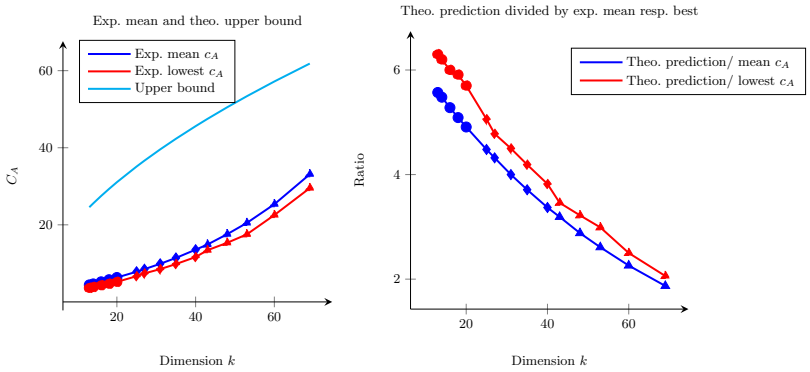


Fig. 2. Left: Mean (blue) and lowest (red) values of c_A observed experimentally for 10.000 lattices sampled randomly with parameter M with $\log_2(M) \in \{12, 14, 16, 18, 20\}$ and using parameter N of magnitude as in CSIDH-512 (circles), CSIDH-1024 (diamonds) and 100 lattices in the case of CSIDH-1792 (triangles). Theoretical upper bounds for the same parameters computed using Eq. (10) as cyan line. Right: Ratio of the theoretical prediction by the experimental mean (blue) and lowest (red) values observed for the same experiments.

We measured the values for different N of sizes approximately 2^{257}, 2^{512}, and 2^{892} roughly matching the one of CSIDH-512, CSIDH-1024 and CSIDH-1792, and taking $M \in \{2^{12}, 2^{14}, 2^{16}, 2^{18}, 2^{20}\}$ – specifying the dimension k. For the two smaller values of N and all values of M we repeated the experiment for 10.000 randomly chosen lattices and 100 lattices in the case of the largest N. The measurements are summarized in Fig. 2 and plotted against the proven upper bound of c_A. Further, we plot the ratio between the measurements and the

theoretical upper bound, showing that the theoretical bound becomes quickly tighter as the dimension grows. The values of c_A used to compute the concrete complexity estimates of our attack using enumeration are recorded in Tab. 3.[9]

	CSIDH-512, $N \approx 2^{257}$		CSIDH-1024, $N \approx 2^{512}$	
M	mean c_A	min c_A	mean c_A	min c_A
2^{20}	4.406	3.615	7.833	6.665
2^{16}	5.210	4.307	9.882	8.506
2^{12}	6.332	5.187	13.553	11.640

Table 3. Experimental results for c_A for different values of M over 10,000 randomly generated lattices in the case of both CSIDH-512 (as suggested in CSI-SharK) and CSIDH-1024 (approximating its class number by a 512-bit prime).

Note that an exhaustive search over (short) linear combinations of lattice vectors can further improve the basis to give a better c_A (at an exponential cost). However, we will ignore this potential improvement in our analysis.

5.3 Integer N-ary lattices of the form $\Lambda(\boldsymbol{y})$

We now explain the bounds from Prop. 5.2.

Let $L \subset \mathbb{R}^k$ be any lattice. We denote by L^* its dual lattice. For $1 \leq i \leq k$, we denote by $\lambda_i^{(p)}(L)$ the i-th successive minima of L in the ℓ^p-norm:

$$\lambda_i^{(p)}(L) = \min\{r > 0 \mid \dim \mathrm{Span}(L \cap B^{(p)}(r)) \geq i\},$$

where $B^{(p)}(r)$ denotes the ball centered at $\mathbf{0}$ with radius r in the ℓ^p norm.

The enumeration algorithm relies on having good control on $\|A^{-1}\|_\infty$ where A is a matrix whose columns form a basis of $\Lambda(\boldsymbol{y})$. We first observe that $\|A^{-1}\|_\infty = \max_{1\leq i\leq k} \|\mathrm{row}_i(A^{-1})\|_1$, and that the rows of A^{-1} form a basis of the dual lattice $\Lambda(\boldsymbol{y})^*$. Thus, we see that the theoretical best case scenario occurs when the ℓ^1 lengths of the basis vectors match the corresponding successive minima. This ideal basis matrix A^{-1} then satisfies $\|A^{-1}\|_\infty = \lambda_k^{(1)}(\Lambda(\boldsymbol{y})^*)$. Thus, we need to understand $\lambda_k^{(1)}(\Lambda(\boldsymbol{y})^*)$.

The results we need from the geometry of numbers are known as *transference theorems*, more precisely, for ℓ^1/ℓ^∞ duality.

Theorem 5.4. *Let $L \subset \mathbb{R}^k$ be a lattice. Then,*

$$\lambda_i^{(\infty)}(L) \cdot \lambda_{k-i+1}^{(1)}(L^*) \leq C(k) \cdot (k \ln k)^{1/2} \quad (1 \leq i \leq k), \tag{8}$$

where $C(k)$ is the same constant as in Prop. 5.2.

[9] The code to run the experiments, implemented in Magma [24], is provided in https://github.com/vgilchri/CvD-analysis.

Because we have an explicit constant and since Thm. 5.4 is a (minor) improvement of results of Banaszczyk [10]. Applying Thm. 5.4 with $i = 1$, we see that it remains to find a lower bound for $\lambda_1^{(\infty)}(L)$ when $L = \Lambda(\boldsymbol{y})$; this is provided by the next lemma (see Sect. A for the proof).

Lemma 5.5. *Deterministically, we have*

$$\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \leq N^{1/k}.$$

Assume that $\boldsymbol{y}$ is uniformly random. Fix $1/M \leq \epsilon \leq 1$. Then,

$$\mathbf{E}_{\boldsymbol{y}}\left[\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y}))\right] \geq \frac{k}{k+1} \cdot \frac{N^{1/k}}{2+\epsilon},$$
$$\Pr_{\boldsymbol{y}}\left(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \geq \frac{\gamma}{2+\epsilon} N^{1/k}\right) \geq 1-\gamma^k \quad (0 < \gamma < 1).$$

Therefore, with probability $\geq 1-\gamma^k$, there exists a basis matrix A of $\Lambda(\boldsymbol{y})$ such that

$$N^{-1/k} \leq \|A^{-1}\|_\infty \leq (2+\epsilon)C(k)\gamma^{-1} \cdot (k \ln k)^{1/2} \cdot N^{-1/k}. \tag{9}$$

Similarly, for the optimal basis matrix A of an average lattice, we have:

$$N^{-1/k} \leq \|A^{-1}\|_\infty \leq \frac{(2+\epsilon)(k+1)}{k} \cdot C(k) \cdot (k \ln k)^{1/2} \cdot N^{-1/k}. \tag{10}$$

Although presently, lattice reduction algorithms do not provide these bounds, they provide an approximation. If we assume that the rows of A^{-1} form a Korkin-Zolotarev basis for the dual lattice (with respect to the ℓ^1-norm), then by [58, Thm. 8], we have $2/(k+1) \leq \|A^{-1}\|_\infty / \lambda_k^{(1)}(\Lambda(\boldsymbol{y})^*) \leq (k+1)/2$. Thus, with probability $\geq 1-\gamma^k$, there is a basis matrix A such that

$$N^{-1/k} \leq \|A^{-1}\|_\infty \leq \frac{(2+\epsilon)C(k)}{\gamma} \cdot \frac{k+1}{2} (k \ln k)^{1/2} \cdot N^{-1/k}. \tag{11}$$

6 Instantiating the attack with sieving

In this section, we explore an alternative strategy to solve the CVP instance from Eq. (7) through sieving. First, we translate the problem into an SVP problem.

6.1 Reducing CVP to SVP

We reduce our CVP to an SVP instance in the ℓ^∞ norm. This is a variation of Kannan's embedding (see [46, p. 437]) that we briefly describe now.

Let $L \subset \mathbb{Z}^k$ be an integer lattice with basis matrix $A \in \mathrm{M}_k(\mathbb{Z})$. Let $\boldsymbol{t} \in \mathbb{Z}^n$, $\boldsymbol{t} \notin L$ – the target vector of the CVP problem – and let $\mu > 0$ be a parameter

to be specified shortly. Of course, our lattice is $L = \Lambda(\boldsymbol{y})$, A is the matrix from Eq. (7), and $\boldsymbol{t} = \boldsymbol{z}_{\alpha'}$. Define a new lattice $L_{\mu,\boldsymbol{t}} \subset \mathbb{Z}^{k+1}$; a basis matrix for $L_{\mu,\boldsymbol{t}}$ is

$$A_{\mu,\boldsymbol{t}} = \begin{bmatrix} A & -\boldsymbol{t} \\ 0 & \mu \end{bmatrix} \in \mathrm{M}_{k+1}(\mathbb{Z}).$$

We parametrize $\boldsymbol{v} \in L_{\mu,\boldsymbol{t}}$ as $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}, z) = A_{\mu,\boldsymbol{t}} \left[\begin{smallmatrix} \boldsymbol{x} \\ z \end{smallmatrix}\right] \in L_{\mu,\boldsymbol{t}}$.

Set $\mu = M$. On the one hand, if there exists $\xi = A\boldsymbol{x} \in L$ with $\|\xi - \boldsymbol{t}\|_\infty \leq M$, then there exists $\boldsymbol{v}(\boldsymbol{x}, \pm 1) \in L_{M,\boldsymbol{t}}$ with $\|\boldsymbol{v}(\boldsymbol{x}, \pm 1)\|_\infty \leq M$. On the other hand, if there exists $\boldsymbol{v}(\boldsymbol{x}, z) \in L_{M,\boldsymbol{t}}$ with $\|\boldsymbol{v}(\boldsymbol{x}, z)\|_\infty \leq M$, this implies that $z = 0$ or $z = \pm 1$. If we find a vector of the form $\boldsymbol{v}(\boldsymbol{x}, 0)$, we just discard it, and continue until we find a vector of the form $\boldsymbol{v}(\boldsymbol{x}, \pm 1)$, whose existence is guaranteed as long as $L \cap (\boldsymbol{t} + B_M) \neq \emptyset$.

We thus consider the augmented lattice $L_+ = L_{M,\boldsymbol{z}_{\alpha'}} \subset \mathbb{Z}^{k+1}$ with basis matrix

$$A_+ = A_{M,\boldsymbol{z}_{\alpha'}} = \begin{bmatrix} A & -\boldsymbol{z}_{\alpha'} \\ 0 & M \end{bmatrix}. \tag{12}$$

6.2 Sieving approach for SVP problems

Sieving algorithms proceed in two phases to solve the short vector problem:

1. *Sampling*, where s_0 elements of norm less than R_0 of the lattice are sampled.
2. *Sieving*, where given a list of length s_i of lattice elements of norm smaller than R_i, another list of elements of length s_{i+1} with norms smaller than R_{i+1} is produced. In practice, this is done by identifying pairs of "close" vectors and keeping their difference (a smaller vector of the lattice).

The sieving is repeated steps many times until we get a vector of the desired norm. We summarize this general approach in Alg. 6.

To solve the shortest vector problem in the ℓ^∞ norm, we will follow the work of Aggarwal and Mukhopadhyay [1,2]. In Sect. 6.3 we recall the relevant parts of their approach. We give complexity estimates for the different steps of sampling and sieving depending on the parameters the sieving is instantiated with, i.e. the lattice dimension $k+1$ and the parameters $\{(R_i, s_i)\}$ from Alg. 6. In Sect. 6.4, we will describe how these complexity estimates lead to an optimization problem. Finally, we approximate the solution to this optimization problem to obtain an upper bound on the cost of the resulting algorithm.

6.3 Sieving with Aggarwal-Mukhopadhyay's algorithm

Sampling. In [2], a randomized version of Babai's nearest plane algorithm is used to sample elements of the lattice. We are interested in the case where the lattice is of the form as in Eq. (12), i.e. the lattice is defined over $\mathbb{Z}$ but contains $(N\mathbb{Z})^k \times \{0\}$. Therefore, we can sample a random element of the lattice of ℓ^∞ norm less than $N/2$ with $k+1$ multiplications in $\mathbb{Z}_N$, as outlined in the following.

Algorithm 6 General approach of sieving algorithms

Input: A basis $A_{M,\boldsymbol{z}_{\alpha'}}$ of the lattice $L_{M,\boldsymbol{z}_{\alpha'}}$, suitable parameters $(R_0, s_0), ..., (R_{\mathsf{steps}}, s_{\mathsf{steps}})$, a sampling algorithm SAMPLE that given A_+ produces a random element of the lattice L of norm less than R_0 and a sieving algorithm SIEVE that given a list of cardinality s_i with lattice vectors of size R_i produces a list of cardinality s_{i+1} with lattice vectors of size R_{i+1}.

Output: An element of L of norm less than R_{steps}.

Sampling part
Initialize $S \leftarrow \emptyset$
for i from 1 to s_0
    $e_i \leftarrow \text{SAMPLE}(A_+, R_0)$, $S \leftarrow S \cup \{e_i\}$
end for
# Sieving part
for i from 1 to steps
    $S \leftarrow \text{SIEVE}(S, R_{i-1}, R_i)$
        ▷ SIEVE gets a list of size s_{i-1} and outputs a list of size s_i.
end for
Return a non-zero element of S.

Lemma 6.1. *For lattices of the same shape as L, Alg. 7 which chooses a "small" coefficient for the last row (less than $N/2M$) and random coefficients elsewhere using a representation of the output in $[-N/2, N/2]$ effectively samples a random element of the lattice of norm less than $N/2$.*

Algorithm 7 Sampling algorithm

Input: A matrix A_+ of the form given by Eq. (12) defining the lattice L_+.

Output: An element of L_+ of norm less than $N/2$.

Sample a random x_{k+1} in $[-N/2M, N/2M]$ and $x_1, ..., x_{k-1}$ in $[-N/2, N/2]$.
Compute $x_k = -\left\lfloor \frac{\alpha' x_{k+1} + \sum_{i=1}^{k-1} y'_{i+1} x_i}{N} \right\rceil$.
Return $[x_1, ..., x_{k+1}]A_+$.

Proof. The first choice gives that the last coefficient is a random element in $M\mathbb{Z} \cap [-N/2, N/2]$. (The matrix imposes that this coefficient is a multiple of M). The remaining choices and the reduction modulo N ensure the randomness of the other coefficients while not affecting the last one. □

Sieving. While [2] explores many refinements for the lattice sieving, these refinements come at a polynomial cost. While asymptotically the exponential cost is what matters most, these added costs are not negligible for the "small" parameters we are interested in. Hence, we only consider the simpler sieving as described in [2, Sect. 3].

Let $\lfloor \cdot \rceil$ denote rounding to the nearest integer. For an element in the lattice $\boldsymbol{x} = (x_1, \ldots, x_{k+1})$, we denote component-wise rounding by $\lfloor 2x/R_i \rceil := (\lfloor 2x_1/R_i \rceil, \ldots, \lfloor 2x_{k+1}/R_i \rceil)$. Let $\boldsymbol{x}$ and $\boldsymbol{x}'$ be two elements of the lattice such that $\lfloor 2\boldsymbol{x}/R_i \rceil = \lfloor 2\boldsymbol{x}'/R_i \rceil$. Then $\boldsymbol{x} - \boldsymbol{x}'$ must be an element of the lattice with

norm smaller than R_i. Therefore, sieving can be done by looking for collisions in the function $x \mapsto \lfloor 2\boldsymbol{x}/R_i \rceil$ (see also [2]).

To estimate the cost of finding such collisions we assume that elements of the lattice behave randomly (as in [2]) and we use the following lemma.

Heuristic 1 *At any stage of the sieving process, the vectors in $S \cap B_{R_i}$ are uniformly distributed in $B_{R_i} = \{\boldsymbol{x} \in \mathbb{Z}^{k+1} | \|\boldsymbol{x}\|_\infty \leq R_i\}$.*

Lemma 6.2. *For a random function $f : \{1, \ldots, n\} \to \{1, \ldots, m\}$, the number of collisions has expected value $\frac{n(n-1)}{2m}$ with variance $\frac{n(n-1)}{2m}\left(1 - \frac{1}{m}\right)$.*

The proof of Lem. 6.2 is included in Sect. D.

Assume that the map $\boldsymbol{x} \mapsto \lfloor 2\boldsymbol{x}/R_i \rceil$ behaves like a random function on elements of norm less than R_{i-1} in our $k+1$-dimensional lattice, i.e. it maps the elements uniformly randomly to the possible $(2R_{i-1}/R_i)^{k+1}$ buckets. Starting from a list of 2^n random elements of norms less than R_{i-1} and applying Lem. 6.2, we expect a list of at most $2^{2n-1}(R_i/(2R_{i-1}))^{k+1}$ elements of norm less than R_i (with standard deviation $2^{n-\frac{1}{2}}(R_i/(2R_{i-1}))^{(k+1)/2}$). Here, we omit the factor $(1 - \frac{1}{m})$ in the variance given by Lem. 6.2 as m is will be large. This formula provides constraints on the suitable values for (R_i, s_i).

Classical "many collisions finding" algorithm. As part of the sieving process, we need an algorithm to compute collisions. Note that there is not just one collision to find in this context, but *many* of them, and that this computation will be performed as part of Step 4 of Alg. 2, i.e. by a quantum computer on inputs in superposition. We use an algorithm from [22], which implements Alg. 9 given in Appendix D reversibly. Note that this algorithm requires a QRAM containing nearly all elements, but with stricter conditions on the data structure. The proof of the following Lem. 6.3 is given in Sect. D.

Lemma 6.3. *Let the notation be as in Alg. 6. Using classical many collisions finding, see Alg. 9, there exists a quantum procedure for sieving that takes a list of s_{i-1} elements of norms less than R_{i-1}, and outputs a list of s_i elements of norms less than R_i in time*

$$\frac{1}{\log(e)}(\log(s_i) + (k+1)\log(2R_{i-1}/R_i) + 1)\sqrt{2s_i}\left(\frac{2R_{i-1}}{R_i}\right)^{(k+1)/2},$$

whenever $s_i \leq \frac{1}{2}s_{i-1}^2(R_i/(2R_{i-1}))^{k+1}$.

Quantum collision finding. We sketch an alternative quantum approach to running the many collisions finding algorithm in Sect. D. The savings here are not significant for our scale so going forward we employ the classical collision finding technique instead.

6.4 Solving the optimization problem

To solve a concrete instance of the SVP using sieving as sketched in Alg. 6, Lem. 6.3 provides constraints and guidance to set the values for s_i and R_i in the sieving procedure.

To simplify notation, we define $n_i := \log(s_i)$ and $m_i := (k+1)\log(2R_{i-1}/R_i)$. Further, we recall that steps denotes the number of sieving steps, which is also a parameter to optimize. The constraints outlined in the previous subsections give rise to the following optimization problem.

Problem 6.4. Given variables $n_0, \ldots, n_{\mathsf{steps}}$ and $m_1, \ldots, m_{\mathsf{steps}}$ we have the following constraints:

$$\begin{cases} 2^{n_0} \leq \frac{N^{k+1}}{M^2}, \\ n_i \geq 1, \\ n_{i+1} \leq 2n_i - m_{i+1} - 1, \\ \sum_{i=1}^{\mathsf{steps}} (m_i - (k+1)) = (k+1)\log(N/M). \end{cases}$$

The complexity of the algorithm in T-gates is then given by the function

$$(k+1)C_N 2^{n_0} + \sum_{i=1}^{\mathsf{steps}} \frac{2(n_i + m_i + 1)}{\log(e)} 2^{(n_i+m_i+1)/2},$$

where C_N is the cost of a multiplication in $\mathbb{Z}_N$. Next, we minimize this cost.

Since N^k is significantly larger than M, the first condition is satisfied for every possible parameter set relevant to us. The other conditions are linear, so the solution set can be treated as a polyhedron. For simplicity, we replace $n_{i+1} \leq 2n_i - m_{i+1} - 1$ with $n_{i+1} = 2n_i - m_{i+1} - 1$. The cost function to be optimized, however, is not linear. In order to find sufficiently good solutions, we fix steps to an appropriate value and then treat this as a linear program. Note, the coefficient of n_0 is much larger in our applications than the other variables. Hence, we choose the objective function as $C_n \cdot n_0 + \sum_{i=1}^{n} n_i$ (note that the n_i already define the m_i). This alone would yield bad results as optimal solutions are not balanced and slightly large n_i blow up the complexity. By imposing an upper bound on all n_i depending on the dimension k, we can prevent this. Once a solution is computed, we plug it into the function $(k+1)C_N 2^{n_0} + \sum_{i=1}^{\mathsf{steps}} \frac{2(n_i+m_i+1)}{\log(e)} 2^{(n_i+m_i+1)/2}$ to compute the complexity of sieving.

Experimentally we observe that there appears to be an optimal choice for the upper bound on the n_i and the value steps. It makes sense to take steps small, albeit large enough that a solution exists.

We chose $\mathsf{steps} = 600$ and the upper bound on the n_i to be $k+3$; we do not claim that this is an optimal choice. The following table lists the number of operations necessary for various dimensions k after computing solutions to the optimization problem[10].

[10] The optimization was run with the Magma computer algebra system [24] and the code is provided in https://github.com/vgilchri/CvD-analysis. Linear pro-

k	13	16	20	24	29	31	40
complexity	$2^{33.0}$	$2^{35.0}$	$2^{38.9}$	$2^{43.2}$	$2^{48.4}$	$2^{50.5}$	$2^{59.8}$

Table 4. We list the resulting complexity in number of T-gates for running the sieving approach to solving the knapsack problem for different values of k.

Using these experimental values, we can estimate the amount of resources necessary to run the entire sieving procedure.

Lemma 6.5. *Let L_{sieving} be as in Lem. 6.3. Then Alg. 6 can be run using approximately $6.45 \cdot L_{\mathsf{sieving}}$ T-gates and L_{sieving} many ancilla qubits.*

Proof. The time complexity for running Alg. 6, L_{sieving}, was computed in Lem. 6.3. Note, experimental values for L_{sieving} are computed in Tab. 4. The claimed T-gate complexity follows, except for the factor of 6.45.

In terms of memory requirements, using a merge sort algorithm during the sieving procedure would mean that we would need to store the entire lists in quantum accessible quantum memory (QRAQM). This would give a QRAQM cost of size L_{sieving}. However, these costs can be circumvented. Indeed, the sieving procedure only uses QRAQM for two purposes : sorting lists of elements and extracting the list of collisions from the sorted lists. The sorting could be changed to a "comb" sort (Alg. 8) from Dobosiewicz [35]. Lacey and Box conjectured that the shrink factor of 1.3 made sure that the returned list was sorted [55].

Algorithm 8 Comb sort

Input: A list L of n (comparable) elements and a shrink factor shr (usually 1.3)
Output: The list L sorted.

$gap = \lfloor \frac{n}{shr} \rfloor$
while $gap \geq 1$
  for i from 1 to $n - gap$
    if $L(i) > L(i + gap)$
      $L(i), L(i + gap) := L(i + gap), L(i)$
    end if
  end for
end while
Return L.

While this sorting algorithm is $\log(2)/\log(1.3) \simeq 2.64$ times slower than merge sort and relies on a heuristic, the indices of the compared elements do not depend on the value of the compared elements. It means that it can be fully implemented as a quantum circuit without any QRAQM or QRACM gates.

gramming in Magma is implemented using the lp solve library written by Michel Berkelaar, the library source may be found at `ftp://ftp.ics.ele.tue.nl/pub/lp_solve/`.

The extraction can be handled with the resolved sorting : build the list of differences of elements that are close in the sorted list (we expect collisions to involve at most $\log(n)/\log(\log(n))$ elements), sort that new list regarding the norm (we only need the first $\log_{1.3}(1.3\log(n)/0.3)$ rounds of the comb sort as we only need to approximately separate the small elements from the rest), and the small values are kept (the number of elements to keep is predetermined by the parameters of Alg. 6).

This gives a total overhead factor of $\log(2e)/\log(1.3) \simeq 6.45 \approx 2^{2.7}$ (as the extraction part used to be of negligible cost). The ancilla qubit count is now dominated by the extraction part which requires L_{sieving} qubits. □

7 Quantum security of the Vectorization Problem with Shifted Inputs

We now apply our analysis to assess the security of the Vectorization Problem with Shifted Inputs, in particular for CSIDH parameters as suggested in the CSI-SharK and BCP protocols. We begin by computing the T-gate complexity and memory requirements of our quantum algorithm, detailed in Alg. 2.

Theorem 7.1. *Fix N (the size of the class group), M (the number of consecutive samples provided), $n := \log N$, and $k := n/\log(2M+1)$.*

Suppose that one group action evaluation requires G_T T-gates. Let Δ denote the number of times that the lattice $\Lambda(\boldsymbol{y})$ is resampled. Let L_T denote the cost of solving the CVP instance formulated in Sect. 4. The number of T-gates necessary to run Alg. 2 is

$$(3\Delta k+1)\frac{n(n+1)}{2} + \Delta G_T k + L_{\mathsf{T}}.$$

Suppose one group action evaluation requires G_q ancilla qubits, and that the CVP solver requires L_q ancilla qubits. Then Alg. 2 requires $\max\{G_q, L_q\}$ ancilla qubits to run.

Proof. Recall from Sect. 3.5 that the T-gate complexity of Childs–van Dam's algorithm is

$$(3k+1)\text{ QFTs } + k\cdot G_T + \text{ knapsack problem.}$$

Further, recall that the cost of a QFT is estimated to be around $n(n+1)/2$. However, each time the basis which fixes the CVP instance is resampled, it requires rerunning Step 2 of Alg. 2. This requires an additional $3k$ QFTs and an additional k calls to the group action oracle. Putting this together, the final T-gate count is $(3\Delta k+1)\frac{n(n+1)}{2} + \Delta G_T k + L_{\mathsf{T}}$.

The two subroutines requiring a significant number of ancilla qubits are the group action evaluation oracle and the CVP solver. These two subroutines are not run in parallel, thus we may take the maximum between these resources as the required number of ancilla qubits for the overall algorithm. □

Recall that the values of L_T and L_q when using enumeration from Sect. 5 were computed in Lem. 5.3. The exact cost formula depends on the constant c_A. Some example values of c_A are listed in Tab. 3. Since the experiments were taken over 10.000 lattices, we can either select the mean c_A value and set $\Delta = 1$, or we can use the min c_A value and set $\Delta = 2^{12}$. Using the minimum of the value computed either with the mean c_A and $\Delta = 1$ or the minimum c_A and a respectively larger Δ, we obtain the values listed in Tab. 5 .

The formulae for L_T and L_q when using the sieving algorithm from Sect. 6 were given in Lem. 6.5. This approach did not involve resampling the lattice, in which case $\Delta = 1$.

group action cost estimate		Peikert Alg. [63, Sect. 4.1]		M	k	Alg. 2 + sieving		Alg. 2 + enum.	
T-gates	ancillas	T-gates	QRACM			T-gates	ancillas	T-gates	ancillas
(from [13])				2^{8}	29	$\mathbf{2^{51.7}}$	$2^{48.4}$	$2^{72.9}$	$2^{17.7}$
$2^{43.8}$	2^{40}	$2^{59.8}$ to $2^{62.8}$	2^{40} to 2^{32}	2^{12}	20	$\mathbf{2^{50.8}}$	2^{40}	$\mathbf{2^{54.9}}$	$2^{16.7}$
				2^{16}	16	$\mathbf{2^{50.5}}$	2^{40}	$\mathbf{2^{51.7}}$	$2^{16.1}$
				2^{20}	13	$\mathbf{2^{50.2}}$	2^{40}	$\mathbf{2^{47.5}}$	$2^{15.5}$
(from [23])				2^{8}	29	$\mathbf{2^{60}}$	$2^{48.4}$	$2^{72.9}$	$2^{17.7}$
$2^{52.4}$	$2^{15.3}$	$2^{68.4}$ to $2^{71.4}$	2^{40} to 2^{32}	2^{12}	20	$\mathbf{2^{59.4}}$	$2^{38.9}$	$\mathbf{2^{56.7}}$	$2^{16.7}$
				2^{16}	16	$\mathbf{2^{59.1}}$	2^{35}	$\mathbf{2^{56.4}}$	$2^{16.1}$
				2^{20}	13	$\mathbf{2^{58.8}}$	$2^{33.0}$	$\mathbf{2^{56.1}}$	$2^{15.5}$

Table 5. Using CSIDH-512 parameter sets, we list complexities in number of T-gates and memory required of the Childs–van Dam algorithm when given access to shifts of the form $g^{c \cdot z}$ for $c \in [-M, M]$. We compare against Peikert's algorithm, based on Kuperberg's collimation sieve, whose (oracle calls, QRACM) usage ranges from $(2^{16}, 2^{40})$ to $(2^{19}, 2^{32})$. Improved T-gate counts are bolded.

Concrete security estimates. In Tab. 5 , we list the concrete total quantum security complexities of our modified Childs–van Dam algorithm depending on the choice of CVP solver, and on the choice of cost model for the oracle query according to the resource estimates from Thm. 7.1. Recall that the two group action cost estimates used are from [13] and [23] and are listed in Tab. 2. Note that $M = 2^{12}$ was suggested as a valid parameter in CSI-SharK, and $M = 2^{12}, 2^{15}, 2^{18}$ were suggested in BCP [8]. This means that we improve upon the state-of-the-art complexity estimate for both CSI-SharK and BCP protocols. Overall, we were able to decrease the number of oracle calls required (compared to running Kuperberg on a single shift) to solve the Vectorization Problem with Shifted Inputs and avoid the use of QRAM.

We compare our algorithm against the attack from Peikert [63, Sect. 4.1].

Concrete comparison between Childs–van Dam and Kuperberg. As the parameters k and M in our quantum algorithm are related by $k := \log N/(\log(2M+1))$, larger M lead to a smaller k, i.e. a smaller-dimensional lattice with an easier CVP instance.

When very few shifts are available, Peikert's version of Kuperberg still performs better than our version of the Childs–van Dam algorithm, due to some overheads in the latter. As the number of available shifts increases, our modified Childs–van Dam becomes the most efficient algorithm. Judging by the results displayed in Tab. 5 the cross-over point for the sieving approach is below 2^8. When $M \geq 2^{12}$, the enumeration approach can work with as little as $2^{16.7}$ ancilla qubits and still outperform the state-of-the-art for T-gates. The sieving approach consistently outperforms the enumeration approach in terms of T-gates. On the other hand, the enumeration approach can be performed with low memory complexity so depending on the quantum cost metric adopted it may be considered more practical. In both cases, group action evaluation queries eventually become the dominating costs. At that point, increasing M further only marginally decreases the T-gate complexity.

We outline how the T-gate requirements for the algorithms scale to the CSIDH-1024 parameter set in Sect. E.

8 Conclusion and Perspectives

In this work, we revisited the concrete quantum security of the Vectorization Problem with Shifted Inputs, a variant of the vectorization problem used in isogeny-based cryptography. The best quantum cryptanalysis attacks on the Vectorization Problem with Shifted Inputs consisted in attacking the Vectorization Problem itself, and parameter selection in [7, 8] suggests that the two problems are considered equivalently hard to solve in practice.

In contrast, our work shows that the two problems are in fact not equivalent with respect to quantum computers. By specifying and analyzing a modified version of a quantum algorithm of Childs and van Dam, we leverage the additional information provided in the Vectorization Problem with Shifted Inputs, and we obtain new quantum attacks with lower T-gate complexities. A core subtask in our quantum algorithm (as in Childs and van Dam's original algorithm) consists of solving a knapsack problem. We considered two approaches based on enumeration and sieving for this. Our analysis suggests that the sieving approach has better T-gate complexity for relevant parameters, while the enumeration stands out for its low memory requirements. For several parameters we improve upon previous work with respect to both cost metrics simultaneously. While this was the first direct application of the (modified version of the) Childs–van Dam algorithm in a cryptographic setting, it may be applicable elsewhere. We give a comprehensive list of possible directions for future work.

Potential improvements In this work we did not consider using Voronoi sets to solve the instance of CVP due to the high memory requirements, however, this

approach may offer benefits of its own. When working with ℓ^p norms for $p \neq 2$, the Voronoi set can be superexponential [16], but it is not clear if this is the case in the instance of CSI-SharK. A second area that may benefit from independent study is the parameter selection for the sieving approach in Sect. 6.4. Here, the parameters $\{n_i, m_i\}$ had to be selected in such a way that optimized the final complexity of the algorithm. More sophisticated optimization algorithms could lead to improved results.

Countermeasures A potential countermeasure for CSI-SharK may be to choose the shifts such that they are not consecutive integers. Doing so may not be trivial, however, as variants of this attack would still apply in some instances. Using a punctured list of consecutive integers (i.e. simply skipping a few values) does not completely avoid the attack, but only decreases the probability of success depending on how many holes there are. If the set of integers is uniformly spaced, for example, they are of the form $\{2, 4, 6, \ldots\}$, then we can replace $f(b, x)$ from Sect. 3 by $g(b, x) = f(2b, x)$ and the period becomes $(1, 2z)$ instead of $(1, z)$. Note that increasing the size of the integers appearing in the (super)exceptional sets would in turn allow to decrease k. Otherwise, the rest of the attack follows just the same.

Related protocols In this work, we focused on the Vectorization Problem with Shifted Inputs, a variant of vectorization problem used in the CSI-SharK and BCP protocols. Many other variants have appeared in the literature, some of them partly similar to the same problem. We list some of these problems below, and we encourage the community to extend our results and study their exact quantum security.

CSI-Otter. In [47], the authors give a (partially) blind signature scheme from isogenies called CSI-Otter. Though the soundness of the scheme was recently attacked in a special case [48], the scheme in practice is unaffected. In particular, its underlying hard problem, ζ_d-rGAIP [47, Definition 2.9], remains unaffected. We state it here.

Problem 8.1 (ζ_d-Ring Group Action Inverse Problem). Let $E \in \mathcal{E}\ell\ell_p(\mathcal{O})$. Given

$$E \cup \{[\mathfrak{g}^{\zeta_d^j \cdot z}]E\}_{j \in [d]},$$

where $z \in \mathbb{Z}_N$ is secret and $d \mid \lambda(N)$ (here λ is the Carmichael function), compute z.

This problem is very similar to that of Problem 2.2 except that the shifts are not uniformly spaced. Can adjustments be made to Alg. 2 to accommodate this difference?

k-power DDHA. Another hard problem to consider is the *k-power Decisional Diffie-Hellman Group Action Problem* which was first used in an isogeny-based threshold signature by De Feo and Meyer [41, Problem 1] but was later generalized to any group action in [38, Definition 8]. This problem (when $k = 2$) was also used in a quantum money construction from Zhandry [66], who suggests isogenies may be a good candidate for instantiation. We state the problem in terms of the isogeny group action.

Problem 8.2 (k-power Decisional Diffie-Hellman Group Action Problem). Let $E \in \mathcal{E}\ell\ell_p(\mathcal{O})$, $1 < k < N$ an integer, and $\mathfrak{g}^z \in \mathrm{cl}(\mathcal{O})$. Given $(k, E, [\mathfrak{g}^z]E, F)$, where $F \in \mathcal{E}\ell\ell_p(\mathcal{O})$, determine whether $F = [\mathfrak{g}^{k\cdot z}]E$ or whether it was sampled uniformly randomly from $\mathcal{E}\ell\ell_p(\mathcal{O})$.

At first sight, when F is not random, this problem could be viewed as a hidden shift problem where we have access to two shifts, namely $(E, [\mathfrak{g}^z]E, [\mathfrak{g}^{k\cdot z}]E)$. Thus we could attempt to run the algorithm using $(E, [\mathfrak{g}^z]E, F)$. If we obtain a viable candidate for z, then F was in fact of the form $F = [\mathfrak{g}^{k\cdot z}]E$, otherwise it was random. The main issues here are that due to the very small number of shifts, and the possibility of k being very large, the probability of the algorithm returning a correct answer when F is not random will not be very high. Nonetheless, this approach may shed some light on the quantum security of [41], either affirming its claims or improving the state-of-the-art.

An OPRF from CSIDH. In [34], Delpech de Saint Guilhem and Pedersen give an OPRF from CSIDH. In their proof of one-more unpredictability, the adversary has access to an OPRF oracle that on input $m \in \mathbb{Z}_N$ outputs $[\mathfrak{g}^{f(m)}]E$, where f is a secret polynomial of the form $f(x) = a(x + b)^3 + c$. The adversary in this context can make polynomially many queries to the oracle, however, due to the fact that the polynomial is kept secret, it is not possible to know what the shifts are.

Other group action based cryptography. None of our work relies on the use of isogenies, and so applying this attack to other abelian group actions remains an interesting question.

References

1. Divesh Aggarwal and Priyanka Mukhopadhyay. Improved algorithms for the shortest vector problem and the closest vector problem in the infinity norm. In Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao, editors, *29th International Symposium on Algorithms and Computation, ISAAC 2018, December 16-19, 2018, Jiaoxi, Yilan, Taiwan*, volume 123 of *LIPIcs*, pages 35:1–35:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
2. Divesh Aggarwal and Priyanka Mukhopadhyay. Improved algorithms for the shortest vector problem and the closest vector problem in the infinity norm. *arXiv preprint arXiv:1801.02358*, 2018.
3. Graeme Ahokas, Richard Cleve, and Lisa Hales. The complexity of Quantum Fourier Transforms and integer multiplication. ERATO Conference on Quantum Information Science, 2003.
4. Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 601–610. ACM, 2001.
5. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 411–439. Springer, 2020.
6. Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.
7. Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. CSI-SharK: CSI-FiSh with Sharing-friendly Keys. In *Information Security and Privacy - 28th Australasian Conference, ACISP 2023, Brisbane, QLD, Australia, July 5-7, 2023, Proceedings*, volume 13915 of *Lecture Notes in Computer Science*, pages 471–502. Springer, 2023.
8. Karim Baghery, Daniele Cozzo, and Robi Pedersen. An isogeny-based ID protocol using structured public keys. In Maura B. Paterson, editor, *Cryptography and Coding - 18th IMA International Conference, IMACC 2021, Virtual Event, December 14-15, 2021, Proceedings*, volume 13129 of *Lecture Notes in Computer Science*, pages 179–197. Springer, 2021.
9. W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Math. Ann.*, 296(4):625–635, 1993.
10. W. Banaszczyk. Inequalities for convex bodies and polar reciprocal lattices in $\mathbf{R}^n$. *Discrete Comput. Geom.*, 13(2):217–231, 1995.
11. Naomi Benger, Dario Catalano, Manuel Charlemagne, David Conti, Biljana Cubaleska, Hernando Fernando, Dario Fiore, Steven Galbraith, David Galindo, Jens Hermans, Vincenzo Iovino, Tibor Jager, Markulf Kohlweiss, Benoit Libert, Richard Lindner, Hans Loehr, Danny Lynch, Richard Moloney, Khaled Ouafi, Benny Pinkas, Frantisek Polach, Mario Di Raimondo, Markus Ruckert, Michael Schneider, Vijay Singh, Nigel Smart, Martijn Stam, Fre Vercauteren, Jorge Villar Santos, and Steve Williams. Main computational assumptions in cryptography. *European Network of Excellence in Cryptology II*, 2010.
12. Charles H Bennett. Logical reversibility of computation. *IBM journal of Research and Development*, 17:525–532, 1973.

13. Daniel J. Bernstein, Tanja Lange, Chloe Martindale, and Lorenz Panny. Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 409–441. Springer, 2019.
14. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. *Advances in Cryptology – ASIACRYPT 2019*, pages 227–247, 2019.
15. Jean-François Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In Willi Meier and Debdeep Mukhopadhyay, editors, *Progress in Cryptology – INDOCRYPT 2014*, pages 428–442, Cham, 2014. Springer International Publishing.
16. Johannes Blömer and Kathlén Kohn. Voronoi cells of lattices with respect to arbitrary norms. *SIAM J. Appl. Algebra Geom.*, 2(2):314–338, 2018.
17. Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.
18. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.
19. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.
20. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.
21. Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.
22. Xavier Bonnetain, André Chailloux, André Schrottenloher, and Yixin Shen. Finding many collisions via reusable quantum walks: Application to lattice sieving. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 221–251. Springer, 2023.
23. Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceed-*

ings, Part II, volume 12106 of *Lecture Notes in Computer Science*, pages 493–522. Springer, 2020.
24. Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system I: The user language. *Journal of Symbolic Computation*, 24(3-4):235–265, 1997.
25. Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.
26. Andries E. Brouwer and Willem H. Haemers. *Spectra of graphs*. Universitext. Springer, New York, 2012.
27. J. W. S. Cassels. *An introduction to the geometry of numbers*. Die Grundlehren der mathematischen Wissenschaften, Band 99. Springer-Verlag, Berlin-New York, 1971. Second printing, corrected.
28. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427. Springer, 2018.
29. Jung Hee Cheon. Discrete logarithm problems with auxiliary inputs. *J. Cryptol.*, 23(3):457–476, 2010.
30. Andrew M. Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *J. Mathematical Crytology*, 8(1):1–29, 2014.
31. Andrew M Childs and Wim Van Dam. Quantum algorithm for a generalized hidden shift problem. *arXiv preprint quant-ph/0507190*, 2005.
32. David A. Cox. *Class Field Theory*, chapter 2, pages 87–179. John Wiley & Sons, Ltd, 2013.
33. Pierrick Dartois, Jonathan Komada Eriksen, Tako Boris Fouotsa, Arthur Herlédan Le Merdy, Riccardo Invernizzi, Damien Robert, Ryan Rueger, Frederik Vercauteren, and Benjamin Wesolowski. PEGASIS: Practical effective class group action using 4-dimensional isogenies. Cryptology ePrint Archive, Paper 2025/401, 2025.
34. Cyprien Delpech de Saint Guilhem and Robi Pedersen. New proof systems and an OPRF from CSIDH. In Qiang Tang and Vanessa Teague, editors, *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part III*, volume 14603 of *Lecture Notes in Computer Science*, pages 217–251. Springer, 2024.
35. Wlodzimierz Dobosiewicz. An efficient variation of bubble sort. 1980.
36. Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 2005.
37. Thomas G. Draper. Addition on a quantum computer, 2000.
38. Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Generic models for group actions. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I*. Springer, 2023.

39. Friedrich Eisenbrand, Nicolai Hähnle, and Martin Niemeier. Covering cubes and the closest vector problem. In Ferran Hurtado and Marc J. van Kreveld, editors, *Proceedings of the 27th ACM Symposium on Computational Geometry, Paris, France, June 13-15, 2011*, pages 417–423. ACM, 2011.
40. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: scaling the csi-fish. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 345–375. Springer, 2023.
41. Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II*, volume 12111 of *Lecture Notes in Computer Science*, pages 187–212. Springer, 2020.
42. Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 2007–2023. ACM, 2020.
43. Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.
44. Guillaume Hanrot and Damien Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007*, pages 170–186, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
45. Samuel Jaques and Arthur G. Rattew. Qram: A survey and critique, 2023.
46. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987.
47. Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. CSI -otter: Isogeny-based (partially) blind signatures from the class group action with a twist. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 729–761. Springer, 2023.
48. Shuichi Katsumata, Yi-Fu Lai, and Michael Reichle. Breaking parallel ROS: implication for isogeny and lattice-based blind signatures. In Qiang Tang and Vanessa Teague, editors, *Public-Key Cryptography - PKC 2024 - 27th IACR International Conference on Practice and Theory of Public-Key Cryptography, Sydney, NSW, Australia, April 15-17, 2024, Proceedings, Part I*, volume 14601 of *Lecture Notes in Computer Science*, pages 319–351. Springer, 2024.
49. Taechan Kim. Security analysis of group action inverse problem with auxiliary inputs with application to CSIDH parameters. In Jae Hong Seo, editor, *Information Security and Cryptology - ICISC 2019 - 22nd International Conference, Seoul, South Korea, December 4-6, 2019, Revised Selected Papers*, volume 11975 of *Lecture Notes in Computer Science*, pages 165–174. Springer, 2019.
50. Neal Koblitz and Alfred Menezes. Critical perspectives on provable security: Fifteen years of "another look" papers. *Adv. Math. Commun.*, 13(4):517–558, 2019.

51. Stephan Krenn, Omid Mir, and Daniel Slamanig. Structure-preserving compressing primitives: Vector commitments, accumulators and applications. Cryptology ePrint Archive, Paper 2024/1619, 2024.
52. Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.
53. Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *arXiv preprint arXiv:1112.3333*, 2011.
54. Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. In Simone Severini and Fernando G. S. L. Brandão, editors, *8th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2013, May 21-23, 2013, Guelph, Canada*, volume 22 of *LIPIcs*, pages 20–34. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2013.
55. Stephen Lacey and Richard Box. A fast, easy sort. *Byte*, 16(4):315–ff, 1991.
56. J. C. Lagarias, H. W. Lenstra, Jr., and C.-P. Schnorr. Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348, 1990.
57. H. W. Lenstra. Integer programming with a fixed number of variables. Mathematics of Operations Research, Vol. 8, No. 4 (Nov., 1983), pp. 538-548, 1983.
58. László Lovász and Herbert E. Scarf. The generalized basis reduction algorithm. *Math. Oper. Res.*, 17(3):751–764, 1992.
59. Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 575–584, 2007.
60. Olivia Di Matteo, Vlad Gheorghiu, and Michele Mosca. Fault-tolerant resource estimation of quantum random-access memories. *IEEE Transactions on Quantum Engineering*, 1:1–13, 2020.
61. Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. *SIAM J. Comput.*, 42(3):1364–1391, 2013.
62. Archimedes Pavlidis and Dimitris Gizopoulos. Fast quantum modular exponentiation architecture for shor's factoring algorithm. *Quantum Inf. Comput.*, 14(7-8):649–682, 2014.
63. Chris Peikert. He gives c-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 463–492. Springer, 2020.
64. Oded Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space, 2004.
65. Charles Yuan and Michael Carbin. Tower: data structures in quantum superposition. *Proceedings of the ACM on Programming Languages*, 6(OOPSLA2):259–288, 2022.
66. Mark Zhandry. Quantum money from abelian group actions. In Venkatesan Guruswami, editor, *15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA*, volume 287 of *LIPIcs*, pages 101:1–101:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

A More on N-ary lattices

In this appendix, we prove our results pertaining to N-ary lattices of the form $\Lambda(\boldsymbol{y})$, namely Lem. 3.2, Lem. 3.3, and Lem. 5.5.

Lem. 3.2. *Given $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_N^k$ uniformly random, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is almost uniformly random over $\mathbb{Z}_N$, in the sense that*

$$\Pr_{\boldsymbol{x},\boldsymbol{y}}(\langle \boldsymbol{x}, \boldsymbol{y} \rangle \equiv \alpha \pmod N) = \begin{cases} N^{-1} - N^{-(k+1)} & \text{if } \alpha \not\equiv 0 \pmod N, \\ N^{-1} + N^{-k} - q^{-(k+1)} & \text{if } \alpha \equiv 0 \pmod N. \end{cases}$$

Let $\alpha \in \mathbb{Z}_N$ be uniformly random. Then, $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}|] = (2M+1)^k/N$.

Proof. Recall that we assume that N is prime (otherwise, the proof can be adapted). First assume that $\boldsymbol{x} \neq 0 \pmod N$ – the generic case – with, say, $x_j \not\equiv 0 \pmod N$. Then, the equation $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \equiv \alpha \pmod N$ implies that

$$y_j \equiv x_j^{-1} \left(\alpha - \sum_{i \neq j} x_i y_i \right) \quad \text{in } \mathbb{Z}_N.$$

This shows that if $(k-1)$ coordinates of $\boldsymbol{y}$ are chosen, only one choice of the last coordinate allows for the equation to be satisfied. There are N^{k-1} choices for the first $(k-1)$ coordinates, and N^k total choices for $\boldsymbol{y}$. Thus,

$$\Pr_{\boldsymbol{x},\boldsymbol{y}}(\langle \boldsymbol{x}, \boldsymbol{y} \rangle \equiv \alpha \pmod N \mid \boldsymbol{x} \not\equiv 0 \pmod N) = \frac{N^{k-1}}{N^k} = \frac{1}{N}.$$

For the special case, $\boldsymbol{x} \equiv 0 \pmod N$. In this case, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \equiv 0 \pmod N$, so if $\alpha \equiv 0 \pmod N$, the probability that a random $\boldsymbol{y}$ satisfies the equation is 1, while if $\alpha \not\equiv 0 \pmod N$, this probability is 0. The distribution of residues then follows by the law of total probability.

If X is a uniform random variable over $\mathbb{Z}_N$, we have for each $\alpha \in \mathbb{Z}_N$, $\Pr(X = \alpha) = 1/N$. Let $S \subset \mathbb{Z}^k$. For $\boldsymbol{y}$ fixed, the number $|\Lambda_X(\boldsymbol{y}) \cap S|$ is a random variable, with expectation

$$\begin{aligned} \mathbf{E}_X[|\Lambda_X(\boldsymbol{y}) \cap S|] &= \sum_{\alpha \in \mathbb{Z}_N} \Pr(X = \alpha) \cdot |\Lambda_\alpha(\boldsymbol{y}) \cap S| \\ &= \frac{1}{N} \sum_{\alpha \in \mathbb{Z}_N} |\Lambda_\alpha(\boldsymbol{y}) \cap S| = \frac{|S|}{N}. \end{aligned}$$

This equality follows because the sets $\Lambda_\alpha(\boldsymbol{y}) \cap S$ form a partition of S. The result follows by taking $S = B_M$ which has cardinality $(2M+1)^k$. □

Lem. 3.3. *If $\boldsymbol{y}$ and α are uniformly random over $\mathbb{Z}_N^k$ and $\mathbb{Z}_N$ respectively, then*

$$\mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|] = \frac{(2M+1)^k}{N}, \quad \mathbf{Var}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|] = \left(1 - \frac{1}{N}\right) \mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|].$$

Further, if $1 - N^{-1} \le \mathbf{E}_{\boldsymbol{y},\alpha}[|S_\alpha^{\boldsymbol{y}}|]$, *then*

$$\Pr_{\alpha,\boldsymbol{y}}(|S_\alpha^{\boldsymbol{y}}| \ge 1) \ge \frac{1}{2}.$$

Proof. Lem. 3.2 (and its proof) show that, conditioned on $\boldsymbol{y}$, we have the conditional expectation $\mathbf{E}_\alpha[|S_\alpha^{\boldsymbol{y}}| \mid \boldsymbol{y}] = (2M+1)^k/N$, so by the law of total expectation, we immediately get $\mathbf{E}_{\alpha,\boldsymbol{y}}[|S_\alpha^{\boldsymbol{y}}|] = (2M+1)^k/N$. Note that the proof of Lem. 3.2 also shows that for $\boldsymbol{x} \not\equiv \mathbf{0} \pmod N$ fixed, we have

$$\Pr_{\boldsymbol{y}}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \equiv \alpha \pmod N) = \frac{1}{N},$$

and for $\boldsymbol{x} \equiv \mathbf{0} \pmod N$, if $\alpha \equiv 0 \pmod N$, the probability that a random $\boldsymbol{y}$ satisfies the equation is 1, while if $\alpha \not\equiv 0 \pmod N$, this probability is 0.

Now we count solutions. We express $|S_\alpha^{\boldsymbol{y}}|$ as a random variable, a sum of indicator variables:

$$|S_\alpha^{\boldsymbol{y}}| = \sum_{\boldsymbol{x} \in B_M} I_{\boldsymbol{x}}, \quad I_{\boldsymbol{x}} = \mathbf{1}\{\langle \boldsymbol{x}, \boldsymbol{y}\rangle \equiv \alpha \pmod N\}.$$

(We suppress the dependence on α in the notation of $I_{\boldsymbol{x}}$ for simplicity.) We immediately have $\mathbf{Var}_{\alpha,\boldsymbol{y}}(I_{\boldsymbol{x}}) = (N-1)/N^2$. It is not hard to see that the variables $I_{\boldsymbol{x}}$ and $I_{\boldsymbol{x}'}$ are independent for distinct nonzero $\boldsymbol{x} \ne \boldsymbol{x}' \pmod N$, from which we deduce that $\mathbf{E}_{\alpha,\boldsymbol{y}}[I_{\boldsymbol{x}} I_{\boldsymbol{x}'}] = \Pr_{\alpha,\boldsymbol{y}}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle \equiv \alpha \pmod N)^2 = \mathbf{E}[I_{\boldsymbol{x}}]\mathbf{E}[I_{\boldsymbol{x}'}]$. We conclude that the variance of the sum is the sum of the variances, whence

$$\mathbf{Var}_{\alpha,\boldsymbol{y}}(|S_\alpha^{\boldsymbol{y}}|) = \sum_{\boldsymbol{x} \in B_M} \mathbf{Var}_{\alpha,\boldsymbol{y}}(I_{\boldsymbol{x}}) = |B_M| \cdot \frac{N-1}{N^2},$$

and the result follows.

Write $X = |S_\alpha^{\boldsymbol{y}}|$ to simplify notation. Let $\mu = \mathbf{E}_{\alpha,\boldsymbol{y}}[X]$, with $s = |B_M|$ so that $\mu = s/N$. Then, $\mathbf{E}[X^2] = \mathbf{Var}(X) + \mu^2 + (1 - N^{-1})\mu$. The Payley-Zygmund inequality implies that

$$\Pr(X > 0) \ge \frac{\mathbf{E}[X]}{\mathbf{E}[X^2]} = \frac{\mu}{\mu + 1 - N^{-1}}.$$

So if $1 - N^{-1} \le \mu \le 1$, we get $\Pr(X > 0) \ge \frac{1}{2}$. □

Lem. 5.5. *The following upper bound always holds:*

$$\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \le N^{1/k}.$$

Assume that $\boldsymbol{y}$ *is uniformly random. Fix* $1/M \le \epsilon \le 1$. *Then,*

$$\mathbf{E}\left[\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y}))\right] \ge \frac{k}{k+1} \cdot \frac{N^{1/k}}{2+\epsilon},$$

$$\Pr\left(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \ge \frac{\gamma}{2+\epsilon} N^{1/k}\right) \ge 1 - \gamma^k \quad (0 < \gamma < 1).$$

Proof. The upper bound follows from Minkowski's first theorem (if a symmetric convex set has volume greater than $2^k \det(\Lambda(\boldsymbol{y}))$, then it must contain a nonzero lattice point). The box $B^{(\infty)}(r) = [-r, r]^k$ has volume $(2r)^k$. Put $r = |\det(\Lambda(\boldsymbol{y}))|^{1/k}$. Then, $B^{(\infty)}(r)$ has volume $(2r)^k = 2^k|\det(\Lambda(\boldsymbol{y}))|$. By Minkowski's theorem, $(\Lambda(\boldsymbol{y}) \setminus \{0\}) \cap B^{(\infty)}(r) \neq \emptyset$, so there is a nonzero lattice point, and we conclude that $\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \leq r = N^{1/k}$.

For the expectation. Let $N_M = N_M^*(y; 0)$ be the number of nonzero lattice points in $B_M \cap \Lambda(\boldsymbol{y})$, which is a random variable. Recall that $\mathbf{E}[N_M] = s/N$, where $s = |B_M \setminus \{0\}|$. Then,

$$\Pr(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \leq M) = \Pr(N_M \geq 1) \leq \mathbf{E}[N_M] = \frac{(2M+1)^k - 1}{N}.$$

Let $\epsilon \geq 1/M$. Then,

$$\Pr(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) \leq M) = \frac{[(2+\epsilon)M]^k}{N}.$$

This is ≤ 1 for $0 \leq M \leq N^{1/k}/(2+\epsilon) = R$. Then,

$$\begin{aligned}
\mathbf{E}[\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y}))] &= \int_0^\infty \Pr(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) > M) dM \\
&\geq \int_0^R \left(1 - \frac{[(2+\epsilon)M]^k}{N}\right) dM \\
&= \frac{k}{k+1}\frac{N^{1/k}}{2+\epsilon}.
\end{aligned}$$

Further, if we set $M = \gamma N^{1/k}/(2+\epsilon)$, we get

$$\Pr\left(\lambda_1^{(\infty)}(\Lambda(\boldsymbol{y})) > \frac{\gamma}{2+\epsilon} N^{1/k}\right) \geq 1 - \gamma^k.$$

B The probability of success of the quantum algorithm

We now give the proof of the lower bound for the probability of success of our quantum algorithm (Prop. 3.4).

Prop. 3.4. *With $\boldsymbol{y}$ and α uniformly random over $\mathbb{Z}_N^k$ and $\mathbb{Z}_N$ respectively, we have*

$$\Pr(\text{success}) \geq \frac{N}{(2M+1)^k} \cdot \left[\Pr_{\alpha,\boldsymbol{y}}(|S_\alpha^{\boldsymbol{y}}| \geq 1)\right]^2 \cdot (P_{\text{succ,feas}}^C)^2.$$

In particular, if $1 - N^{-1} \leq \mathbf{E}_{\boldsymbol{y}}[|S_\alpha^{\boldsymbol{y}}|] \leq 1$, then

$$\Pr(\text{success}) \geq \frac{1}{4} \cdot \left(P_{\text{succ,feas}}^C\right)^2.$$

Proof. In our case, since the output of our algorithm is in a pure state, the probability of success (the probability that the output is $|\mathbf{0}, z\rangle$) is given by the square of the amplitude of the inner product of the output ψ_{out} with the desired solution. By duality, we compute

$$O_C \circ (\mathrm{Id} \otimes \mathrm{QFT}_{\mathbb{Z}_N})(|\mathbf{0}, z\rangle) = \frac{1}{N^{1/2}} \sum_{\beta \in \mathbb{Z}_N} \omega^{z\beta} \, |C_{\boldsymbol{y}}(\beta), \beta\rangle \, .$$

By orthogonality, only the terms with $\alpha = \beta$ survive (and for those, $\omega^{-z\alpha}\omega^{z\alpha} = 1$), and we have $\sum_{\boldsymbol{i} \in S_\alpha^{\boldsymbol{y}}} \langle \boldsymbol{i}, \alpha \mid C_{\boldsymbol{y}}(\alpha), \alpha\rangle = \mathbf{1}_{\{C_{\boldsymbol{y}}(\alpha) \in S_\alpha^{\boldsymbol{y}}\}}$ (since we only output one solution). Thus, we finally get

$$\langle \psi_{\text{out}} \mid \mathbf{0}, z\rangle = \frac{1}{N^{1/2}(2M+1)^{k/2}} \sum_{\alpha \in \mathbb{Z}_N} \mathbf{1}_{\{C_{\boldsymbol{y}}(\alpha) \in S_\alpha^{\boldsymbol{y}}\}}.$$

The probability of success is therefore

$$\begin{aligned}
\Pr(\text{success} \mid \boldsymbol{y}) &= \frac{N}{(2M+1)^k} \left(\frac{1}{N} \sum_{\alpha \in \mathbb{Z}_N} \mathbf{1}_{\{C_{\boldsymbol{y}}(\alpha) \in S_\alpha^{\boldsymbol{y}}\}} \right)^2 \\
&= \frac{N}{(2M+1)^k} \mathbf{E}_\alpha[\mathbf{1}_{\{C_{\boldsymbol{y}}(\alpha) \in S_\alpha^{\boldsymbol{y}}\}}]^2 \\
&= \frac{N}{(2M+1)^k} \cdot \Big(\Pr_\alpha(C_{\boldsymbol{y}}(\alpha) \in S_\alpha^{\boldsymbol{y}} \mid \boldsymbol{y}) \Big)^2 .
\end{aligned}$$

For $\boldsymbol{y}$ and α as in the algorithm and, let $\mathbf{s}(\boldsymbol{y}, \alpha) = 1$ if O_C succeeds and 0 otherwise, that is, $\mathbf{s}(\boldsymbol{y}, \alpha) = 1$ if $O_C(|\mathbf{0}, \alpha\rangle) = |\boldsymbol{i}, \alpha\rangle$ with $\boldsymbol{i} \in S_\alpha^{\boldsymbol{y}}$, and $\mathbf{s}(\boldsymbol{y}, \alpha) = 0$ otherwise. Then, we see from (5) that

$$p_{\boldsymbol{y}} = \mathbf{E}_\alpha[\mathbf{s}(\boldsymbol{y}, \alpha) \mid \boldsymbol{y}] = \Pr_\alpha\big(O_C \text{ succeeds at } (\boldsymbol{y}, \alpha) \mid \boldsymbol{y}\big).$$

Since $\mathbf{s}(\boldsymbol{y}, \alpha)$ is an indicator variable, it is now clear that the definition we gave of the success probability is natural:

$$P_{\text{succ}}^C = \mathbf{E}_{\boldsymbol{y}}[p_{\boldsymbol{y}}] = \mathbf{E}_{\alpha, \boldsymbol{y}}[\mathbf{s}(\boldsymbol{y}, \alpha)]$$

and

$$\Pr(\text{success}) = \mathbf{E}_{\boldsymbol{y}}[\Pr(\text{success} \mid \boldsymbol{y})] = \frac{N}{(2M+1)^k} \mathbf{E}_{\boldsymbol{y}}[p_{\boldsymbol{y}}^2].$$

Then, we conclude from the Cauchy-Schwarz inequality that

$$\Pr(\text{success}) \geq \frac{N}{(2M+1)^k} (P_{\text{succ}}^C)^2.$$

This completes the proof. □

C A Transference Theorem

In this section, we will give a proof of a transference theorem with explicit bounds. In this appendix, the dimension is n, instead of k throughout the paper.

Thm. 5.4. *For any lattice L and its dual lattice L^*, the successive minima satisfy*

$$\lambda_i^{(1)}(L) \cdot \lambda_{n-i+1}^{(\infty)}(L^*) \leq C(n) \cdot (n \ln n)^{1/2},$$

where

$$C(n) = \sqrt{\frac{6}{\pi \ln n}} + \frac{2\sqrt{3}}{\pi}.$$

Note that earlier in the paper, we stated Thm. 5.4 directly for the dual lattice, as it is the one that we are interested in. Banaszczyk proved this upper bound with a worse upper bound $C \cdot n(\ln n)^{1/2}$ and without an explicit constant [10, Proposition 3.5].

The first transference theorem available was proved by Lagarias, Lenstra, and Schnorr [56, Theorem 2.4]. We keep the previous notation for successive minima, that is, $\lambda_i^{(p)}(L)$ denotes the i-th successive minimum of the lattice L in the ℓ^p norm. If L is a lattice in $\mathbb{R}^n$, we denote by L^* its dual (reciprocal, polar) lattice. *Hermite's constant* γ_n is defined by

$$\gamma_n = \sup\{\lambda_1(L)^2 d(L)^{-2/n} : L \text{ is a lattice of rank } n\}.$$

Minkowski's convex body theorem implies that $\gamma_n \leq 4\pi^{-1}\Gamma(1+n/2)^{2/n}$ (see Cassels [27, IX.7]), which yields $\gamma_n \leq 2n/3$ for all $n \geq 2$. It is known that

$$\frac{n}{2\pi e}(1+o(1)) \leq \gamma_n \leq \frac{n}{\pi e}(1+o(1)) \quad \text{as } n \to \infty.$$

To get a non-decreasing function on n, we define $\gamma_n^* = \max\{\gamma_i : 1 \leq i \leq n\}$. Then, $\gamma_n^* \leq 2n/3$ for all $n \geq 2$.

Lagarias, Lenstra, Schnorr [56, Theorem 2.4] proved that the successive minima of a lattice L of rank n and its reciprocal lattice L^* satisfy

$$1 \leq \lambda_i^{(2)}(L)\lambda_{n-i+1}^{(2)}(L^*) \leq \frac{\sqrt{(i+3)\cdot(n-i+4)}}{4} \cdot \gamma_n^* \quad (1 \leq i \leq n).$$

Banaszczyk [9, Theorem (2.1)] improved this result and proved that

$$\lambda_i(L)\lambda_{n-i+1}(L^*) \leq n \quad (1 \leq i \leq n).$$

Rather than using basic inequalities between ℓ^2 and ℓ^p norms, we are interested in obtaining directly ℓ^1/ℓ^∞ versions of these transference theorems via duality. Note that the definition of polar body given in [9] differs from that given in [10] but the two coincide for convex symmetric bodies; in [10] the non-standard definition with the absolute value is given, but it is only applied to symmetric bodies.

Banaszczyk [10, Proposition 3.5] proved that there exists a numerical constant $C > 0$ such that

$$\lambda_i^{(1)}(L)\lambda_{n-i+1}^{(\infty)}(L^*) \le C\, n\, (\log n)^{1/2} \quad (1 \le i \le n).$$

In this section, we will prove a stronger result with an explicit constant. Although this result is not stated explicitly in [10] it can be obtained by similar methods. We will give the proof for completeness.

Let $L \subset \mathbb{R}^n$ be a lattice. We denote the set of full rank lattices of $\mathbb{R}^n$ by L_n. Thus, L is generated by n linearly independent vectors. Given a lattice $L \subset \mathbb{R}^n$, we define the *dual lattice* (or *polar lattice* or *reciprocal lattice*) L^* by:

$$L^* = \{\boldsymbol{u} \in \mathbb{R}^n : \boldsymbol{u}, \boldsymbol{v} \ \in \mathbb{Z},\ \forall \boldsymbol{v} \in L\},$$

where $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ is the canonical inner product in $\mathbb{R}^n$. We have $L^{**} = L$.

A *convex body* in $\mathbb{R}^n$ is a compact convex subset of $\mathbb{R}^n$ with nonempty interior. We consider only convex bodies in $\mathbb{R}^n$ which are *symmetric* with respect to zero, so any convex body in what follows is assumed symmetric. Given a convex body $U \subset \mathbb{R}^n$, we define the *polar body* U^0 in the usual way:

$$U^0 = \{\boldsymbol{u} \in \mathbb{R}^n : |\langle \boldsymbol{u}, \boldsymbol{v} \rangle| \le 1,\ \forall \boldsymbol{v} \in U\}.$$

Since we are assuming symmetry, observe that the absolute value in the definition can be dropped. A convex body and its polar body satisfy $(U^0)^0 = U$. Observe also that if $B^{(p)}$ denotes the unit ball in the ℓ^p norm, then $(B^{(p)})^0 = B^{(q)}$ where $(1/p) + (1/q) = 1$. In particular, for ℓ^1/ℓ^∞ duality, we have $(B^{(1)})^0 = B^{(\infty)}$.

For U symmetric, by $\| \ \|_U$ we denote the norm on $\mathbb{R}^n$ induced by U (the *Minkowski functional* of U):

$$\|\boldsymbol{x}\|_U = \inf\{\lambda > 0 : \boldsymbol{x} \in \lambda U\}.$$

Thus, by definition, U is the unit ball in the norm $\| \ \|_U$.

By $\operatorname{span} A$ we denote the linear subspace of $\mathbb{R}^n$ spanned by a subset A. Given a lattice $L \subset \mathbb{R}^n$ and a convex body $U \subset \mathbb{R}^n$, we define

$$\lambda_i(L, U) = \min\{r > 0 : \dim \operatorname{span}(L \cap rU) \ge i\} \quad (i = 1, \ldots, n).$$

These are called the *successive minima* of L with respect to U. For any $\alpha > 0$, one verfies easily that

$$\lambda_i(L, aU) = \frac{1}{a} \cdot \lambda_i(L, U). \tag{13}$$

We also note that if $\| \ \|_p$ denotes the ℓ^p norm on $\mathbb{R}^n$ $(1 \le p \le \infty)$, then

$$\lambda_i^{(p)}(L) = \lambda_i(L, B^{(p)}).$$

For any convex body $U \subset \mathbb{R}^n$ we define

$$\boldsymbol{\tau}(U) = \sup_{L \in L_n} \max_{1 \le i \le n} \lambda_i(L, U)\lambda_{n-i+1}(L^*, U^0).$$

More generally, instead of considering just one convex body U and its polar body U^0, we consider a pair of independent symmetric convex bodies $U, V \subset \mathbb{R}^n$, and define

$$\boldsymbol{\tau}(U,V) = \sup_{L \in L_n} \max_{1 \le i \le n} \lambda_i(L,U)\lambda_{n-i+1}(L^*,V).$$

Of course, $\boldsymbol{\tau}(U,U^0) = \boldsymbol{\tau}(U)$.

The starting point of our transference theorem is the following result [10, Thm 3.2].

Theorem C.1. *Let $a_1, \dots, a_n$ be arbitrary positive numbers. Denote*

$$P = \{(x_1, \dots, x_n) \in \mathbb{R}^n : |x_k| \le a_k \text{ for } k = 1, \dots, n\}.$$

Let t_0 be the root of the equation

$$\sum_{k=1}^{n} e^{-\pi(\sqrt{ta_k + \frac{1}{4}} - \frac{1}{2})^2} = \frac{1 - e^{-\pi}}{6}.$$

Then $\boldsymbol{\tau}(B^{(\infty)}, P) \le t_0$.

From this theorem, it is not hard to obtain the bound $\boldsymbol{\tau}(B^{(\infty)}, B^{(1)}) \le C(n) \cdot (n \ln n)$ for some constant $C(n)$. To improve this, we need a comparison result [10, Lemma 1.6].

For convex bodies U and V, we define $\alpha(U)$ and $\beta(V)$ as follows. For a discrete set A, we define

$$\varrho(A) = \sum_{x \in A} e^{-\pi \|\boldsymbol{x}\|^2} \quad (A \subset \mathbb{R}^n).$$

Let L be a full-rank lattice in $\mathbb{R}^n$. By σ_L we denote the probability measure on L given by the formula

$$\sigma_L(A) = \frac{\varrho(A)}{\varrho(L)} \quad (A \subset L).$$

Let U be a symmetric convex body in $\mathbb{R}^n$. We define

$$\alpha(U) = \sup_{L \in L_n} \frac{\varrho(L \backslash U)}{\varrho(L)} = \sup_{L \in L_n} \sigma_L(L \backslash U),$$
$$\beta(U) = \sup_{L \in L_n} \sup_{u \in \mathbb{R}^n} \frac{\varrho((u+L) \backslash U)}{\varrho(L)}.$$

The following lemma [10, Lem 1.6] shows how to use α and β to derive an upper bound on $\boldsymbol{\tau}(U,V)$.

Lemma C.2. *Let $B^{(2)}$ denote the unit Euclidean ball in $\mathbb{R}^n$. Let U and V be convex symmetric bodies. If $2\alpha(U) + \beta(V) \le 1 - e^{-\pi}$, then $\boldsymbol{\tau}(U, V + B^{(2)}) \le 1$.*

The final ingredient is the following lemma which gives estimates on α and β [10, Lemmas 2.8 and 2.10].

Lemma C.3. *For any $r \geq \sqrt{n/2\pi}$ and any $s > 0$, we have*

$$\alpha(rB^{(2)}) < \left(\frac{2\pi e}{n}\right)^{n/2} r^n e^{-\pi r^2}, \quad \beta(sB^{(\infty)}) < 2ne^{-\pi s^2}.$$

A final observation regards the behaviour of $\boldsymbol{\tau}(U, V)$ with respect to inclusion and scaling. Given two convex bodies V and V', we have $V \subset V' \Rightarrow \lambda_i(L, V') \leq \lambda_i(L, U)$. Thus,

$$V \subset V' \quad \Rightarrow \quad \boldsymbol{\tau}(U, V') \leq \boldsymbol{\tau}(U, V).$$

Similarly, if U and U' are two convex bodies, then

$$U \subset U' \quad \Rightarrow \quad \boldsymbol{\tau}(U', V) \leq \boldsymbol{\tau}(U, V).$$

Regarding scaling, since $\lambda_i(L, aU) = a^{-1}\lambda_i(L, U)$, we get

$$\boldsymbol{\tau}(rU, sV) = \frac{1}{rs}\boldsymbol{\tau}(U, V). \tag{14}$$

Finally, we are ready to give the proof of Theorem 5.4. A key observation is that for any $s > 0$, we have $sB^{(\infty)} + B^{(2)} \subset (s+1)B^{(\infty)}$. Indeed, if $\|\boldsymbol{x}\|_\infty \leq 1$ and $\|\boldsymbol{y}\|_2 \leq 1$, then

$$\|s\boldsymbol{x} + \boldsymbol{y}\|_\infty \leq |s| \cdot \|\boldsymbol{x}\|_\infty + \|\boldsymbol{y}\|_\infty \leq |s| \cdot \|\boldsymbol{x}\|_\infty + \|y\|_2 \leq s + 1.$$

It follows from this inclusion that

$$\boldsymbol{\tau}(U, (s+1)B^{(\infty)}) \leq \boldsymbol{\tau}(U, sB(\infty) + B^{(2)}).$$

Using the scaling properties of $\boldsymbol{\tau}$ (Eq. (14)), we have

$$\begin{aligned}\boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) &= r(s+1) \cdot \boldsymbol{\tau}(rB^{(2)}, (s+1)B^{(\infty)}) \\ &\leq r(s+1) \cdot \boldsymbol{\tau}(rB^{(2)}, sB^{(\infty)} + B^{(2)}).\end{aligned}$$

Using Lemma C.2, we see that if

$$2\alpha(rB^{(2)}) + \beta(sB^{(\infty)}) \leq 1 - e^{-\pi},$$

then

$$\boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) \leq r(s+1) \cdot \boldsymbol{\tau}(rB^{(2)}, sB^{(\infty)} + B^{(2)}) \leq r(s+1).$$

Now we use the estimates on α and β. Choose s such that $s^2 = \frac{1}{\pi}\ln(Sn)$. Then, $\beta((s-1)B^{(\infty)}) < 2/S$. Now, choose r such that $\pi r^2 = R \cdot n$ with $R \geq \frac{1}{2}$. then, $r^2 \geq \frac{n}{2\pi}$, as required, and

$$\alpha(rB^{(2)}) < \left(\frac{2\pi e r^2}{n}\right)^{n/2} e^{-\pi r^2} = (2eR)^{n/2} e^{-Rn}.$$

If $R \geq 2e$, then $2eR \leq R^2$, so we get a simplified bound

$$\alpha(rB^{(2)}) < (Re^{-R})^n.$$

We conclude that if

$$\frac{2}{S} + (Re^{-R})^n < 1 - e^{-\pi},$$

then $\boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) \leq rs$. We may choose $R = 6 \geq 2e$ and $S = 3$. Then,

$$\frac{2}{S} + (Re^{-R})^n \approx 0.6815 < 1 - e^{-\pi} \approx 0.9568,$$

so we conclude that

$$\boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) \leq r(s+1) = \sqrt{\frac{6n}{\pi}} \cdot \left(1 + \sqrt{\frac{1}{\pi}\ln(3n)}\right).$$

For $n \geq 3$, $\ln(3n) \geq 2\ln(n)$, so we have a simpler upper bound

$$t(n) := \sqrt{\frac{6n}{\pi}} + \sqrt{\frac{12n}{\pi^2}\ln(n)}.$$

We want $t(n) \leq C(n)n^{1/2}(\ln n)^{1/2}$ for some explicit $C(n)$. We have

$$\frac{t(n)}{n^{1/2}(\ln n)^{1/2}} = \left(\frac{6}{\pi \ln n}\right)^{1/2} + \frac{2\sqrt{3}}{\pi}.$$

This is a decreasing function of n, so we conclude that for any $n \geq 2$, we have

$$\boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) \leq C(n)n^{1/2}(\ln n)^{1/2},$$

where

$$C(n) = \sqrt{\frac{6}{\pi \ln n}} + \frac{2\sqrt{3}}{\pi}. \tag{15}$$

We observe that the upper bound improves as n grows; the limit of the constant $C(n)$ is $\lim_{n\to\infty} C(n) = 2\sqrt{3}/\pi \approx 1.103$.

Finally, we observe that $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{n}\|\boldsymbol{x}\|_2$, and therefore $B^{(2)} \subset B^{(1)}$, so

$$\boldsymbol{\tau}(B^{(1)}, B^{(\infty)}) \leq \boldsymbol{\tau}(B^{(2)}, B^{(\infty)}) \leq C(n) \cdot (n \ln n)^{1/2}.$$

This means that for any lattice L,

$$\lambda_i^{(1)}(L) \cdot \lambda_{n-i+1}^{(\infty)}(L^*) \leq C(n) \cdot (n \ln n)^{1/2}.$$

This completes the proof of Theorem 5.4. □

Algorithm 9 Classical many collision algorithm

Input: A function $f : \{1, \ldots, 2^n\} \to \{1, \ldots, 2^m\}$.
Output: 2^t collisions.
1: **for** i from 1 to $2^{(t+m+1)/2}$
2: $\quad S_i = \{\}$
3: **end for**
4: **for** i from 1 to $2^{(t+m+1)/2}$
5: $\quad S_{f(i) \bmod 2^{(t+m+1)/2}} = \{(i, f(i))\} \cup S_{f(i) \bmod 2^{(t+m+1)/2}}$
6: **end for** ▷ The elements with the same image by f are now next to each other
7: Walk through the different S_i and list the pairs of elements with the same output by f in S'.
8: Return S'

D More on Collision Finding

Lemma 6.2. For a random function $f : \{1, \ldots, n\} \to \{1, \ldots, m\}$, the number of collisions $\#\{\{i, j\} : i \neq j, f(i) = f(j)\}$ has expected value $\frac{n(n-1)}{2m}$ and variance of $\frac{n(n-1)}{2m}\left(1 - \frac{1}{m}\right)$.

Proof. Let $X = \sum_{1 \leq i < j \leq n} X_{ij}$, where each X_{ij} is the indicator variable $X_{ij} = \mathbf{1}_{\{f(i)=f(j)\}}$. Thus, X is exactly the total number of collisions.

Given $\alpha \in \{1, \ldots, m\}$, we have $\mathbf{P}(f(i) = f(j) = \alpha) = 1/m^2$, since f is uniformly random, and there are m^2 choices for the pair $(f(i), f(j))$. It follows that

$$\mathbf{P}(f(i) = f(j)) = \sum_{\alpha=1}^{m} \mathbf{P}(f(i) = f(j) = \alpha) = m \cdot (1/m^2) = 1/m,$$

and we conclude that

$$\mathbf{E}[X] = \sum_{1 \leq i < j \leq n} \mathbf{E}[X_{ij}] = \sum_{1 \leq i < j \leq n} \mathbf{P}(f(i) = f(j)) = \binom{n}{2} \cdot \frac{1}{m} = \frac{n(n-1)}{2m}.$$

To compute the variance, we need to compute

$$\mathbf{E}[X^2] = \sum_{1 \leq i < j \leq n} \mathbf{E}[X_{ij}^2] + 2 \sum_{1 \leq i < j < k < \ell \leq n} \mathbf{E}[X_{ij} X_{k\ell}].$$

Since $X_{ij}^2 = X_{ij}$, the first sum is exactly $\mathbf{E}[X]$. Put $s = \binom{n}{2}$, so that the first sum is $\mathbf{E}[X] = s/m$. It remains to compute $\mathbf{E}[X_{ij} X_{k\ell}]$ when $\{i, j\}$ and $\{k, \ell\}$ are *two distinct unordered pairs*. Note that since we have s total unordered pairs $\{i, j\}$, the number of such distinct unordered pairs is $\binom{s}{2}$.

Case 1. If $\{i, j\} \cap \{k, \ell\} = \emptyset$. Then, by independence and the above calculation,

$$\mathbf{E}[X_{ij} X_{k\ell}] = \mathbf{P}(f(i) = f(j), f(k) = f(\ell))$$

$$= \mathbf{P}(f(i) = f(j))\mathbf{P}(f(k) = f(\ell)) = \frac{1}{m} \cdot \frac{1}{m} = \frac{1}{m^2}.$$

Case 2. $|\{i, j\} \cap \{k, \ell\}| = 1$, say $j = k$, but i, j, ℓ all distinct. Then, $f(i)$, $f(j)$, $f(\ell)$ are three independent choices in $\{1, \ldots, m\}$, so

$$\begin{aligned}\mathbf{P}(f(i) = f(j) = f(\ell)) &= \sum_{\alpha=1}^{m} \mathbf{P}(f(i) = f(j) = f(\ell) = \alpha) \\ &= m \cdot (1/m^3) = 1/m^2.\end{aligned}$$

In both cases we thus have $\mathbf{E}[X_{ij}X_{k\ell}] = 1/m^2$, and we conclude that the second sum contribution to $\mathbf{E}[X^2]$ is

$$2 \sum_{1 \le i < j < k < \ell \le n} \mathbf{E}[X_{ij}X_{k\ell}] = 2\binom{s}{2} \cdot \frac{1}{m^2} = \frac{s(s-1)}{m^2}.$$

The variance is thus

$$\begin{aligned}\mathbf{Var}(X) &= \mathbf{E}[X^2] - (\mathbf{E}[X])^2 \\ &= \frac{s}{m} + \frac{s(s-1)}{m^2} - \frac{s^2}{m^2} = \frac{s(m-1)}{m^2} = \frac{n(n-1)(m-1)}{2m^2}.\end{aligned}$$

Lemma 6.3. Let the notation be as in Alg. 6. Using classical many collisions finding, there exists a quantum procedure for sieving that takes a list of s_{i-1} elements of norms less than R_{i-1}, and outputs a list of s_i elements of norms less than R_i in time

$$\frac{1}{\log_2(e)}(\log_2(s_i) + (k+1)\log_2(2R_{i-1}/R_i) + 1)\sqrt{2s_i}\left(\frac{2R_{i-1}}{R_i}\right)^{(k+1)/2}$$

whenever $s_i \le \frac{s_{i-1}^2}{2}\left(\frac{R_i}{2R_{i-1}}\right)^{k+1}$.

Proof. The classical many collisions finding algorithm (Alg. 9) takes a function $f : \{1, \ldots, 2^n\} \to \{1, \ldots, 2^m\}$ and returns 2^t collisions.

It begins by sorting (with merge-sort for consistency between best and worst cases) $2^{(t+m+1)/2}$ elements depending on their output under f. Then it retrieves the collisions – the corresponding elements will be next to each other. This is an implementation of a hash table. Its complexity is $\frac{1}{\log_2(e)}(t + m + 1)2^{(t+m+1)/2}$ bit operations.

We now apply this cost evaluation of the many collision finding algorithm to the function $x \mapsto \lfloor 2x/R_i \rceil$ for sieving (using the notation of Alg. 6). Recall, the sieving procedure takes a list of s_{i-1} elements of norm less than R_{i-1} and outputs a list of s_i elements of norm less than R_i. Substituting $2^n, 2^m$ and 2^t by s_{i-1}, $\left(\frac{2R_{i-1}}{R_i}\right)^{k+1}$ and s_i, respectively, yields the result of the lemma. □

D.1 Quantum many collision finding algorithm

We analyse the non-asymptotic complexity of the quantum algorithm for many collisions finding given by Bonnetain, Chailloux, Schrottenloher and Shen [22]. This algorithm is based on the analysis of quantum walks from Magniez, Nayak, Roland and Santha [59] and the use of quantum walks for collisions from Ambainis [6].

Quantum walks The quantum walk algorithm is quantum search analogous to Grover's algorithm [43] where the set of the search is a regular graph. For further use, we note the proportion of desired vertices ϵ and the spectral gap of the graph δ.

Setup cost S**.** The cost of producing the uniform superposition of vertices.
Update cost U**.** The cost of making the uniform superposition of neighbors of a vertex.
Checking cost C**.** The cost of checking if a vertex is a desired one.

From [59], for an integer s, the total cost of a quantum walk is less than (but close to)

$$S + \frac{\pi}{4\sqrt{\epsilon}}\left(\left(\frac{8\pi}{\sqrt{\delta}}U + \frac{1}{2}\left(\log\frac{2\pi}{\sqrt{\delta}} + s\right)^2 + \frac{3}{2}\left(\log\frac{2\pi}{\sqrt{\delta}} + s\right)\right) + C\right)$$

with success probability more than $(1 - \arcsin\sqrt{\epsilon}) \times \left(1 - \frac{\pi}{2\sqrt{\epsilon}2^s}\right)$.

Quantum walks for collisions. From now on, we consider a function $f : \{0,1\}^n \to \{0,1\}^m$ on which we want to get collisions. The expected number of collisions is roughly 2^{2n-m-1}.

In [6], quantum walks are used for computing collisions. The idea is to apply a quantum walk on a Johnson graph. Let $n, K \in \mathbb{N}$. The Johnson graph $J(2^n, K)$ is the graph of subsets of size K in $\{1, \ldots, 2^n\}$ and two subsets are connected if and only if their intersection is of size $K - 1$. The spectral gap of the normalized adjacency matrix of $J(2^n, K)$ is $\delta = \frac{2^n}{K(2^n-K)}$ (see [26, §12.4.2]). The desired vertices are the ones containing a collision, so $\epsilon \simeq \frac{K^2}{2^{m+1}}$. Indeed, a simple modification of the proof of the expected number of collisions for a random function shows that with K values, the expected number of collisions is $2^{-m}\binom{K}{2}$. The quantity ϵ can be computed as the probability that there is at least one collision, so $\epsilon = 1 - 2^m(2^m - 1)\cdots(2^m - K)/2^{mK}$. This yields $\epsilon \approx K(K-1)/2^{m+1} \approx K^2/2^{m+1}$; this approximation may also be obtained by Poisson approximation, $\epsilon \approx 1 - e^{-\lambda}$, where λ is the expected number of collisions.

The Setup, Update and Checking costs depend on the data structure used for the subsets. We use the quantum radix tree structure from [65] for practical

estimations. Insertion costs $1440k^2+5056k$ gates, and checking if an element is in the tree costs $784k^2+1612k+1$ gates, where k is the height of the tree. Checking is made with the update by checking whether the new element added to the set and the one deleted from the set make a collision or not. For $s = m - 4t + 21$, the cost is (heavily) dominated by the term

$$(1440m^2 + 5056m)K + \frac{2^{m/2}}{\sqrt{K}}(248336m^2 + 744560m)$$

with success probability more than $\left(1 - \frac{K\sqrt{8}}{2^{m/2}}\right)\left(1 - \frac{\pi 2^{m/2}}{K\sqrt{22^s}}\right)$.

Reusable quantum walks for collisions. In [22], a new operation Extract is introduced. The observation is that from the result of the quantum walk, one can extract the collision and the rest can be used as a new setup state for a quantum walk. For 2^t collisions, the total cost is (heavily) dominated by the term

$$(1440m^2 + 5056m)K + \frac{2^{t+m/2}}{\sqrt{K}}(248336m^2 + 744560m)$$

with individual success probability more than $\left(1 - \frac{K\sqrt{8}}{2^{m/2}}\right)\left(1 - \frac{\pi 2^{m/2}}{K\sqrt{22^s}}\right)$. The second term is not dominant from $s = m - 4t + 21$.

This is optimized for

$$K = \left(\frac{2^{t+m/2+1/2}(15521m + 46535)}{90m + 316}\right)^{2/3},$$

giving a total complexity of

$$\frac{3}{2}2^{2t/3+m/3}\left(248336m^2 + 744560m\right)^{2/3}\left(1440m^2 + 5056m\right)^{1/3} \simeq 2^{2t/3+m/3+16.03}(m^2 + 3.17m)$$

with the condition that $K^2/2 \leq 2^m$ which can be approximated to $t \leq m/4 - 8.18$ and an individual success probability higher than $1/2$ (each pair of the output has probability higher than $1/2$ to be a valid collision).

One can ensure the returned elements to be different by adding a verification to the checking cost. This additional cost is negligible.

E CSIDH-1024 parameters

Bonnetain and Schrottenloher [23, Table 3], as well as Peikert [63, Figure 1] also give complexity estimates for the CSIDH-1024 parameter set. Using these analyses, we compare how our version of Childs-van Dam's algorithm compares to Peikert's algorithm in Tab. 6 with access to M shifts.

group action cost estimate	Peikert Alg. [63]	M	Alg. 2 + sieving	Alg. 2 + enumeration
$2^{57.6}$ [23]	$2^{78.4}$ to $2^{85.5}$	2^{12}	$2^{62.9}$	$2^{105.8}$
		2^{16}	$2^{62.6}$	$2^{85.6}$
		2^{20}	$2^{62.2}$	$2^{72.2}$

Table 6. Using the group action estimates from Bonnetain and Schrottenloher for the CSIDH-1024 parameters, we compare the CvD algorithm variants to Peikert's algorithm (based off of Kuperberg) in number of T-gates.

Part III

Efficiency

7 Computing ascending isogenies

Publication data. Steven Galbraith, Valerie Gilchrist, Damien Robert. Improved algorithms for ascending isogeny volcanoes, and applications. LatinCrypt 2025.

My Contribution. In this work I focused on the application sections of the paper (Sections 5 and 6). I also contributed to the writing and development of Section 2, and ran experiments that were not presented in the paper.

Improved algorithms for ascending isogeny volcanoes, and applications

Steven D. Galbraith[1], Valerie Gilchrist[2], Damien Robert[3]

[1] University of Auckland, Auckland, New Zealand
[2] Université Libre de Bruxelles, Brussels, Belgium
[3] Inria Bordeaux, Institut de Mathématiques de Bordeaux, France

Abstract. Given two elliptic curves over $\mathbb{F}_q$, computing an isogeny mapping one to the other is conjectured to be classically and quantumly hard. This problem plays an important role in the security of elliptic curve cryptography. In 2024, Galbraith applied recently developed techniques for isogenies to improve the state-of-the-art for this problem.
In this work, we focus on computing ascending isogenies with respect to an orientation. Our results apply to both ordinary and supersingular curves. We give a simplified framework for computing self-pairings, and show how they can be used to improve upon the approach from Galbraith to recover these ascending isogenies and eliminate a heuristic assumption from his work. We show that this new approach gives an improvement to the overall isogeny recovery when the curves have a small crater (super-polynomial in size). We also study how these self-pairings affect the security of the (PEARL)SCALLOP group action, gaining an improvement over the state-of-the-art for some very particular parameter choices. The current SCALLOP parameters remain unaffected.

1 Introduction

Tate's isogeny theorem [30] states that two elliptic curves defined over $\mathbb{F}_q$ have the same number of points if and only if they are isogenous. To date, it is thought to be a hard problem to recover an isogeny between two fixed elliptic curves. The case where these two curves are supersingular greatly concerns modern-day isogeny-based cryptography, however, the case where they're ordinary still remains pertinent to elliptic curve cryptography and pairing cryptography. In 1999, Galbraith [13] gave algorithms solving the ordinary case, and then improved upon the worst case complexities in [14] using Kani's lemma.

In the later work, given elliptic curves on the floor of an isogeny volcano, Galbraith solves the problem by first computing paths up to the crater from

* Authors listed in alphabetical order: see https://www.ams.org/profession/leaders/CultureStatement04.pdf. Valerie Gilchrist is supported by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium. Damien Robert is supported by the France 2030 program under grant agreement ANR-22-PETQ-0008 PQ-TLS.

Date of this document: 2025-09-02.

each curve, and then solving a meet-in-the-middle search on the crater to complete the path. In this work we focus on how to compute the paths up to the crater, these paths are called *vertical* or *ascending* isogenies. In Section 3.2 we give a new method to determine an ascending isogeny when the endomorphism ring is known. In Section 4, we make use of the work from Castryck et al [5], extended by Macula and Stange in [20], that shows how *self-pairings* can be used to recover unknown *horizontal* isogenies. These are pairings that can be evaluated using only one elliptic curve point and give a non-trivial output, i.e. $e(P,P) \neq 1$. We give a simplified framework for these pairings, and show how they can be used to recover vertical isogenies as well. In Section 4.2 we give a simplified proof of a special case of a result from [20] where we use only the standard Weil pairing rather than sesquilinear pairings. This proof is designed to be accessible to non-experts. We carefully characterize the cases when the order of the torsion subgroup does or does not divide the conductor, as well as the additional case when it divides the overall degree of the secret isogeny. This increases the available torsion subgroups on which we can gain *partial* information about how the isogeny acts. We then show how to encode the *missing* information into a conic equation, and treat the case of degenerate and non-degenerate conics individually.

With this new toolkit to compute unknown vertical isogenies, we consider the computational isogeny problem on certain curves with large volcanos (for example, pairing-friendly ordinary elliptic curves). In this case, the isogeny volcano is tall and the crater is very small (usually of size 1). The main result of [14] is to show that this case can be solved in $\tilde{O}(q^{1/4})$ field operations. In Section 5 we first show how to eliminate a heuristic assumption (about Elkies primes) from Galbraith's work using our pairing construction. We then detail an algorithm for solving the computational isogeny problem in volcanoes that are tall but with crater larger than polynomial size. Denote by Δ the discriminant of the maximal order and suppose $|\Delta| = q^a$ for some $0 < a < 1$. Write $t^2 - 4q = N^2\Delta$, where the curves in the isogeny class have $q + 1 - t$ points. The state-of-the-art algorithm from Galbraith [14] has complexity $\tilde{O}(h_0 N^{1/2})$ operations over $\mathbb{F}_q$ when the class number, h_0, is small enough, which works out as $\tilde{O}(q^{(1+a)/4})$ operations when a is small. Our new approach using self-pairings gives an improved complexity of $\tilde{O}(q^{(1-a)/4})$ operations, for most integers Δ.

In addition, in Section 2.4, we consider the case of volcanoes with small (but super-polynomial) conductor. The result is implicit in [13] but has not been explicitly stated anywhere. We show these two improvements compared to [14] in Figure 1.

Lastly, in Section 6, we demonstrate a second application of our approach to computing vertical isogenies. In this section our focus is on supersingular curves. Namely, we outline an attack on the SCALLOP group action for maximal orders with (almost) smooth discriminants. We show that if the fundamental discriminant Δ has a smooth factor of size at least $\Delta^{1/2+\epsilon}$, then our isogeny recovery approach from self-pairings potentially gives an improvement over the

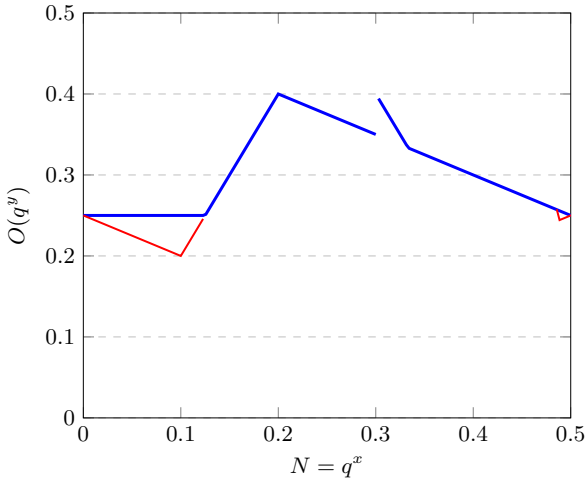


Fig. 1. The state-of-the-art from Galbraith is plotted in blue. The two improvements that will be presented in this paper are plotted in red.

state-of-the-art *classical* attacks. The exact complexity of our attack depends on the exact number and size of the prime factors of the smooth part of Δ.

Self pairings were introduced in [5] as a way to attack isogeny class group actions; they were then extended in [20]. However, they only apply when the degree d of the unknown isogeny $\phi : E_0 \to E_1$ is known. A key feature of our attacks in Section 5 and Section 6 is that we use the self-pairings only to ascend the volcano, and use this to compute an isogeny whose degree is not known. We can always recover the degree of the going up isogeny (and it is usually part of the SCALLOP parameter anyway).

2 Isogeny graphs and previous results on ascending isogenies

We study elliptic curves over a finite field $\mathbb{F}_q$. In some sections of the paper the focus is on ordinary curves and in some sections on supersingular curves.

In the ordinary case, the fundamental problem is to construct an isogeny $\phi : E_0 \to E_1$ between two given curves E_0, E_1 over $\mathbb{F}_q$. We are mainly interested in the case when ϕ is a vertical degree-N isogeny, which means one of the orders $\mathrm{End}(E_0), \mathrm{End}(E_1)$ has index N in the other.

We will sometimes use the notation that $\mathrm{End}(E_0)$ and $\mathrm{End}(E_1)$ have discriminants Δ_0, Δ_1 respectively, and write Δ for the discriminant of the imaginary quadratic field $K = \mathrm{End}(E_0) \otimes \mathbb{Q}$. Note that $\Delta_i = f_i^2 \Delta$ for $i = 0, 1$ and the integer f_i is called the conductor of the ring $\mathrm{End}(E_i)$. Sometimes we will abuse

notation and talk of the discriminant of a curve E when we mean the discriminant of $\mathrm{End}(E)$.

We write π_q for the q-power Frobenius map on E_0 and E_1. If R is a quadratic imaginary order, we will also often denote by $\Delta_R = \Delta(R)$ its discriminant, and likewise if $\gamma \in R$ we will denote by $\Delta_\gamma = \Delta(\gamma)$ the discriminant of its minimal polynomial (which is also the discriminant of $\mathbb{Z}[\gamma]$).

When we talk about supersingular curves we will always take them to be defined over $\mathbb{F}_{p^2}$ and we will assume that we are in the isogeny class of maximal curves, meaning that $\#E(\mathbb{F}_{p^2}) = (p+1)^2$.

2.1 Isogeny volcanos

Let E_0 be an elliptic curve over $\mathbb{F}_q$. Let S be a set of primes ℓ coprime to the characteristic of $\mathbb{F}_q$. The isogeny graph of E_0 with isogeny degrees in S is the graph whose vertex set is elliptic curves E over $\mathbb{F}_q$ that are isogenous to E_0, and there is an edge $\{E, E'\}$ if there exists an isogeny $\psi : E \to E'$ such that $\deg(\psi) \in S$. In the case $S = \{\ell\}$ we call the graph the ℓ-isogeny graph.

The connected components of the ℓ-isogeny graphs of ordinary elliptic curves have a specific structure. They constitute a cycle where each vertex of the cycle is the root of a tree. As can be seen in the example given in Figure 2, this structure resembles a volcano.

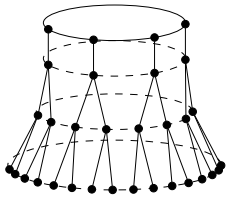


Fig. 2. Example 2-isogeny volcano of depth 3.

The vertices forming the top cycle of the volcano are often called the *crater*, or the *surface* of the volcano. The leaf vertices of the trees are called the *floor* of the volcano. A set of vertices at the same distance from the crater are called a level. The distance of a vertex on the floor from the cycle is called the *depth*. Vertices in the same level correspond to curves whose endomorphism ring has the same discriminant. Vertices on the crater correspond to curves whose discriminant is equal to the fundamental discriminant, Δ (or at least to a ring whose conductor is not divisible by ℓ). In an ℓ-isogeny volcano, a vertex in level k will correspond to a curve with discriminant $\Delta' = \ell^{2k}\Delta$.

In the ordinary case we will also consider volcanos with respect to a set S consisting of all primes dividing the conductor of the ring $\mathbb{Z}[\pi_q]$.

The Computational Isogeny Problem asks to recover an isogeny between two given elliptic curves, E_0, E_1. Taking a walk in the ℓ-isogeny volcano, where the

starting point is the vertex associated to E_0 and the ending point is the vertex associated to E_1, can be seen as a special case of this problem where the recovered isogeny is a power of ℓ.

2.2 The 1999 algorithm

In [13] Galbraith gave an algorithm that computes an isogeny between two ordinary elliptic curves. For two elliptic curves E_0, E_1, the approach used was to take a path starting at each of these curves that climbs up to the crater using modular polynomials. Once on the crater, a meet-in-the-middle computation determines how to join the two paths. If N is the largest prime divisor of the conductor and h is the class number of the maximal order then equation (5) of Section 6 of [13] states the complexity of the algorithm as

$$\begin{aligned} O(N^3(\log(N)+\log(q)) + \sqrt{h}(\log(h)^2 + \log(h)\log(q)^5) \\ + \log(h)\log(q)^6 + \log(q)^8/\log\log(q)) \end{aligned} \tag{1}$$

field operations. When N is constant or polynomial in $\log q$, the complexity is $\tilde{O}(q^{1/4})$. The worst case is when h is small and $N \approx q^{1/2}$, in which case the complexity is $\tilde{O}(q^{3/2})$ operations over $\mathbb{F}_q$.

2.3 The 2024 algorithm

Recently, Kani's lemma [18] was leveraged in some attacks [3, 21, 26] that were able to reconstruct a secret isogeny given some information about how the isogeny acts on a sufficiently large torsion group. Galbraith used this idea constructively in his recent paper [14] where he improves upon the worst case complexity of his previous work [13].

Instead of using modular polynomials to walk up the volcano from each of E_0, E_1 to the crater, Galbraith uses Kani's lemma to reconstruct the path. Suppose that N divides the conductor and that E_0 is directly above E_1 in the sense that there is a descending N-isogeny $\phi : E_0 \to E_1$ (see Figure 3). Then if we know ϕ on $E_0[M]$ for some suitable torsion group, the Kani construction can provide a representation of ϕ. Since we do not have any way to compute ϕ on $E_0[M]$, the method requires guessing how the isogeny acts on some torsion points. In other words, for several small primes $\ell \mid M$, given an ℓ-torsion basis (P_0, Q_0) on E_0, we need to guess $\phi(P_0), \phi(Q_0)$. It is explained in [14] that we need $M \approx N^{1/2}$.

To get the desired complexity we need the number of guesses for $\phi(P_0), \phi(Q_0)$ to be at most $O(M)$. Since there are M^2 choices for each M-torsion point, this seems to be a challenge. In [14] it is suggested to choose M to be a product of Elkies primes ℓ, namely primes such that $E_0[\ell]$ has a Frobenius eigenbasis with distinct eigenvalues, so $\pi_q(P_0) = uP_0$ and $\pi_q(Q_0) = qu^{-1}Q_0$ where π_q is the q-power Frobenius and $u \in (\mathbb{Z}/\ell\mathbb{Z})^*$. By choosing P_1 and Q_1 in the appropriate eigenspaces of $E_1[\ell]$, we know that $P_1 = [\lambda]\phi(P_0)$ and $Q_1 = [\nu]\phi(Q_0)$ for some integers λ, ν. Using the Weil pairing one can reduce to a single unknown integer,

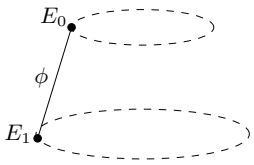


Fig. 3. Two curves at different levels of a volcano.

hence reducing the problem to trying $O(\ell)$ values. The drawback of this approach is that one needs to apply heuristic assumptions about the distribution of Elkies primes, see [14, Assumption 1] or [3, Section 10]).

For each guess, we run the Kani attack. If it returns a viable isogeny, then we can stop. Otherwise we repeat with a new guess. To run the Kani attack we have to choose our integer M so that $M - N$ is the sum of integer squares. One can write all positive integers as a sum of 4 squares, but the attack is faster if one can write $M - N$ as a sum of two squares.

2.4 Precise analysis in the case of small conductor

One of the observations in [14] is that it is sometimes more efficient to compute descending isogenies than ascending isogenies. Hence, [14] advocates descending to the floor, and solving the isogeny problem there using a meet-in-the-middle algorithm. This results in complexity $\tilde{O}(q^{1/4})$ operations for all conductors up to $q^{1/8}$. This is seen in Figure 1 with the flat blue line on the left hand side of the figure.

However, as already implied by equation (1), one can do better. We reproduce the analysis in this case.

First we recall the process for computing ascending and descending isogenies. In [13] ascending was done using modular polynomials and required cubic complexity. In [14] it is shown how to compute descending isogenies more efficiently (at least, asymptotically) when the group order is known by generating random kernel points. Given a curve over $\mathbb{F}_q$ with $q+1-t$ points and such that $\ell^4 \mid t^2-4q$, computing an ascending ℓ-isogeny may require $\tilde{O}(\ell^3)$ field operations, while computing a descending ℓ-isogeny takes $\tilde{O}(\ell^2)$ operations (see [14, Sect. 2.3] for details). It is possible, however, to compute the ascending isogeny in $\tilde{O}(\ell^2)$ operations as well. This will be detailed in Section 3.2. Now recall that we are trying to recover an isogeny, $\phi : E_0 \to E_1$ where $[\mathrm{End}(E_0) : \mathrm{End}(E_1)] = N$ and $\mathrm{End}(E_0) = \mathcal{O}_K$, i.e. E_0 lies on the crater. The approach from [13] is to take a walk up to the crater and then solve the problem there using meet-in-the-middle.

Suppose $N = q^b$ for a number $0 < b < 1/2$. As noted in [14, Section 2.3], walking up to the crater can always be done in $\tilde{O}(q^{2b})$ finite field operations (even with the $\tilde{O}(\ell^3)$ ascending algorithm from [14]). Call this ending curve E_2, so we have recovered the ascending isogeny $\varphi : E_1 \to E_2$ (see Figure 2.4 for reference).

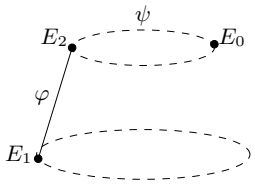


Fig. 4. Volcano where E_0 is not directly above E_1.

The class number of the crater is around $\sqrt{|\Delta_0|}$. Thus since $N^2\Delta_0 = t^2 - 4q$, we get that the class number will be $h_0 = \tilde{O}(\sqrt{q/N^2}) = \tilde{O}(q^{(1-2b)/2})$. Hence the meet-in-middle computation to recover the map $\psi : E_2 \to E_0$ on the crater will cost $\sqrt{h_0} = \tilde{O}(q^{(1-2b)/4})$.

The total complexity to recover $\hat{\varphi} \circ \hat{\psi} : E_0 \to E_1$ therefore comes from the cost to walk up to the crater and then conduct meet-in-the-middle which is $\tilde{O}(q^{2b}) + \tilde{O}(q^{(1-2b)/4})$ operations over $\mathbb{F}_q$. When $0 < b < 1/10$, we get that the meet-in-the-middle computation is dominating, giving a total complexity of $\tilde{O}(q^{(1-2b)/4})$. Otherwise, when $b \geq 1/10$, the total complexity will be $\tilde{O}(q^{2b})$ operations. The details are given in Algorithm 1.

Algorithm 1: Isogeny recovery in volcanoes with small conductor

Input : elliptic curves E_0, E_1 such that $\mathrm{End}(E_0) = \mathcal{O}_K$ and $[\mathrm{End}(E_0) : \mathrm{End}(E_1)] = N$
Output: $\phi : E_0 \to E_1$

1 Compute the unique ascending path $\varphi : E_1 \to E_2$ by sampling N-torsion points in E_1 and applying the formulas from Vélu ;
2 Compute an isogeny $\psi : E_2 \to E_1$ via meet-in-the-middle ;
3 Compute $\phi = \hat{\varphi} \circ \hat{\psi}$;
4 **return** ϕ ;

We gain an improvement over [13] when $0 < b < 1/8$. Suppose, for example, that $b = 1/14$. Then we get a complexity proportional to $q^{(1-2b)/4} \approx q^{0.21}$ operations over $\mathbb{F}_q$ which is better than $q^{0.25}$ operations. We outline this range of improvement in Figure 5.

3 Orientations

One can find a volcano structure in the supersingular isogeny graph over $\mathbb{F}_{p^2}$ when we endow the supersingular elliptic curves with an *orientation*. This terminology was introduced by Colo and Kohel [8].

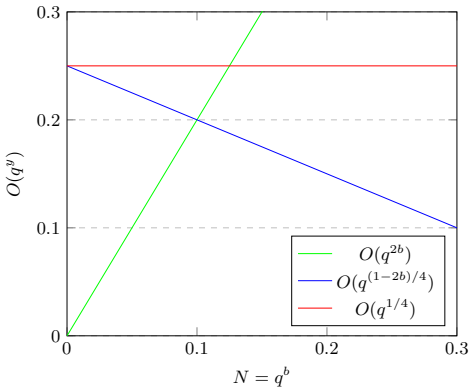


Fig. 5. The algorithmic complexity given in [14] is plotted in red. The complexity of the algorithm described in Section 2.4 is plotted in blue and green. Here we see that for $N < q^{1/8}$, the approach from Section 2.4 gives an improvement.

The terminology and notions in this section apply to both the ordinary and supersingular case. So let E be any elliptic curve over $\mathbb{F}_q$.

Definition 3.1 (K-orientation). *Let $K = \mathbb{Q}(\sqrt{d})$ be a quadratic imaginary field. We say an elliptic curve E is* K-oriented *if there exists a ring homomorphism*

$$\iota : K \hookrightarrow \mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}.$$

The pair (E, ι) is called a K-oriented elliptic curve.
If $R \subseteq K$ is a ring and $\iota(R) \subseteq \mathrm{End}(E)$ then we say E is R-oriented.

Suppose ι is the orientation on such a curve E. There exists a unique quadratic order $\mathcal{O} \subset K$ such that $\iota(\mathcal{O}) = \mathrm{End}(E) \cap \iota(K)$. Then we call ι a *primitive $\mathcal{O}$-orientation*, and we say that E is $\mathcal{O}$-orientable.

For any suborder $\mathcal{O}' \subset \mathcal{O}$, ι still induces an orientation on $\mathcal{O}'$, which is no longer primitive. Conversely, given an orientation ι on $\mathcal{O}'$, it extends uniquely to a K-orientation, and if $\mathcal{O}$ is the associated primitive orientation, we have $\mathcal{O}' \subset \mathcal{O}$. We call $\mathcal{O}$ the *saturation* of the orientation on E. We have $\Delta(\mathcal{O}') = \Delta(\mathcal{O})f^2$, and if m is an integer, we say that $\mathcal{O}'$ is *m-locally primitive* if m is coprime to f.

An oriented isogeny between two $\mathcal{O}$-oriented elliptic curves is an isogeny which commutes with the orientations. We will often drop the ι from our notation.

Example 3.1. The advantage of the above framework is that it applies just as well to ordinary curves.

If $E/\mathbb{F}_q$ is ordinary, there is a unique possible quadratic field K given by $K = \mathbb{Q}(\pi_q)$, and we will always use the natural orientation ι on K which sends π_q to the Frobenius endomorphism on E.

In particular, E is always oriented by the order $\mathbb{Z}[\pi_q]$, and the saturation of the orientation is simply given by $R = \mathrm{End}(E)$.

Finally, an oriented isogeny between ordinary elliptic curves with the natural Frobenius orientation is simply an $\mathbb{F}_q$-rational isogeny.

The class group $Cl(\mathcal{O})$ acts freely and transitively on the set of primitive $\mathcal{O}$-oriented curves up to isomorphisms (and Galois conjugacy in the special case where p is inert in $\mathcal{O}$, which can only happen in the supersingular case). A proof of this statement is given by Onuki in [23, Thm. 3.4]. This gives rise to a group action

$$\mathfrak{a} \star (E, \iota) = (E_{\mathfrak{a}}, \iota_{\mathfrak{a}})$$

where $\mathfrak{a}$ is an invertible $\mathcal{O}$-ideal coprime to the conductor of $\mathcal{O}$. To define it concretely, let $E[\mathfrak{a}] := \cap_{\alpha \in \mathfrak{a}} \ker \iota(\alpha)$, and denote by $\varphi_{\mathfrak{a}}^E$ the isogeny whose kernel is $E[\mathfrak{a}]$. Then

$$\varphi_{\mathfrak{a}}^E : E \to E_{\mathfrak{a}} = E/E[\mathfrak{a}] \text{ and } \iota_{\mathfrak{a}}(x) = \frac{1}{n(\mathfrak{a})} \varphi_{\mathfrak{a}}^E \circ \iota(x) \circ \hat{\varphi}_{\mathfrak{a}}^E.$$

If we also allow isogenies between $\mathcal{O}$-oriented curves, relaxing the primitive condition, then Colo and Kohel [8] show that there is a volcano structure just like the ordinary case: if E_1 is oriented by $\mathcal{O}$, and $\mathcal{O}_1$ is its saturation in E_1, and we have an order $\mathcal{O}_2$ containing $\mathcal{O}_1$ such that $\mathcal{O}_1$ is of conductor f in $\mathcal{O}_2$, then there is a unique "ascending" isogeny $\phi : E_1 \to E_2$ of degree f such that E_2 is primitively oriented by $\mathcal{O}_2$.

3.1 The module structure of the rational points of an oriented elliptic curve

In this section we let R be a quadratic imaginary ring, and we assume that we have an orientation $R \to \mathrm{End}(E)$ of an elliptic curve $E/\mathbb{F}_q$. Recall that the orientation is said to be primitive whenever R is saturated in $\mathrm{End}(E)$ (i.e., $R = (\mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}) \cap R$), or equivalently $\mathrm{End}(E)/R$ is torsion free. We call $E[n]$ a cyclic R-module if there is some $P \in E[n]$ such that $E[n] = RP = \{\phi(P) : \phi \in R\}$, in which case we call P and R-generator.

Theorem 3.1. *Let n be prime to the characteristic, and assume that the orientation R on E is primitive. Then $E[n]$ is a cyclic R-module.*

In particular, if $P \in E[n]$ is any R-generator, and $\gamma \in R$ is of discriminant $\Delta(\gamma) = f^2 \Delta_R$, then $\mathbb{Z}[\gamma]P$ is of cardinality $n^2/\gcd(n, f)$. So $E[n]$ is a cyclic $\mathbb{Z}[\gamma]$ module if and only if $\gcd(n, f) = 1$. It also follows that the Weil pairing $e_n(P, \gamma P)$ is a root of unity of exact order $n/\gcd(n, f)$.

Proof. The first statement is [20, Theorem 3]. For the second one, if $R = \mathbb{Z}[\omega]$, then we can write $\gamma = a + f\omega$. We have that $P, \omega P$ is a $\mathbb{Z}/n\mathbb{Z}$ basis of $E[n]$, and $\mathbb{Z}[\gamma]P$ is generated by P and $f\omega P$, and the latter is a point of order $n/\gcd(n, f)$. Finally $e_n(P, \gamma P) = e_n(P, \omega P)^f = \zeta^f$ is of order $n/\gcd(n, f)$ since $e_n(P, \omega P)$ is of exact order n. □

We can restate Theorem 3.1 as saying that $E[n]$ is R-cyclic if and only if R is n-locally primitive.

Corollary 3.1. *With the hypothesis of Theorem 3.1, $E[n]$ is isomorphic as an R-module to R/nR.*

Proof. By Theorem 3.1, $E[n]$ is a cyclic R-module, and if P is a generator, $E[n]$ is isomorphic to $R/\operatorname{ann}(P)$ where $\operatorname{ann}(P)$ is the annihilator of P. If $\gamma \in \operatorname{ann}(P)$, then since $\gamma(P) = 0$, R is commutative, and P generates $E[n]$ as an R-module, we find that $\gamma = 0$ on $E[n]$, so γ is divisible by $[n]$. The converse is obvious, so $\operatorname{ann}(P) = nR$, and $E[n] \simeq R/nR$. □

Corollary 3.2. *Let $n = \ell^e$ be prime to the characteristic, and assume that the orientation $R = \mathbb{Z}[\omega_R]$ on E is primitive. Let u be the degree of the field of definition of the geometric points of $E[\ell^e]$. Then $u \leq \ell^{e+1}$ in the ordinary case, and $u \leq \ell^e$ in the supersingular case.*

We can find an R-generator $P \in E[n]$, normalised such that $e_n(P, \omega_R P) = \zeta$, in time $\tilde{O}(\ell + u^2 \log^2 q + \log^{O(1)} q)$ (which can be improved to $\tilde{O}(\ell^{1/2} + u^2 \log^2 q + \log^{O(1)} q)$ in the supersingular case, or if ℓ is not inert in R).

For a general n, if R is n-locally primitive, we can find a basis (P_1, P_2) or an R-generator P of $E[n]$ in time $\tilde{O}(n^4 \log^2 q + \log^{O(1)} q)$ (resp. $\tilde{O}(n^2 \log^2 q + \log^{O(1)} q)$ in the supersingular case), by factorising n and applying the result above.

Proof. Assume for now that $E/\mathbb{F}_q$ is an ordinary curve (this is the harder case). We first explain how to find u. Recall that we can compute the discriminant of π_q acting on E in polynomial time in $\log q$ by point counting algorithms, and that we can use the endomorphism ring algorithm of [25] to also compute the ℓ-saturation R of $\mathbb{Z}[\pi_q]$ in E in polynomial time in $\log q$, so that R is ℓ-locally primitive in E. Now $E[\ell^e] \simeq R/\ell^e$ by Corollary 3.1, and u is the order of $\pi_q \in R/\ell^e$.

Finding u reduces to the computation of the order of elements in $\mathbb{F}_\ell$ if ℓ is ramified or split in R, and in $\mathbb{F}_{\ell^2}$ if ℓ is inert in R, hence costs $\tilde{O}(e \log \ell + \sqrt{\ell})$ or $\tilde{O}(e \log \ell + \ell)$ respectively. In the worst case $u \leq \ell^{e+1}$ (this can be improved to $u \leq \ell^e$ if ℓ is split in R).

Now $\pi_q^u - 1$ is divisible by ℓ^e in R. Write $\pi_q^u - 1 = \ell^e \gamma$, we have $N(\gamma) \leq q^u$. We can sample uniform points in $E[\ell^e]$ as follows. First we sample uniformly random points on $E(\mathbb{F}_{q^u})$ by computing square roots; this costs $\tilde{O}(u^2 \log^2 q)$ [24]. Then we apply γ to get uniformly random points on $E[\ell^e]$. This costs $\tilde{O}(u^2 \log^2 q + \log^{O(1)}(q))$. One then discards points that do not have the required order $n = \ell^e$. Finally, one takes pairs (P_1, P_2) and checks whether the Weil pairing $e_{\ell^e}(P_1, P_2)$ is of full order in μ_{ℓ^e}, in which case (P_1, P_2) is a basis. This requires generating $O(1)$ points and checking $O(1)$ conditions. Hence the overall cost is $\tilde{O}(\log(\ell^e) \log(q^u)) = \tilde{O}(eu \log \ell \log q)$ (because $\mu_{\ell^e} \subset \mathbb{F}_{q^u}$).

Once we have a basis, we know that either P_1 or P_2 is a generator of $E[\ell^e]$ as an R-module, and find it by evaluating ω_R followed by a Weil pairing. The total

cost, since $\ell^e < q^u$, is then $\tilde{O}(\ell + u^2 \log^2 q + \log^{O(1)} q)$ to find a basis (P_1, P_2) and an R-generator P of $E[\ell^e]$.

If $E/\mathbb{F}_{p^2}$ is a maximal supersingular curve, then $E(\mathbb{F}_{p^{2u}}) \simeq \mathbb{Z}/(p^u+1) \times \mathbb{Z}/(p^u+1)$, and so u is the order of p^2 modulo n. In particular, $u < n = \ell^e$.

We can also easily sample uniform points in $E[n]$ by sampling them in $E(\mathbb{F}_{q^u})$ and multiplying by the scalar cofactor (no need to use an endomorphism here). This costs $\tilde{O}(u^2 \log^2 q)$ operations. Then we use pairings to extract a basis (P_1, P_2), and then we apply ω_R and pairings again to find an R-generator R. The total cost is thus $\tilde{O}(\ell^{1/2} + u^2 \log^2 q)$ to find a basis (P_1, P_2) of $E[\ell^e]$, to which we need to add an endomorphism evaluation of $\tilde{O}(\log^{O(1)} q)$ to extract an R-generator P.

Finally we mention the case of general n. We simply handle each power of ℓ in turn and apply the Chinese remainder theorem to compute the point. The worst case is when $n = \ell$ is prime, in which case $u = O(n^2)$ and $u^2 = O(n^4)$, giving the stated complexity. □

Example 3.2. Let $E/\mathbb{F}_q$ be an ordinary curve. Specializing to the case $n = \ell$, we have three (well known) possibilities.

The first one is when ℓ splits in $\mathbb{Z}[\pi_q]$ (ℓ is Elkies), $\mathbb{Z}[\pi_q]/\ell\mathbb{Z}[\pi_q] \simeq \mathbb{F}_\ell^2$, π_q acts diagonally, and $u \mid \ell - 1$.

If ℓ is inert in $\mathbb{Z}[\pi_q]$ (ℓ is Atkin), $\mathbb{Z}[\pi_q]/\ell\mathbb{Z}[\pi_q] \simeq \mathbb{F}_{\ell^2}$, so $u \mid \ell^2 - 1$.

Finally, if ℓ is ramified in $\mathbb{Z}[\pi_q]$, then either π_q acts as a diagonal matrix $\begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}$ (so $u \mid \ell - 1$) on $E[\ell]$, or as $\begin{pmatrix} \lambda & 1 \\ 0 & \lambda \end{pmatrix}$ (so $u \mid \ell(\ell-1)$). But the diagonal matrix case can happen if and only if $\pi_q - \lambda$ is divisible by ℓ in $\mathrm{End}(E)$, which is equivalent to $\mathbb{Z}[\pi_q]$ being of conductor divisible by ℓ in R. And the second case happens precisely when $\mathbb{Z}[\pi_q]$ is ℓ-locally primitive. Similar analysis holds for $n = \ell^e$, see for instance [22] for some results.

Remark 3.1. As a consequence of the proof above, we remark that in the ordinary case, if $\phi : E_0 \to E_1$ is an ascending oriented isogeny, so that $R_0 \subset R_1$, and u_i denotes the degree of the field of definition of the points in $E_i[n]$, then $u_1 < u_0$, because u_i is the order of π_q in R_i/nR_i.

Definition 3.2. *Let R be a quadratic order of discriminant Δ_R. We define the canonical (up to sign) imaginary element ω_R as follows. If $\Delta_R \equiv 1 \mod 4$, we let $\omega_R = \sqrt{\Delta_R}$; in that case $\mathbb{Z}[\omega_R]$ is of conductor 2 in R. Otherwise, $\Delta_R \equiv 0 \mod 4$, and we let $\omega_R = \sqrt{\Delta_R}/2$; which means $R = \mathbb{Z}[\omega_R]$.*

Corollary 3.3. *Let E be an elliptic curve oriented by R, $n \mid \Delta_R$, and assume that the orientation is n-locally primitive. We let ω_R be as in Definition 3.2.*

Then $E[n, \omega_R]$ is cyclic, and there exists a basis $(P, \omega_R(P))$ of $E[n]$.

Furthermore, if either n is odd, or $n \mid \Delta_R/4$, the matrix of ω_R on this basis is given by

$$\begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

In particular, for this n, $E[n, \omega_R] = \omega_R(E[n])$.

Proof. If R is a primitive orientation on E, then $E[\omega_R]$ is cyclic because ω_R is not divisible by any integer in R by construction, so $E[\omega_R]$ cannot contain any subgroups $E[\ell]$. It follows that $E[n, \omega_R]$ is still cyclic if R is just n-locally primitive.

If n is even, then Δ_R has to be divisible by 4, so $R = \mathbb{Z}[\omega_R]$. If n is odd, $\mathbb{Z}[\omega_R]$ might be of index 2 in R, but it is still n-locally primitive.

We deduce that there exists a basis $(P, \omega_R(P))$ of $E[n]$. Since ω_R^2 is either Δ_R or $\Delta_R/4$, under our assumptions $\omega_R^2 = 0 \mod n$, so we get the form of the matrix. □

3.2 The kernel of an ascending isogeny

This section gives new results that allow to compute an ascending isogeny in certain situations. These results extend previous work by Kohel [19] and Ionica and Joux [16].

Kohel [19] pointed out that for ordinary curves E over $\mathbb{F}_q$, if one is on the floor with respect to a prime ℓ, then there is only one $\mathbb{F}_q$-rational ℓ-isogeny from E and it is automatically ascending. This is linked to the fact that the ℓ-torsion is $\mathbb{Z}[\pi_q]$-cyclic for such a curve. Since the cyclic subgroup corresponding to the kernel of the isogeny is fixed by Frobenius, the kernel of the isogeny is $E[\ell, \pi_q - \lambda]$ for some integer λ. In other words, the ascending isogeny has kernel $E[\ell, \pi_q - \lambda]$, and this is a special case of our main result.

Ionica and Joux [16] extended this further, by showing how pairings can, in some situations, identify which subgroup of $E[\ell]$ leads to an ascending isogeny. Their results apply to ordinary curves, and require certain self-pairings to be non-trivial, which does not always hold.

Our result in this section is much more general. It shows that, if we are given a basis of $E[\ell]$, defined over some extension $\mathbb{F}_{q^d}$, we can find the kernel of the ascending isogeny in polynomial time in $\log \ell$ and $d \log q$. It applies in all ordinary settings. It also applies to the oriented supersingular case (provided we have an effective representation of the orientation). It is important to stress that the notion of "ascending" isogeny is not well-defined for supersingular curves in general. Instead, one must consider supersingular curves with an orientation, and it is the orientation that specifies whether or not an isogeny is ascending. This is exactly the information provided by the SCALLOP system.

Theorem 3.2. *Let $\phi : E_0 \to E_1$ be an ascending R-oriented cyclic isogeny of degree d. Let R_i be the saturation of R on E_i, let f be the conductor of R_0 in R_1. Then $f \mid d$, and ϕ factorizes uniquely as $\phi = \phi_2 \circ \phi_1$ where ϕ_1 is a purely ascending isogeny of degree f and ϕ_2 is an R_1-horizontal oriented isogeny.*

Furthermore, for any ω_1 such that $R_1 = \mathbb{Z}[\omega_1]$ (i.e. such that $\Delta_{\omega_1} = \Delta_{R_1}$), $R_0 = \mathbb{Z}[f\omega_1]$ and the kernel of ϕ_1 is given by $E_0[f\omega_1, f]$.

Proof. The unicity of the decomposition comes from the volcano structure.

The form of the kernel is an easy consequence of the general equivalence of categories described in [17, 27], which implies that the kernel of the ascending

isogeny is given by the action of the conductor ideal. We give a direct proof in a special case that is sufficient for our applications.

Let $\phi_1 : E_0 \to E_0'$ be our strictly ascending f-isogeny. Since ω_1 is an endomorphism on E_0', we have that $f\omega_1$ is 0 on $E_0'[f]$. Since $f\omega_1 \in R_0$ and ϕ_1 is R_0-oriented, if $P \in E_0$ is such that $\phi_1(P) \in E_0'[f]$, then $\phi(f\omega_1 P) = f\omega_1\phi(P) = 0$. In particular, $\ker\phi_1$ contains $(f\omega_1)E_0[f]$.

We also remark that $E_0[f, f\omega_1]$ does not depend on the choice of generator ω_1 for R_1: another generator will be of the form $\omega_1' = a \pm \omega_1$ with $a \in \mathbb{Z}$, so $E_0[f, f\omega_1] = E_0[f, f\omega_1']$. In fact, the argument shows that we can even take $\omega_1' = a + b\omega_1$, of discriminant $\Delta(\omega_1') = b^2\Delta_R$, as long as b is coprime to f.

So we can take ω_1 to be the canonical imaginary element of R_1 from Definition 3.2. Now for simplicity, we assume that f is odd or that Δ_0 is even. In that case, $f\omega_1$ is the canonical imaginary element of R_0, and by Corollary 3.3 there is a basis $(P, f\omega_1 P)$ of $E_0[f]$ and $(f\omega_1)^2$ is zero on $E_0[f]$. It follows that $(f\omega_1)E_0[f] = E_0[f, f\omega_1]$, and so $\ker\phi_1$ contains $E_0[f, f\omega_1]$. But ϕ_1 is of degree f, and the above basis shows that $E_0[f, f\omega_1]$ is too. □

Looking at the proof of the theorem above, we can give another description of a generator of the kernel of ϕ_1, more useful for algorithmic applications:

Corollary 3.4. *With the notations above, assume that R is f-locally primitive in E_0, and let ω_R be the canonical imaginary element from Definition 3.2.*

If either f is odd, or Δ_1 is even, then the kernel K of ϕ_1 is given by $E_0[\omega_R, f]$. If P is any generator of $E_0[f]$ as an R-module, then $\omega_R P$ is a generator of this kernel K.

We can compute the ascending isogeny $\phi_1 : E_0 \to E_0'$ in $\tilde{O}(u^2 \log f \log^2 q + uB_f^{1/2} \log f \log q + \log f \log^{O(1)} q) = \tilde{O}(f^2 \log^2 q + \log f \log^{O(1)} q)$ operations, where B_f is a bound on the largest prime divisor of f, and $u < f$ a bound on the degrees of the field extensions where the points of $(\ker\phi)[\ell_i]$ live, for $f = \prod \ell_i^{e_i}$.

Proof. If $\omega_0, \omega_1, \omega_R$ are the canonical imaginary elements of R_0, R_1, R respectively, then the proof of Theorem 3.2 shows that under our assumptions, we have $\omega_0 = f\omega_1$ and $\omega_R = u\omega_0$ for u coprime to f. From the R module structure of $E_0[f]$, we see that the kernel of ϕ_1, given by $E_0[f, \omega_R]$, is generated by $\omega_R P$ for any R-generator P of $E_0[f]$.

Now, to compute ϕ_1, we will work prime divisor by prime divisor of f (recall that we assumed that we know the factorisation of f_R, hence of f); there are at most $O(\log f)$ such primes. So in what follows we can assume that f is prime.

As in Corollary 3.2, in the ordinary case, we can determine the field extension where the points of the kernel of ϕ_1 are defined by determining the smallest u such that $\pi_q^u = 1$ in $R_0/(f\omega_1, f)$. Since the kernel is of order f, we have $u \mid f-1$. We also remark at this point that, as in Remark 3.1, since the saturation of R in E_0' will contain R_0, the degrees we will subsequently obtain on E_0' when we work prime by prime will be smaller than on E_0.

Now if $E_0[f](\mathbb{F}_{q^u})$ is cyclic, it is automatically equal to $\ker\phi_1$ since the kernel is rational. Sampling a point in $E_0(\mathbb{F}_{q^u})$ and multiplying by the (scalar) cofactor now gives a uniform point in the kernel.

The second case is that $E_0[f](\mathbb{F}_{q^u})$ is the full f-torsion. We can apply Corollary 3.2 to sample a generator P of $E_0[f]$ as an R-module. Then, in the situation of Corollary 3.4, we can apply ω_R to P to get a generator of $\ker \phi_1$.

Finally, computing ϕ_1 from a generator of the kernel can be done in $\tilde{O}(f^{1/2} u \log q)$ operations as shown in [2]. All in all, we can compute ϕ_1 in $\tilde{O}(u^2 \log f \log^2 q + uf^{1/2} \log q + \log^{O(1)} q)$ operations.

The supersingular case works as above, except that this time u is automatically also the degree of the field of definition of the full f-torsion. □

Remark 3.2. The only case not covered by Corollary 3.4 is when Δ_1 is odd and $f = 2f'$ is even. Then we can write ϕ_1 as an f_0-ascending isogeny $E_0 \to E_0''$ with R' the saturation of R in E_0'', of discriminant $4\Delta_1$, followed by a 2-isogeny $E_0'' \to E_0'$. We can apply Corollary 3.4 to the isogeny $E_0 \to E_0''$. And by Theorem 3.2, since $R' = \mathbb{Z}[\frac{1+\sqrt{\Delta_1}}{2}]$, the kernel of the 2-isogeny $E_0'' \to E_0'$ is given by $E_0''[1 + \sqrt{\Delta_1}, 2]$.

4 Recovering an oriented isogeny of known degree

In this section we study the following problem: let R be a quadratic imaginary order. Let $\phi : E_0 \to E_1$ be an R-oriented cyclic isogeny of known degree d between elliptic curves defined over $\mathbb{F}_q$. Our goal is, given R, E_0, E_1 and d, to recover ϕ. Following [14] we are going to do this by trying to determine ϕ on $E[n]$ for suitable n. In this paper the key tools are self-pairings.

We recall from Section 3 that this encompasses two cases. If $E_0/\mathbb{F}_q$ is ordinary, we can take $R = \mathbb{Z}[\pi_q]$ and we will always use the natural Frobenius orientation on E_0, E_1. This is the case we will consider for the (ordinary) computational isogeny problem. We remark that point counting algorithms give Δ_R in polynomial time in $\log q$. The other case is when E_0, E_1 are maximal supersingular elliptic curves defined over $\mathbb{F}_{p^2}$. In that case, ϕ is also automatically rational over $\mathbb{F}_{p^2}$. This is the case we will consider for the SCALLOP group action.

Let R_i be the saturation of R in $\mathrm{End}(E_i)$, and Δ_i the discriminant of R_i. We assume that ϕ is "ascending", meaning that $R_0 \subset R_1$, and we let $f = [R_1 : R_0]$ be the conductor of R_0 in R_1. We will say that ϕ is purely ascending if $f = d$. We remark that ϕ is automatically R_0-oriented: if $\gamma \in R_0$, there is some multiple $m\gamma$ in R, so since ϕ is R-oriented we have $[m]\gamma \circ \phi = \phi \circ [m]\gamma = [m]\phi \circ \gamma$. Dividing by $[m]$, we get $\gamma \circ \phi = \phi \circ \gamma$.

We will make the following assumptions for our complexity analysis:

- First, for simplicity, we will assume that $\log \Delta_R$ is polynomial in $\log q$. This is automatic in the ordinary case, since $|\Delta_{\pi_q}|$ is at most $4q$; and is also the case in all currently proposed instances of a supersingular oriented group action. All statements in this section are still valid in the general case if we replace the $O(\log q^{O(1)})$ in the complexity statements by $O((\log q + \log \Delta_R)^{O(1)})$.

- The orientation R is effective on both E_0 and E_1, meaning that we can evaluate any endomorphism $\gamma \in R$ on a point P using a polynomial in $\log N(\gamma)$ arithmetic operations over the field of definition of P.
  We can refine this as follows: we fix $\omega_R = \sqrt{\Delta_R}/2$ if $\Delta_R \equiv 0 \mod 4$, and $\omega_R = (1+\sqrt{\Delta_R})/2$ if $\Delta_R \equiv 1 \mod 4$. In both cases $N(\omega_R) \leq |\Delta_R|$, and if $\gamma = a + b\omega_R$, $a, b \leq \sqrt{2N(\gamma)}$. If we have an efficient representation of ω_R, then evaluating γ on P requires the evaluation of ω_R, scalar multiplication by $[a], [b]$ and a sum. This thus costs $O(\log(|a|) + \log(|b|) + \log^{O(1)} \Delta_R) = O(\log(N(\gamma)) + \log^{O(1)} \Delta_R)$ arithmetic operations over the field of definition of P.
  As an aside, by adapting point counting algorithms, given an efficient representation of ω_R we can recover Δ_R in polynomial time. From now on, for our complexity analysis, whenever we pick up a generator ω_R of R, we will always assume that ω_R is a generator like the above, such that $N(\omega_R) = O(\Delta_R)$. This prevents silly situations like using $a + \omega_R$ for a very large a.
- As a consequence, the orientations by R_0 and R_1 are also effective. If γ_0 is an element of R_0, then some multiple n_0 of γ_0 is in R, hence can be efficiently evaluated (since $n_0 \mid \Delta_R$ cannot be too large). Now to evaluate γ_0 on a point P, we would like to find a point P' such that $P = [n_0]P'$; then $\gamma_0 P = (n_0\gamma_0)P'$. Such a point can be very expensive to compute for large, non-smooth n_0 since we would require a large extension field. Instead, Robert shows in [25, Sect. 2.3] how to efficiently divide an endomorphism by using higher dimensional isogenies.
  If γ_0 is any generator of R_0, we can thus find an efficient representation of γ_0, and use this efficient representation to evaluate the other endomorphisms. Finding this efficient representation involves the evaluation of an endomorphism of R, but this is a precomputation that only needs to be done once.
  Similar techniques also allow us to push forward the R-representation through an explicit R-isogeny $\phi' : E_0 \to E_0'$, by computing the image by ϕ' of $(P, \omega_R P)$ for several points P. This allows to find an efficient representation of the R-orientation on E_0' through a polynomial in $\log \Delta_R$ number of calls to the evaluation of ϕ'.
- Finally, we will assume that the factorisation of Δ_R is known. As a consequence, we can write $\Delta_R = \Delta f_R^2$ where Δ is a fundamental discriminant and the factorisation of f_R is known. We also have $\Delta_0 = \Delta f_0^2$ and $\Delta_1 = \Delta f_1^2$, with $f_0 = f f_1$. Then thanks to [25], we can recover f_0 and f_1, hence R_0 and R_1 in polynomial time in $\log \Delta_R + \log q$.

4.1 The general strategy

We will adapt the strategy of [14], which consists in guessing the image of ϕ on suitable subgroups of E_0, typically the image on the ℓ-torsion $E_0[\ell]$ for several small primes ℓ, and then reconstructing ϕ using a higher dimensional isogeny.

There are a number of different ways to reduce the number of guesses required to determine ϕ on $E_0[\ell]$ and we give a high level overview of them now.

- *The case of Elkies primes.* We can adapt the strategy of Section 2.3 to the general oriented case as follows. If $\ell \nmid \Delta_R$ splits as $\ell = \mathfrak{l}_1\mathfrak{l}_2$, then $E_0[\mathfrak{l}_i]$ is cyclic of order ℓ with generator P_i. Likewise, $E_1[\mathfrak{l}_i]$ is cyclic of order ℓ with generator Q_i. Since ϕ is R-oriented, $\phi(E_0[\mathfrak{l}_i]) \subset E_1[\mathfrak{l}_i]$, so we have $\phi(P_i) = a_iQ_i$, for some unknown scalars a_1, a_2 modulo ℓ. The Weil pairing gives us some information on a_1, a_2: we have $e_\ell(\phi(P_1), \phi(P_2)) = e_\ell(P_1, P_2)^d = e_\ell(Q_1, Q_2)^{a_1a_2}$. So, provided discrete logarithms are easy in the multiplicative group μ_ℓ of ℓ-th roots of unity in $\overline{\mathbb{F}}_q$, we can compute c such that $e_\ell(Q_1, Q_2) = e_\ell(P_1, P_2)^c$, and we know that $a_1a_2 = d/c$ modulo ℓ.
  In the special case that $\ell \mid d$, then either a_1 or a_2 is 0, and $E_0[\mathfrak{l}_1]$ or $E_0[\mathfrak{l}_2]$ is in the kernel of ϕ. So we recover part of the kernel of ϕ up to a choice. Otherwise, $\ell \nmid d$, and there are $\ell - 1$ possibilities for a_1, and then a_2 is completely determined by a_1.
- *Using self-pairings.* If $\ell \mid \Delta_R$, and R is a ℓ-locally primitive orientation on E_0, E_1, then we follow the insight of [5] that there exists a self-pairing that gives the image of ϕ up to a sign on a cyclic subgroup of $E_0[\ell]$. In [20, § 8], Macula and Stange give a more efficient construction of this self-pairing (see in particular [20, Example 3]), and we use a slight variant of their approach.
- *Using sesquilinear self-pairings.* If $\ell \nmid \Delta_R$, then $E[\ell]$ is a cyclic R-module, and Macula and Stange show in [20, Theorem 6] that there exists an R-sesquilinear self-pairing. While this gives less information than the previous case of self pairing, this still allows them to reduce the number of choices for the action of ϕ on $E_0[\ell]$ to $\ell - 1$ (if ℓ splits) or $\ell + 1$ (if ℓ is inert), see [20, Theorem 8 and 9]. In particular, this subsumes the previous special case of Elkies primes.
- The remaining case, at least if $R = R_0$ which we can assume from our hypotheses, is when $\ell \mid f$. In that case, R is not an ℓ-locally primitive orientation on E_1, and we cannot follow the pairing approach. Since we are speaking at the moment about small ℓ, this case is treated in [13, 14] as an easy case, because one can ascend in the volcano efficiently using the methods in those papers. We have shown in Section 3.2 an improved method to compute $(\ker \phi)[\ell]$ directly.

In summary, we will show that the self-pairings approaches of [5, 20], which were mainly used in the horizontal case, work just as well for the ascending case. The main difference compared to [20, Theorem 9] (which treats the case $\ell \nmid \Delta_R$) and [20, Theorem 11] (which treats the case $\ell \mid \Delta_R$) is that we also treat the case when ℓ divides d. This gives more flexibility in our choice of ℓ.

We also give in Section 4.2 a simplified and unified construction of "self-pairings", which only requires the standard Weil pairing (but is heavily inspired by the sesquilinear pairings used in [20]). By contrast, in [20] two different pairing constructions are given, depending on whether $\ell \mid \Delta_R$ or not. Our approach is similar to the use of the distortion map in pairing based cryptography. We refer to the end of Section 4.2 for a more detailed comparison of our approaches with the approach of [5, 20]. We also remark that the use of the distortion map was also explored in [12] as a way to determine whether an ℓ-isogeny was descending.

Finally, since we will apply self-pairings to large-ish ℓ, we will present some precise complexity statements that will be needed for the analysis in Section 5 and Section 6.

4.2 Pairings and distortion maps

We can reduce the computation of ϕ to the computation of the purely ascending isogeny ϕ_1 and an horizontal isogeny ϕ_2. We might not want to compute the purely ascending isogeny ϕ_1 fully, especially if the conductor f has large prime factors.

Section 3.2 can be seen as recovering partial information on the action of ϕ on $E[n]$ when $n \mid f$. But in this section, we focus on the case n coprime to f. More precisely, we explain how we can combine pairings with the orientation R to recover partial information about the image of ϕ on a basis of $E[n]$.

Recall that a pairing is a bilinear and non-degenerate map. Let E be an elliptic curve over $\mathbb{F}_q$ and m an integer co-prime to q. The most well-known and familiar pairing in elliptic curve cryptography is the Weil pairing

$$e_m : E[m] \times E[m] \to \mu_m \subseteq \bar{\mathbb{F}}_q^*$$

where μ_m is the multiplicative group of order m.

An important property of pairings, which has been widely used in isogeny cryptography and cryptanalysis, is that if $\phi : E_0 \to E_1$ is an isogeny of elliptic curves of degree d and $P, Q \in E_0[m]$ then

$$e_m(\phi(P), \phi(Q)) = e_m(P, Q)^d$$

where the first pairing is being computed on E_1 and the second pairing on E_0. Sometimes this property is used to learn information about the degree of an unknown isogeny. Other times, and this is how we use it, knowing d, one can use the above property to constrain the possible values for $\phi(P)$. These approaches have been used in a number of papers, including [5, 14, 20].

As mentioned in the introduction, an important set of techniques about self-pairings was given in [5], however in some contexts (including in our application) this requires a large field extension to be computed. Macula and Stange [20] showed a different approach that enables to obtain the same results without enlarging the fields over which one is working. Of particular importance for us will be [20, Theorem 8], which treats the case when n is coprime to Δ_R (this is a hidden hypothesis which comes from the invocation of Theorem 6 and Theorem 11 of [20] which treats the case when $n \mid \Delta_R$). Since [20] treats sesquilinear pairings and other advanced topics, the proof of this theorem may not be easily accessible to some researchers. Hence we take the opportunity to give an elementary proof of the result in a special case, and also note that the condition of supersingularity is not required.

Here is our version of "self-pairings".

Theorem 4.1. *Let $\phi : E_0 \to E_1$ be an R-oriented cyclic isogeny of degree d, let $n = \ell^e$ be an integer coprime to d, and assume that R is n-locally primitive on E_0 and E_1. Assume that we are also given $P_0 \in E_0[n], P_1 \in E_1[n]$ cyclic R-generators, and write $\phi(P_0) = \gamma(P_1)$ for some unknown endomorphism $\gamma \in R/nR$.*

Then we can compute $v = \deg(\gamma) \pmod n$ in time $\tilde{O}(ue \log \ell \log q + u' e\sqrt{\ell} \log q)$, where u is the degree of the field of definition of $E_0[\ell^e]$, and $u' \mid u$ the degree of the field of definition of μ_{ℓ^e}.

Proof. Take ω_R a generator of R, of norm bounded by $|\Delta_R|$ so that we can evaluate it efficiently. Since ϕ is oriented, it commutes with ω_R. Furthermore, since $RP_0 = E_0[n]$ it follows that $\{P_0, \omega_R(P_0)\}$ generates $E_0[n]$ and so $e_n(P_0, \omega_R(P_0))$ is an exact n-th root of unity. Similarly, $e_n(P_1, \omega_R(P_1))$ is an exact n-th root of unity. (The endomorphism ω_R here is performing the role of a "distortion map".) Suppose $\phi(P_0) = \gamma(P_1)$.

By compatibility of the Weil pairing with isogenies, we have

$$\begin{aligned} e_n(P_1, \omega_R(P_1))^{\deg(\gamma)} &= e_n(\gamma(P_1), \gamma\omega_R(P_1)) \\ &= e_n(\phi(P_0), \omega_R(\phi(P_0))) \\ &= e_n(P_0, \omega_R(P_0))^{\deg(\phi)}. \end{aligned}$$

Thus by computing these two pairings in $\tilde{O}(ue \log \ell \log q)$ and solving the discrete logarithm problem in $\tilde{O}(u' e\sqrt{\ell} \log q)$, one can compute $\deg(\gamma)$ modulo the order of $e_n(P_0, \omega_R(P_0))$, which is n. □

By combining Corollary 3.2 and Theorem 4.1, we see that sampling P_0, P_1 dominates the complexity:

Corollary 4.1. *With the notations above, we can sample cyclic R-generators $P_0 \in E_0[n], P_1 \in E_1[n]$, and compute $\deg \gamma \pmod n$ where $\gamma(P_1) = \phi(P_0)$ in $\tilde{O}(\ell + u^2 \log^2 q + \log^{O(1)} q)$.*

Let $\gamma = a + b\omega_R \in R \subseteq \mathrm{End}(E_1)$ be such that $\phi(P_0) = \gamma(P_1)$. Then $\deg(\gamma) = N(\gamma) = a^2 + ab\,\mathrm{Tr}(\omega_R) + b^2 N(\omega_R)$.

From now on, we will suppose that n is coprime to d, since the non-coprime case is easier and will be treated in Section 4.3, so that $v = \deg(\gamma) \neq 0 \pmod n$.

For our applications it will not be enough to know $\deg(\gamma)$. We will need to know the integers a and b, because we want to actually compute $\phi(P_0)$ by computing $\gamma(P_1)$. (From this we will also be able to compute $\phi(Q_0)$ where $\{P_0, Q_0\}$ is a basis for $E_0[n]$, by taking $Q_0 = \omega_R(P_0)$ and $\phi(Q_0) = \omega_R(\phi(P_0))$.) To find these integers we need to solve the conic equation

$$x^2 + xy\,\mathrm{Tr}(\omega_R) + y^2 N(\omega_R) = v \pmod n. \tag{2}$$

In practice it is usually simpler to separately solve the conic modulo each prime dividing n, and then use Hensel lifting and the Chinese remainder theorem to compute the set of solutions. We remark that this conic is of discriminant $\Delta(\omega_R) = \Delta_R$.

There are two cases of relevance in our paper. The first is when n is coprime to Δ_R and the conic is non-singular. In this case there are $O(n)$ solutions. To find solutions one calculates a rational parameterization of the conic and hence one can easily enumerate all solutions. The second case is when $n \mid \Delta_R$ and the conic degenerates, typically as the union of lines. In this case, as long as n does not have too many distinct prime factors, we can obtain a lot of useful information. We explain in more detail below.

The case of a non-degenerate conic We suppose here for simplicity that $n = \ell$ is a prime, and that $\ell \nmid \Delta_R$.

By assumption, we know there exists an endomorphism γ, hence a solution of Eq. (2) for $x, y \in \mathbb{Z}/\ell\mathbb{Z}$. Thus, the homogenised conic $X^2 + XY\,\mathrm{Tr}(\omega_R) + Y^2 N(\omega_R) - vZ^2$ is isomorphic to $\mathbb{P}^1$ over $\mathbb{Z}/\ell\mathbb{Z}$.

At infinity, $Z = 0$, we have 2 or 0 rational solutions $(X : Y : 0)$ depending on whether Δ_R is a square or not modulo ℓ. We deduce that if ℓ splits in R, the conic has $\ell - 1$ solutions (x, y), and if ℓ is inert in R, the conic has $\ell + 1$ solutions (x, y).

The case of a degenerate conic In this subsection, we assume that $n \mid \Delta_R$.

Theorem 4.2. *Let ω_R be the canonical imaginary element of R as defined in Definition 3.2. Let $n \mid \Delta_R$ be coprime to d, and such that $n \mid \Delta_R/4$ if n is even. Let $P_0 \in E_0[n]$ be a cyclic generator, and same for P_1. Let $Q_0 = \omega_R(P_0)$ and $Q_1 = \omega_R(P_1)$. Then $\phi(Q_0) = \alpha Q_1$, where $\alpha^2 = v \mod n$, where v is computed using pairings as in Theorem 4.1.*

Proof. As in Corollary 3.3, if n is even then $R = \mathbb{Z}[\omega_R]$, otherwise n is odd and $\mathbb{Z}[\omega_R]$ may be of index 2. In both cases, $\mathbb{Z}[\omega_R]/n\mathbb{Z}[\omega_R] = R/nR$, so we may assume that $\gamma = x + y\omega_R$ is such that $\phi(P_0) = \gamma(P_1)$. Now since ω_R is imaginary, the norm is $N(\gamma) = x^2 + \Delta_{\omega_R} y^2$. And by hypothesis, $n \mid \Delta_{\omega_R}$. So our conic degenerates to $x^2 = v \mod n$. The solutions are given by (α, y) for any α such that $\alpha^2 = v$.

We now recall that $\omega_R^2 = 0$ modulo n by Corollary 3.3. So if $\gamma = \alpha + y\omega_R$, then $\phi Q_0 = \phi\omega_R P_0 = \omega_R \phi P_0 = \omega_R(\gamma P_1) = \omega_R(\alpha P_1 + y\omega_R P_1) = \alpha\omega_R P_1 = \alpha Q_1$. In particular, the image of ϕ on Q_0 only depends on the possible solutions α. □

We remark that the equation $\alpha^2 = v$ has at most 2^{m+1} solutions, where m is the number of distinct prime factors of n. More precisely, since there exists at least one solution by hypothesis, there are exactly 2^{m-1}, 2^m or 2^{m+1} solutions according to whether $v_2(n) = 1$, $v_2(n) = 0, 2$, or $v_2(n) > 2$ respectively.

This means that while the image of Q_0 is fairly constrained (if n has not too many prime factors), the conic itself actually has $\approx 2^m n$ solutions, rather than just $O(n)$ (because it is not irreducible). We will call this version of self-pairing the cyclic version, because it gives information on a cyclic subgroup of $E_0[n]$.

Comparison with self-pairings As mentioned in [20], the sesquilinear pairings used in that article can be seen as a neat way to package together the pairing data $e_n(P,Q), e_n(\omega_R P, Q), e_n(P, \omega_R Q), e_n(\omega_R P, \omega_R Q)$ together into an R-sesquilinear Weil pairing. (In [20] Macula and Stange look at the R-sesquilinear Tate pairing rather than the sesquilinear Weil pairing, but the sesquilinear Weil pairing works similarly and was constructed in [29].)

Unfortunately, this sesqulinear pairing becomes degenerate when $n \mid \Delta_R$, so to handle this situation, in [20] the authors look at $(P,Q) \mapsto (t_n(\omega_R P, Q), t_n(P,Q))$ where t_n is the Tate pairing. Theorem 4.1 is almost the same, except that using the Weil pairing e_n instead of the Tate pairing allows us to only use $e_n(P, \omega_R Q)$, because the Weil pairing is alternating, and to use the same construction both for $n \mid \Delta_R$ and n coprime to Δ_R. We remark that the same phenomena was present in [6], where the authors remarked that it is often more convenient to use the Weil pairing than the Tate pairing to evaluate the "character" associated to an horizontal isogeny. This is not surprising: as explained in [5, § 6.2], this character evaluation can be done through self-pairings computation.

Finally, still for the case $n \mid \Delta_R$, the authors of [5] construct a self-pairing on the cyclic subgroup $E_0[n, \omega_R]$, using an "oriented" version of the Tate pairing [5, § 5.1]. By the discussion of [5, § 5.3], we see that a way to compute these self-pairings is via the distorted pairing $e_n(P, \omega_R Q)$ from Theorem 4.1. We refer to Appendix A for a generalisation of this result.

4.3 Putting everything together

In this section, we summarise the torsion information we can guess on ϕ, and how to use it to reconstruct ϕ. Recall that we assume $d = \deg(\phi)$ is known and is large, and we are choosing suitable $n = \ell^e$, where ℓ is a small prime, and attempting to deduce candidates for ϕ on $E_0[n]$.

We have developed tools for a number of cases.

1. The case $n = \ell$ is a prime dividing d. In this article, we will be interested in the case where ℓ furthermore divides the conductor f. Then we can use Section 3.2 to compute the associated ascending ℓ-isogeny $\phi_1 : E_0 \to E_0'$. We have $\phi = \phi_2 \circ \phi_1$, and we are reduced to recovering $\phi_2 : E_0' \to E_1$ of degree d/ℓ. We remark that the case where n divides d but not f is standard: we can write as above $\phi = \phi_2 \circ \phi_1$ where this time ϕ_1 is an horizontal ℓ-isogeny, and there are at most two such isogenies.
2. The case n is coprime to Δ_R and d. Then by Section 4.2, we have $O(n)$ possible choices for the action of ϕ on $E_0[n]$, which we can recover in $\tilde{O}(\ell + u^2 \log^2 q + eu \log \ell \log q + \log^{O(1)} q) = \tilde{O}(n^4 \log^2 q + \log^{O(1)} q)$. Here, u is the degree of the field extension of the points in $E_0[n]$. Indeed, this dominates both the cost of finding the conic equation Eq. (2), and the cost of finding all solutions of this conic.
3. The case $n \mid \Delta_R$, but with n coprime to d (hence also to f). For simplicity, we will assume that $n \mid \Delta_R/4$ if n is even. Then by Section 4.2, we have $O(2^m)$ possible choices for the action of ϕ on a specific point Q of $E_0[n]$

(with the notation of Theorem 4.2, the image by ω_R of an R-generator of $E_0[n]$), where m is the number of distinct prime divisors of n. These choices can be computed in $\tilde{O}(\ell + u^2 \log^2 q + eu \log \ell \log q + \log^{O(1)} q) = \tilde{O}(n^4 \log^2 q + \log^{O(1)} q)$.

We now explain how to use the torsion information from cases 2 and 3, which are the main cases of interest since d is coprime to n for these cases and so we are not directly getting information about the kernel of ϕ.

The method in [14] was based on the Kani attack [3, 21, 26], which needed the full torsion information (as provided by case 2). Namely, knowing the torsion information of ϕ on the n-torsion for $n^2 > 4d$ is enough to recover ϕ. This involves computing an n^2-isogeny $\Phi : E_0^r \times E_1^r \to E_0^r \times E_1^r$ in higher dimension $g = 2r$. The higher dimensional isogeny Φ is built out of an $(d, n^2 - d)$-isogeny diamond, involving the isogeny $\phi \,\mathrm{Id}_r$ with $r = 1, 2, 4$ and auxiliary isogenies ϕ', ρ, ρ' in dimension r of polarised degree $d, n^2 - d, n^2 - d$ respectively. Then Φ will be an isogeny of polarised degree n^2 in dimension $2r$.

The reason we need to move from dimension 1 to r is to be able to compute ψ, ψ' of the correct degree efficiently. Namely, if $n^2 - d = \sum_{i=1}^r a_i^2$ is the sum of $r = 1$, 2 or 4 squares, we can build the auxiliary isogenies in dimension r by using suitable $r \times r$ integer matrices. We know by Lagrange's theorem that we can always find a sum of 4 squares (and finding an explicit decomposition is efficient), so we can always work in dimension $g = 8$.

The Kani construction is well-documented in many papers so we do not give all the details. We remind the reader that $\ker \Phi$ may be computed as

$$\{(\hat{\phi}(P), -\rho'(P)) : P \in E_1^r[n^2]\}$$

where we view $\hat{\phi} : E_1^r \to E_0^r$ as a diagonal map, and where $\rho' : E_1^r \to E_1^r$ is coming from the matrix associated with the sum of squares. To compute $(\ker \Phi)[\ell]$ it suffices to know a basis (P, Q) for $E_0[\ell]$ and the values $(\phi(P), \phi(Q)) \in E_1[\ell]$ and everything else follows.

Finally, we mention that in practice, Φ is not computed directly, rather we write it as $\Phi = \Phi_2 \circ \Phi_1$ where Φ_i is of polarised degree n and we compute the two isogenies Φ_1 and $\tilde{\Phi}_2$ from $E_0^r \times E_1^r$ and $E_1^r \times E_0^r$ respectively and meet in the middle. It suffices to know ϕ on $E_0[n]$ to be able to compute the kernel of both these higher dimensional isogenies.

Remark 4.1. To handle case 3 we use an approach that was presented in an invited talk at ANTS in 2024 by Castryck [4]. At a high level, the result is that one can recover an unknown isogeny of degree d given interpolation data coming from a group of size $O(d)$.

We briefly recall the ideas from [4], which apply in a simplified form for our case 3. Reusing the notation and arguments from Theorem 4.2, we have points P_0, Q_0, P_1, Q_1 such that $\phi(P_0) = \gamma P_1$, for some $\gamma = \alpha + y\omega_R$, where α is constrained by an equation $\alpha^2 = v \pmod n$, but y can be arbitrary. From this we know that $\alpha Q_1 = \alpha \omega_R(P_1) = \phi(Q_0)$, but $\phi(P_0) = \alpha P_1 + yQ_1$. The idea is to

compose ϕ with an isogeny $\psi : E_1 \to E_2$ of kernel $\langle Q_1 \rangle$, so that

$$\psi(\phi(P_0)) = \psi(\alpha P_1 + yQ_1) = \alpha\psi(P_1).$$

We remark that $\ker \psi = E_1[n, \omega_R]$, so ψ is R-oriented, and does not depend on α. Now consider $\phi_2 = \psi \circ \phi$, of degree nd. Then we know that $\phi_2(P_0) = \psi(\alpha P_1 + y\omega_R P_1) = \alpha\psi(P_1)$. Since P_0 is an R-generator of $E_0[n]$, this allows to recover how ϕ_2 acts on the full n-torsion of $E_0[n]$.

In conclusion, we now have an isogeny ϕ_2 of degree dn and we know how ϕ_2 acts on the full group $E_0[n]$. We can then proceed with existing techniques.

All in all, let n_1 be the product of all prime powers we use for case 2, and n_2 the product of all prime powers we use for case 3. Let $n = n_1 n_2$. We build a n_2-isogeny $\psi : E_1 \to E_2$ using the method in the previous paragraph to obtain a $n_2 d$-isogeny $\phi_2 = \psi \circ \phi : E_0 \to E_2$ for which we know the action on the full $E_0[n]$-torsion. We need $n_1^2 n_2 > 4d$ to be able to recover ϕ_2 hence ϕ, via a $n_1^2 n_2^2$-isogeny $\Phi : E_0^r \times E_2^r \to E_0^r \times E_2^r$ in dimension $g = 2r$ (see Figure 6). Here the dimension r will depend on whether we can find a solution with $r = 1, 2, 4$ to $n_1^2 n_2^2 - n_2 d = \sum_{i=1}^{r} a_i^2$. Because of the common factor n_2, we require n_2 to be a sum of two squares to be able to work in dimension $g = 4$ (i.e., $r = 2$). In general we will need to work with $r = 4$ and so dimension 8 Abelian varieties.

Remark 4.2. Let E_0 be an ordinary elliptic curve over $\mathbb{F}_q$ with $q+1-t$ points and let $\mathrm{End}(E_0)$ have discriminant Δf^2. If t is odd then $\Delta f^2 = t^2 - 4q \equiv 1 \pmod 4$ and so $|\Delta| \equiv 3 \pmod 4$. So we know Δ is divisible by a prime congruent to 3 modulo 4. So a bad case is when t is odd and $|\Delta|$ is a prime.

Furthermore, while ψ does not depend on the torsion possibilities, we need to compute Φ for all the possibilities of the torsion we have. As explained before, in practice we compute two $n_1 n_2$-isogenies Φ_1, Φ_2 from $E_0^r \times E_2^r$. As a technical detail: to find the kernel of Φ_2 we need to know how $\tilde{\phi}_2$ acts on the n-torsion; this can be done from the knowledge of how ϕ_2 acts on the n-torsion through pairings and DLPs, because $\tilde{\phi}_2$ is the adjoint of ϕ_2 for the Weil pairing.

The complexity of the computation of the isogenies ψ, Φ itself will depend on the dimension where Φ lives, and if $n = \prod \ell_i^{e_i}$, of the largest prime ℓ_i and the degrees u_i of the field generated by the $\ell_i^{e_i}$-torsion points. More precisely, by [28, Lemma 5.7], if $n = \prod_{i=1}^{a} \ell_i^{e_i}$, computing an n-isogeny in dimension g can be done in time $\tilde{O}(ae\ell^g u \log q)$, where ℓ (resp. e) is a bound on the ℓ_i (resp. e_i), and u is a bound on the $lcm(u_i, u_j)$ where u_i is the degree of the field extension defined by the points of the $\ell_i^{e_i}$-torsion (so $u \leq n^2$).

The choice of prime power ℓ^e to use in practice will be a delicate balance between the size of ℓ, the degree of the field extension where the ℓ^e-torsion is defined, and the number of possibilities, mainly whether we are in case 2 or 3.

Remark 4.3. There is a non-trivial way to save time: rather than computing Φ from scratch from every possible choice of torsion information, we remark that we can reuse part of our construction of Φ.

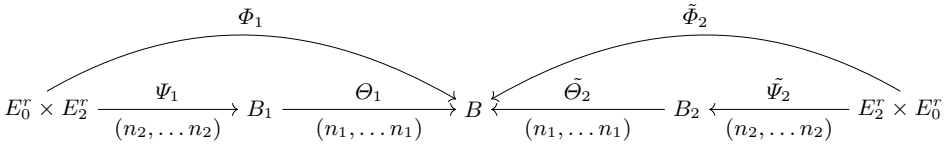


Fig. 6. This isogeny diagram corresponds to the explanation of how to address case 3. We use our knowledge of the $E_0[n]$ torsion group to compute the (probably non-smooth) $(n_2, \dots n_2)$-isogeny and guess the remaining (smooth) $(n_1, \dots n_1)$-isogeny ϕ_2 using a meet-in-the-middle computation (as detailed in Remark 4.3).

First, we can assume that we are given the torsion information of $\phi_2 : E_0 \to E_2$ in terms of matrices acting on basis elements (P_0, Q_0), (P_1', Q_1') of the $\ell_i^{e_i}$-torsion. We remark that the pairing information from Section 4.2 already gives this information on E_1, we have P_i a cyclic R-generator, and $Q_i = \omega_R P_i$, and the torsion information is expressed as a constraint on an endomorphism $\gamma \in R/\ell_i^{e_i}$, which we can recast as a matrix. We propagate this information through ψ, this involves sampling a new basis; we may pick P_1' a cyclic generator of $E_2[\ell_i^{e_i}]$ and let $Q_1' = \omega_R P_1'$ to simplify the computations. Then $\psi(P_1') = cQ_1'$, and we determine c by pairings and DLPs as in Section 4.2.

Now we have several choices of matrices for the action of ϕ_2 on $\ell_1^{e_1}$. We pick one choice, and use it to build the $\ell_1^{e_1}$-component of the kernel of $\Phi_1, \tilde{\Phi}_2$. We push the other $\ell_i^{e_i}$-basis points (using a product basis); this is where the compositum of the field of definition of the $\ell_1^{e_1}$ and $\ell_i^{e_i}$ torsion comes in. Then we make a choice of matrix for the action on $\ell_2^{e_2}$, determine the kernel of the next $\ell_2^{e_2}$ higher dimensional isogenies, and so on.

The key remark is that if we change our choice of matrix for $\ell_m^{e_m}$, we need to recompute the subsequent $\ell_i^{e_i}$ isogenies for $i \geq m$, but we can keep the ones already computed for $i < m$.

We illustrate this with a small example: assume that we want to use $n_1 = \ell_1 \ell_2$ with ℓ_1 split and ℓ_2 inert, and $n_2 = \ell_3 \ell_4$ a product of two different primes. We have $\ell_1 - 1$ possible choices for the action of ϕ on $E_0[\ell_1]$, $\ell_2 + 1$ possible choices for the action of ϕ on $E_0[\ell_1]$, and 2 possible choices for the action of ϕ on a subgroup of order ℓ_i of $E_0[\ell_i]$ for $i = 3, 4$. We can choose in which order to treat our primes, and it makes sense to pick them up by decreasing size. So suppose for instance that $\ell_3 > \ell_1 > \ell_2 > \ell_4$. Then via the above strategy, to compute all possibilities for Φ_1, we will compute in total $2(\ell_1 - 1)(\ell_2 + 1)2$ ℓ_4-isogenies, $2(\ell_1 - 1)(\ell_2 + 1)$ ℓ_2-isogenies, $2(\ell_1 - 1)$ ℓ_1-isogenies, and 2 ℓ_1-isogenies.

We have all the tools to bound the cost of recovering ϕ.

Theorem 4.3. *Let the notation and hypothesis be as at the beginning of Section 4. Specifically, let $\phi : E_0 \to E_1$ be an ascending R-oriented isogeny of degree d. Let $m \mid \Delta_R$ be a factor of Δ_R. Assume that R is m-locally primitive on E_0, and let $m = \prod \ell_i^{e_i}$. Denote the number of square roots of 1 modulo m by T, B*

a bound on the largest prime divisor ℓ_i of m, and u a bound on the $lcm(u_i, u_j)$, where u_i is the degree of the field of definitions of the points of $E_0[\ell_i^{e_i}]$.

Then we can recover $\phi : E_0 \to E_1$ in time $\tilde{O}((TuB^8 + T\sqrt{d/m})(\log q + \log d)^{O(1)})$.

Proof. First, dividing m by 2 or 4 if necessary, we can assume that $m \mid \Delta_R/4$. We can also assume that we know the factorisation of m, since we know the one of Δ_R.

If m is not coprime to d, we use standard tools such as case 1.

Following the notation used above, we take $n_2 = m$, and for n_1 a B'-powersmooth number coprime to Δ_R and of size $\Theta(\sqrt{d/m})$. We can always take a smoothness bound B' polynomial in $\log q + \log d$.

We compute the torsion information on the n_1 and n_2 torsion using Section 4.2. There are T possibilities for the n_2-torsion by Theorem 4.2, and $\tilde{O}(n_1)$ possibilities for the action on the n_1-torsion. By Corollaries 3.2 and 4.1 the cost of recovering this information is not the dominating step.

Computing the n_2-isogeny $\psi : E_1 \to E_2$ as in Remark 4.1 takes $\tilde{O}(Bu)$ field operations.

We apply the strategy of Remark 4.3 by computing the n_2-isogeny first, followed by the n_1-isogeny. We assume that we are in the worst case and that we need to work in dimension 8. The first n_2-isogeny costs $\tilde{O}(uB^8 \log^2(n_2) \log q)$. We also need to push the torsion information on the n_1-torsion, which costs $\tilde{O}(u'B^8 \log n_2 \log n_1 \log q)$. Here u' is a bound on the degree of the compositum field of the $\ell_i^{e_i}$-torsion and the $\ell_j^{e_j}$ torsion for $\ell_i \mid n_2$ and $\ell_j \mid n_1$, so since n_1 is B'-powersmooth, we have $u' \leq u(B')^2$. We need to perform these computations T times.

Then we need to compute the remaining n_1-isogeny and decide if we meet in the middle. This costs $\tilde{O}(B'^4 B'^8 \log^2(n_1) \log q)$. We need to do this computation $\tilde{O}(Tn_1)$ times. This gives us our final complexity result. □

Remark 4.4.

- A bound on u is given by $u = \min(O(m^2), O(B^4))$ in the ordinary case and $u = \min(O(m), O(B^2))$ in the supersingular case.
- In practice for our applications, the term $TuB^8 + T\sqrt{d/m}$ will be large enough to absorb the $\log q + \log d$ part into the $\tilde{O}$ factor, and we will use $\tilde{O}(TuB^8 + T\sqrt{d/m})$ as our complexity.
- If m is a sum of two squares, then heuristically we can work in dimension 4, and replace the term TuB^8 by TuB^4.
- If $\phi : E_0 \to E_1$ is an R-oriented cyclic descending isogeny of degree d, we can follow the same approach, *except* for case 2: ℓ is a prime dividing the conductor. Indeed, in that case, $K = (\ker \phi)[\ell]$ is a kernel of a strictly descending isogeny, but there are between $\ell-1$ and $\ell+1$ descending isogenies, so many more possibilities for K than in the case of a purely ascending isogeny when K is unique. Of course, the better strategy is to reconstruct the dual $\hat{\phi} : E_1 \to E_0$ instead, which is ascending.

5 Improving the computational isogeny problem when the conductor is large

In this section we apply the techniques from Section 4 to the ordinary isogeny problem. In particular, we will show how to use the pairings from Section 4.2 to improve the algorithm from [14]. Our main application is the case of volcanoes with a small crater.

5.1 Improved computation of an ascending isogeny.

Let E_0 and E_1 be ordinary elliptic curves over $\mathbb{F}_q$ with $q+1-t$ points. Let $t^2-4q = f^2\Delta$, where Δ is a fundamental discriminant and $f > 1$. Suppose $\mathrm{End}(E_1)$ has discriminant $\Delta_1 = \Delta f_1^2$, $\mathrm{End}(E_0)$ has discriminant $\Delta_0 = \Delta f_0^2$ with $f_0 = Nf_1$. We want to recover the ascending $\mathbb{F}_q$-rational N-isogeny $\phi : E_0 \to E_1$ of degree $N|f$. Since ϕ is $\mathbb{F}_q$-rational we have $\phi \circ \pi_q = \pi_q \circ \phi$, where π_q denotes the q-power Frobenius maps on E_0 and E_1, so ϕ is $\mathbb{Z}[\pi_q]$-oriented. In this section we will neglect factors polynomial in $\log q$.

Galbraith in [14, Theorem 2] gives an algorithm that can compute ϕ in $\tilde{O}(N^{1/2})$ operations over $\mathbb{F}_q$. Our first improvement concerns a heuristic assumption used in the algorithm. As explained in Section 2.3, a key task of the 2024 algorithm is to guess $\phi(P_0), \phi(Q_0)$ for a suitable basis P_0, Q_0 of $E_0[\ell]$. This was done in [14] by choosing P_0 and Q_0 to be Elkies primes (i.e., eigenvectors for the q-power Frobenius). But by Theorem 4.1, we can use inert primes just as well as split primes. This removes the need for an assumption about the distribution of Elkies primes.

Our second main improvement will come from cyclic self-pairings (or from our point of view, the use of degenerate conics), coming from a factor $m \mid \Delta_1$. The details of the self-pairings algorithm is outlined in Algorithm 2.

In that algorithm we restrict for simplicity to the case where m is odd and $\gcd(m, f) = 1$. This is because in that case we can directly use the Frobenius π_q to get the self-pairing information. Indeed, define $\tau = 2\pi_q - t$, which will be interpreted depending on the context as lying in $\mathrm{End}(E_0)$ and $\mathrm{End}(E_1)$. Note that $\tau = \phi \circ \alpha \circ \hat{\phi}$, where $\alpha \in \mathrm{End}(E_0)$ is the endomorphism corresponding to $\sqrt{\Delta}$. We also have that $\mathrm{Tr}(\tau) = 2\,\mathrm{Tr}(\pi_q) - 2t = 0$ and $\Delta(\tau) = \tau^2 = t^2 - 4q$, $N(\tau) = 4q - t^2$. In particular τ is, up to a factor 2, our canonical imaginary element for $\mathbb{Z}[\pi_q]$ from Definition 3.2. We can thus apply Section 4.2, using $\omega_R = \tau$. Since $e_m(P, \tau Q) = e_m(P, \pi_q Q)^2$, the situation is simplified because we can simply evaluate the Frobenius rather than a more general endomorphism of the form τ/b. We refer to Theorem 4.2 for the general case, when m may have a common factor with f.

We then illustrate how to use the self-pairing information to recover ϕ in Algorithm 3. This algorithm calls two functions Guess() and Kani().

As already explained, we need to compute $(\ker \Phi)[\ell]$ for various primes ℓ, and this is fully determined by a basis (P, Q) for $E_0[\ell]$ and the corresponding image $(P', Q') \in E_2[\ell]$ under $\psi \circ \phi$.

In our applications we will have a point $P_0 \in E_0[\ell]$ such that $RP_0 = E_0[\ell]$ and a point $P_1 \in E_1[\ell]$ such that $RP_1 = E_1[\ell]$. We know that $\phi(P_0) = \gamma(P_1)$ for some $\gamma = a + b\omega_R$, but we don't know (a, b). Indeed, we will extend this to $P_2 \in E_2[\ell]$ such that $\psi \circ \phi(P_0) = \gamma(P_2)$, using the fact that ψ is also R-oriented. The function $\mathsf{Guess}()$ will produce a list of candidates for the unknown coefficients (a, b) in γ. To be precise, it will take as input torsion information as a sequence of data $(P_0, P_2) \in E_0[\ell] \times E_2[\ell]$, and it will output a list of items, each of which is a tuple of candidates for the correct $(a, b) \in (\mathbb{Z}/\ell\mathbb{Z})^2$.

The notation A_ℓ represents the data needed for the Kani construction. For example, $A_{n_2} = (n_2, (P_0, Q_0), (S, 0)))$ represents the information that (P_0, Q_0) is a basis for $E_0[n_2]$ and that $\psi \circ \phi(P_0) = S$ and $\psi \circ \phi(Q_0) = 0$.

The function $\mathsf{Kani}()$ takes as input the torsion data $\ell, (P_0, P_2) \in E_0[\ell] \times E_2[\ell]$ and a candidate (a, b) and constructs some prefix of the map Φ of degree $(\ell, \ell, \ldots, \ell)$, by computing the ℓ-part of $\ker \Phi$. An extra detail is that we will apply this to an abelian variety B_1 that is already part-way along the path of Φ. This is handled by calculating the image of $\psi \circ \phi(P_0) = (a + b\omega_R)(P_2)$ on B_1 by working with the images of the points P_0 and P_2 on B_1. We will also abuse notation and use the function $\mathsf{Kani}()$ for computing the ℓ-part of $\ker \tilde{\Phi}$ corresponding to the right hand side of Figure 6.

Algorithm 3 puts everything together. It describes how to recover an ascending isogeny between two fixed elliptic curves by making use of the partial information gained from self-pairings and then using $\mathsf{Guess}()$ and $\mathsf{Kani}()$ to recover the remaining part.

Applying Theorem 4.3 to our situation, we obtain the following complexity.

Proposition 5.1. *Let E_0 and E_1 be ordinary elliptic curves over $\mathbb{F}_q$. Suppose $\mathrm{End}(E_1)$ has discriminant Δ_1 and $\mathrm{End}(E_0)$ has discriminant $\Delta_0 = \Delta_1 N^2$. Suppose there is an N-isogeny $\phi : E_0 \to E_1$ (in other words E_1 is directly above E_0 in the volcano). Suppose Δ_1 has a large divisor $m = q^a$ which has $O(\log(\log(q)))$ distinct prime factors.*

Then we can recover ϕ using

$$\max\left(\tilde{O}(q^{10a}), \tilde{O}(q^{(1-3a)/4})\right)$$

operations over $\mathbb{F}_q$.

Proof. In the notation of Theorem 4.3, we have $d = N$, and since $N^2|\Delta| = O(q)$, we have $N = O(q^{(1-a)/2})$. We also have $m = O(q^a)$ since $m \mid \Delta$, so $\sqrt{d/m} = O(q^{(1-3a)/4})$. We also take the trivial upper bounds $B = m$ and $u = m^2$. Finally $T = O(\log q)$ because of our assumptions on the number of distinct prime factors of m. Then we can bound the term TuB^8 by $\tilde{O}(q^{10a})$, and the term $T\sqrt{d/m}$ by $\tilde{O}(q^{(1-3a)/4})$. □

Remark 5.1. It remains to address the question of whether $|\Delta_1|$ contains a large divisor m with $O(\log \log q)$ distinct prime factors. Unfortunately this may not

Algorithm 2: Torsion recovery via self-pairings

Input : Ordinary elliptic curves E_0, E_1 over $\mathbb{F}_q$ such that $[\mathrm{End}(E_0) : \mathrm{End}(E_1)] = N$, an odd integer m such that $m \mid \Delta_1$, $(m, f) = 1$;

Output: $S_0 \in E_0(\mathbb{F}_q), \{S_j\}_{j=1}^T$ such that $\phi(S_0) = S_j$ for one $j \in [1, T]$.

1 Sample $P_i \in E_i[m]$ that is a generator for the $\mathbb{Z}[\pi_q]$-module $E_i[m]$ for $i = 0, 1$;
2 Define $\tau = 2\pi_q - t$ and set $Q_i = \tau(P_i)$ for $i = 0, 1$;
3 Compute the following pairings: $T_i = e_m(P_i, Q_i), i = 0, 1$;
4 Compute all possible $\{a_j\}_{j=1}^T$ such that $(T_1)^{a_j^2} = (T_0)^N$;
5 **return** $\big((P_0, Q_0, P_1, Q_1), \{a_j\}_{j=1}^T\big)$;

always be the case. If $|\Delta_1|$ is a primorial then divisors of it, even of size only $|\Delta_1|^{1/2}$, will have too many distinct prime factors.

However, it is well-known (see Theorem 430 of Section 22.10 of Hardy and Wright [15]) that the average number of distinct prime factors of integers up to X is $\log\log(X)$. Hence the required condition on m applies to most integers Δ_1 and our claim holds on average.

We also consider the *very* special case that Δ_1 has a large powersmooth factor in the following corollary.

Proposition 5.2. *Let notation and hypotheses be as in Proposition 5.1. In addition, suppose the discriminant Δ_1 has a factor $m \mid \Delta_1$ of size q^a which is B-powersmooth. Denote the number of square roots of 1 modulo m by T.*

Then we can recover $\phi : E_0 \to E_1$ in time

$$\max\Big(\tilde{O}(TB^{12}), \tilde{O}(T \cdot q^{(1-3a)/4})\Big)$$

Proof. This is immediate from Theorem 4.3 using the bound $u = B^4$ and the analysis from Proposition 5.1. □

5.2 Volcanoes with small crater

Let E_0 and E_1 be ordinary elliptic curves over $\mathbb{F}_q$ with $q + 1 - t$ points where $t^2 - 4q = f^2\Delta$ where Δ is the fundamental discriminant, and $|\Delta| = q^a$ with $a > 0$ small. We want to find an isogeny between them, but this time we do not assume that E_0 is directly below E_1.

If E_0 is on the crater and E_1 on the floor then the approach in Galbraith [14] has complexity $\tilde{O}(h_0 f^{1/2})$ operations over $\mathbb{F}_q$, where h_0 is the class number of Δ. Since $f = O(q^{(1-a)/2})$ and $h_0 = \tilde{O}(\sqrt{|\Delta_0|}) = \tilde{O}(q^{a/2})$, we get that $\tilde{O}(h_0 f^{1/2}) = \tilde{O}(q^{a/2} q^{(1-a)/4}) = \tilde{O}(q^{(1+a)/4})$. The approach in [14] dealt with the cases when

Algorithm 3: Vertical isogeny recovery in volcanoes with small crater

Input : Ordinary elliptic curves E_0, E_1 such that $[\mathrm{End}(E_0) : \mathrm{End}(E_1)] = N$
Output: A representation of $\phi : E_0 \to E_1$

Choose $n_2 \in \mathbb{Z}$ dividing the discriminant Δ that satisfies the conditions in Algorithm 2;
Choose t small distinct primes $\ell_1, \dots \ell_t$ and a (small) integer s such that $N < 3^s \cdot n_2 \cdot \prod_{i\in[1,t]} \ell_i^2 < 2N(c + 2t\log(t))^2)$ for a small constant c;
Set $n_1 = 4 \cdot 3^{\lceil s/2 \rceil} \ell_1 \cdots \ell_t$;
Write $3^s n_2^2 (\ell_1 \cdots \ell_t)^2 - n_2 N > 0$ as a sum of squares;
Run Algorithm 2 to get $(P_0, Q_0, P_1, Q_1, \{a_j\}_{j=1}^T)$ such that $\phi(Q_0) = a_j Q_1$ for one of the $j \in [1, T]$;
Compute $\psi : E_1 \to E_2$ with kernel $\langle Q_1 \rangle$;
Set up torsion information $S_\ell = (P_{0,\ell}, P_{2,\ell}) \in E_0[\ell] \times E_2[\ell]$ for each ℓ_i, and also $S_{3^{\lceil s/2 \rceil}} \in E_0[3^{\lceil s/2 \rceil}] \times E_2[3^{\lceil s/2 \rceil}]$;
for $j \in [1, T]$ **do**
    Set $A_{n_2} = (n_2, (P_0, Q_0), (a_j\psi(P_1), 0)))$;
    Compute $\Psi_1, B_1 \leftarrow \mathsf{Kani}(E_0^4 \times E_2^4, A_{n_2})$; `// The partial Kani isogenies`
    Compute the kernel of the dual isogeny to Ψ_1 and use Kani to compute the image B_2 of the isogeny $\tilde{\Psi}_2 : E_2^4 \times E_0^4 \to B_2$;
    Compute $S'_\ell = \Psi_1((P_0, Q_0, S_j))$ and $S''_\ell = \tilde{\Psi}_2((P_0, Q_0, S_j))$ for each $i = 1, \dots t$; `// Push the torsion basis`
    Let $M = \tilde{O}(n_1)$ be the number of possible guesses for the images of ϕ on n_1-torsion;
    for $i \in [1, M]$ **do**
        Compute bases $A_{3^s/2}, A_{\ell_1}, \dots, A_{\ell_t}$ from $\mathsf{Guess}(S_{3^{\lceil s/2 \rceil}}, \{S_{\ell_1}, \dots S_{\ell_t}\})[i]$;
        $\Theta_1, B \leftarrow \mathsf{Kani}(B_1, n_1, A_{3^s/2}, A_{\ell_1}, \dots, A_{\ell_t})$;
        Use Kani to compute the corresponding $\tilde{\Theta}_2$ from B_2 and call the codomain B';
        `// Test if the two Kani isogenies have met in the middle:`
        if $B \cong B'$ **then**
            return $\Psi_2 \circ \Theta_2 \circ \Theta_1 \circ \Psi_1$ as a representation of ϕ;
return $\perp$;

E_0 and E_1 are not on the crater by descending to the floor and solving the isogeny problem in $\tilde{O}(q^{1/4})$ time.

We can improve these results using the results of Section 5.1, in the case when Δ has $O(\log\log(q))$ prime factors.

Theorem 5.1. *Assume that $|\Delta| = q^a$ has $O(\log\log(q))$ prime factors, and that $a < 1/41$. Then we can find a connecting isogeny between E_0 and E_1 in time $\tilde{O}(q^{(1-a)/4})$.*

Proof. We will show that we can climb up from E_0 to the crater in $\tilde{O}(q^{(1-a)/4})$ when a is small enough. Doing the same for E_1, we can then solve the isogeny problem in the crater in time $\tilde{O}(h_0^{1/2})$ where $h_0 = \tilde{O}(|\Delta|^{1/2}) = \tilde{O}(q^{a/2})$ is the number of curves on the crater, which is the class number of $\mathbb{Q}(\sqrt{\Delta})$. So $\tilde{O}(h_0^{1/2}) = \tilde{O}(q^{a/4})$, and the dominating step is the climbing up phase (since a is small).

We let $m = |\Delta|/4$ if $4 \mid \Delta$, and $m = |\Delta|$ otherwise. We still have $m = \Theta(q^a)$. If we knew the curve E_0' on the crater directly above E_0, we could apply Proposition 5.1 to climb up in time $\tilde{O}(q^{(1-3a)/4})$, when $a < 1/43$ (for larger a, the dominating complexity becomes $\tilde{O}(q^{10a})$). Since we do not know E_0', we need to test all the curves on the crater, for a total time of $\tilde{O}(h_0 q^{(1-3a)/4}) = \tilde{O}(q^{(1-a)/4})$. This is the dominating complexity over $\tilde{O}(q^{10a})$ when $a < 1/41$. □

Remark 5.2. If all primes dividing $f = O(q^{(1-a)/2})$ are smaller than $q^{a'}$ then one can compute ascending isogenies from E_0, E_1 to the crater in $\tilde{O}(q^{2a'})$ operations and solve the isogeny problem in the crater in $\tilde{O}(|\Delta|^{1/2}) = \tilde{O}(q^{a/2})$ operations, for a total cost of $\tilde{O}(q^{2a'} + q^{a/2})$. So Theorem 5.1 is mainly interesting in the case where f has at least one large enough prime divisor.

6 A special case attack attempt on SCALLOP

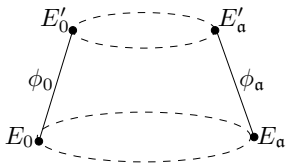
In what follows we outline how the general approach from Section 4 might potentially be leveraged to attack the SCALLOP group action when the fundamental discriminant has a large smooth factor. This indicates that any cryptographic protocol using this group action should exclude (almost) smooth discriminants from their parameter selection.

SCALLOP [11] is a group action on oriented supersingular elliptic curves, initially proposed with the purpose of improving upon the CSIDH group action. There have since been two variants: SCALLOP-HD [7] and PEARL-SCALLOP [1], that improve upon some subroutines and parameter choices but rely on similar security assumptions. Our attack will apply in the special case that the discriminant has a large smooth factor, so we will focus on PEARL-SCALLOP, whose parameter generation uses larger fundamental discriminants.

We consider the SCALLOP class group action by a non-maximal order $\mathcal{O} \subset K$ of discriminant $\Delta = \Delta_0 f^2$. Suppose we are given two oriented curves (E_0, ι_0) and $(E_\mathfrak{a}, \iota_\mathfrak{a})$ on the floor. We would like to recover an ideal $\mathfrak{a}$ in $Cl(\mathcal{O})$ such that $\mathfrak{a} \star (E_0, \iota_0) = (E_\mathfrak{a}, \iota_\mathfrak{a})$. In this section, we only look at classical attacks.

Standard attack. The standard classical attack for this problem is direct meet-in-the-middle [1, Sect. 3.5]. Since the class group is of order $\sqrt{|\Delta|}$ this gives a complexity of $\tilde{O}(|\Delta^{1/4}|) = \tilde{O}(|\Delta_0|^{1/4} f^{1/2})$ field operations. So for a security parameter, λ, we can expect $\log|\Delta| = 4\lambda$ in order to avoid this attack.

The "Climbing Up" approach. We consider the same attack from Galbraith [14] in the ordinary case, that uses the Kani machinery to climb up to the crater from each curve, and then use meet-in-the-middle on the crater to join the paths (see the figure below).



This is the approach we improved in Section 5.1 by using self pairings.

Recall, this approach involves guessing which curve on the crater is "directly above" the curves, i.e. the curve $E'_{\mathfrak{a}}$ such that $[\mathrm{End}(E'_{\mathfrak{a}}) : \mathrm{End}(E_{\mathfrak{a}})] = f$. Note that we usually already know E'_0. For brevity we outline how to recover $\phi_{\mathfrak{a}}$, though recovering ϕ_0 follows symmetrically. There are $\tilde{O}(\sqrt{\Delta_0})$ curves that lie on the crater. For each of these candidate curves, we must then guess the action of $\phi_{\mathfrak{a}}$ on some torsion points. Galbraith [14] shows that this guessing requires a complexity of $O(\sqrt{f})$; so in total the approach requires $(|\Delta_0|f)^{1/2}$ field operations to complete.

From here, we compute a meet-in-the-middle search on the crater to get an ideal mapping $E'_0 \to E'_{\mathfrak{a}}$. This requires $\tilde{O}(|\Delta_0|^{1/4})$ field operations. Let $\mathcal{O}_0$ be the maximal order of $\mathcal{O}$. So far we have recovered an ideal $\mathfrak{b}$ such that $\mathfrak{b} \star E'_0 = E'_{\mathfrak{a}}$, but where $[\mathfrak{b}]$ is in $Cl(\mathcal{O}_0)$ instead of $Cl(\mathcal{O})$. Notice that we have an exact sequence

$$0 \to G \to Cl(\mathcal{O}) \to Cl(\mathcal{O}_0) \to 0,$$

where $G := \ker(Cl(\mathcal{O}) \to Cl(\mathcal{O}_0))$ is of order $\Theta(f)$ that has a very explicit description (see Theorem 7.24 and Equation (7.25) of Cox [9]).

Taking an arbitrary preimage of $[\mathfrak{b}]$ in $Cl(\mathcal{O})$, say $[\mathfrak{b}']$, we may still be off from our target ideal by an element in G. With yet another meet-in-the-middle on the G action we recover this element with a complexity of $O(f^{1/2})$ operations.

Thus, to recover the ascending isogeny, compute the meet-in-the-middle on the crater, and recover the necessary element of G, the final complexity is

$$\tilde{O}(|\Delta_0|^{1/2} f^{1/2}) + \tilde{O}(|\Delta_0|^{1/4}) + \tilde{O}(f^{1/2}) = \tilde{O}(|\Delta_0|^{1/2} f^{1/2})$$

field operations. Note, this complexity is worse than the direct meet-in-the-middle attack from before.

Self-pairings attack. We can follow the same idea as the "Climbing Up" approach, but hope to gain some savings using self-pairings. In the following theorem, we compute the cost of recovering the ascending isogeny as outlined in Algorithm 3.

Theorem 6.1. *Fix an imaginary quadratic order K and a non-maximal order $\mathcal{O} \subset K$ with discriminant $\Delta = \Delta_0 f^2$, where Δ_0 is a fundamental discriminant. Further, suppose Δ_0 has a B-smooth factor m. Denote the number of square roots of 1 modulo m by T.*

Let (E_0, ι_0) and $(E_\mathfrak{a}, \iota_\mathfrak{a})$ be two oriented curves such that there exists an ideal $\mathfrak{a} \in Cl(\mathcal{O})$ such that $\mathfrak{a} \star (E_0, \iota_0) = (E_\mathfrak{a}, \iota_\mathfrak{a})$.

Then we can recover $\mathfrak{a}$ in

$$\tilde{O}\Big(T|\Delta_0|^{1/2}B^{10} + T(f|\Delta_0|/m)^{1/2}\Big)$$

field operations.

Proof. First, suppose we are given two supersingular curves $E_\mathfrak{a}, E'_\mathfrak{a}$ over $\bar{\mathbb{F}}_p$ such that $[\mathrm{End}(E'_\mathfrak{a}) : \mathrm{End}(E_\mathfrak{a})] = f$.

Then recovering the isogeny $\phi_\mathfrak{a} : E_\mathfrak{a} \to E'_\mathfrak{a}$ can be done in $\tilde{O}(T(B^{10} + f^{1/2}m^{-1/2})\log(q)^{O(1)})$ field operations by Theorem 4.3 (here we bound u by B^2).

Since we must also guess which curve $E'_\mathfrak{a}$ on the crater is above $E_\mathfrak{a}$, we must again multiply the complexity above by the number of guesses, which is $\tilde{O}\sqrt{|\Delta_0|}$. This gives a complexity of $\tilde{O}(T|\Delta_0|^{1/2}B^{10} + T(f|\Delta_0|/m)^{1/2})$ field operations.

From here, we have the same task as before of computing the precise ideal $\mathfrak{b}$, requiring yet another meet-in-the-middle computation in the action G. This costs $\tilde{O}(|\Delta_0|^{1/4} + f^{1/2})$ field operations, which is negligible compared to the climbing up phase. □

Let us assume that m has $O(\log\log q)$ distinct factors, so that $T = O(\log q)$. The complexity becomes $\tilde{O}(|\Delta_0|^{1/2}B^{10} + (f|\Delta_0|/m)^{1/2})$ field operations.

We see that for this method to be better than the direct attack on the floor which costs $\tilde{O}(|\Delta_0|^{1/4}f^{1/2})$, we need:

- $|\Delta_0|$ to be not too large compared to f so that $|\Delta_0|^{1/4}B^{10} \ll f^{1/2}$.
- m large enough compared to Δ_0, so that $m \gg |\Delta_0|^{1/2}$, but with few enough prime factors, which forces m not to be too large either: $m \leq B^{O(\log\log q)}$. In particular, we need $|\Delta_0|^{1/2} \ll B^{O(\log\log q)}$.

These make for tight restrictions on the choice of Δ_0 and f even when $|\Delta_0| \ll f$.

For instance, if $B^{10} = f^{1/2}|\Delta_0|^{-1/4-\epsilon}$, and $m = |\Delta_0|^{1/2+\epsilon}$ (with few enough prime factors), we obtain a complexity of $O(f^{1/2}|\Delta_0|^{1/4-\epsilon})$.

In the most favourable scenario for our attack, Δ_0 has a large smooth divisor $m = \prod \ell_i$, such that each ℓ_i is congruent to 1 modulo 4 so that we can work in dimension $g = 4$ rather than 8, and $\ell_i \mid p+1$ so the ℓ_i-torsion is already rational. The reason we need ℓ_i congruent to 1 modulo 4 to be able to work in dimension 4

is that this allows us to write a suitable multiple m' of m as a sum of two squares, which allows us to build a suitable m'-isogeny in dimension 2 given by a matrix of integers. But for SCALLOP we also have access to endomorphisms in $\mathcal{O}$, not just integers, and we can build an m'-isogeny in dimension 2 given by $\begin{pmatrix} \gamma_1 & \overline{\gamma_2} \\ -\gamma_2 & \overline{\gamma_1} \end{pmatrix}$ as long as we find $\gamma_1, \gamma_2 \in \mathcal{O}$ such that $m' = N(\gamma_1) + N(\gamma_2)$. This can allow us to relax the conditions on the ℓ_i. The total complexity of the attack would then be $\tilde{O}(|\Delta_0|^{1/2}B^4 + (f|\Delta_0|/m)^{1/2})$.

We finish by an example showing that our attack does not apply to the public SCALLOP parameter sets.

Example 6.1.

- The original version of SCALLOP uses $|\Delta_0| = O(1)$. This is too small to be of use, since it means that m is $O(1)$ and the self-pairings do not provide enough information.
- For $\lambda = 128$ bits of classical security, the authors of PEARL-SCALLOP use a discriminant $\Delta = \Delta_0 f^2$ of 512 bits, where Δ_0 has 6 factors, $\log(|\Delta_0|) = 256$, and $\log(f) = 128$. Here Δ_0 is too large compared to f for our attack to apply: we would have too many curves on the crater to try.
- For $\lambda = 256$ bits of security the authors of PEARL-SCALLOP [1, Sect. 4.1] suggest a discriminant of 1024 bits where Δ_0 has 4 factors, $\log(|\Delta_0|) = 258$, and $\log(f) = 390$.
  The self-pairings attack described above would give an improvement over state-of-the-art for a smoothness bound B such that
  $$|\Delta_0|^{1/2}B^{10} < |\Delta_0|^{1/4}f^{1/2}.$$
  This gives $\log B < 13$ (of course this is a back of the envelope computation, the exact value would depend on the exact hidden factors). Their Δ_0 contains a 2^{16}-smooth factor m of size $\log(m) \approx 33 \ll 258/2$. So Δ_0 is not smooth enough for our attack to apply either.

References

1. Bill Allombert, Jean-François Biasse, Jonathan Komada Eriksen, Péter Kutas, Chris Leonardi, Aurel Page, Renate Scheidler, and Márton Tot Bagi. Faster SCALLOP from non-prime conductor suborders in medium sized quadratic fields. In Tibor Jager and Jiaxin Pan, editors, *PKC 2025*, volume 15676 of *Lecture Notes in Computer Science*, pages 333–363. Springer, 2025.
2. Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. MSP Open Book Series, 2020.
3. Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 423–447. Springer, 2023.
4. Wouter Castryck, Thomas Decru, Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, Damien Robert, and Benjamin Wesolowski. Isogeny interpolation for elliptic curves, and applications. Presented by Wouter Castryck at

the Sixteenth Algorithmic Number Theory Symposium, Massachusetts Intitute of Technology, July 15, 2024. https://antsmath.org/ANTSXVI/schedule.html.
5. Wouter Castryck, Marc Houben, Simon-Philipp Merz, Marzio Mula, Sam van Buuren, and Frederik Vercauteren. Weak instances of class group action based cryptography via self-pairings. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023*, volume 14083 of *Lecture Notes in Computer Science*, pages 762–792. Springer, 2023.
6. Wouter Castryck, Marc Houben, Frederik Vercauteren, and Benjamin Wesolowski. On the decisional Diffie–Hellman problem for class group actions on oriented elliptic curves. *Research in Number Theory*, 8(4):99, 2022.
7. Mingjie Chen, Antonin Leroux, and Lorenz Panny. SCALLOP-HD: group action from 2-dimensional isogenies. In *PKC 2024*, volume 14603 of *Lecture Notes in Computer Science*, pages 190–216. Springer, 2024.
8. Leonardo Colò and David Kohel. Orienting supersingular isogeny graphs. *Journal of Mathematical Cryptology*, 14(1):414–437, 2020.
9. David A. Cox. *Primes of the form $x^2 + ny^2$: Fermat, Class Field Theory, and Complex Multiplication*. AMS Chelsea Publishing/American Mathematical Society, third edition, 2022.
10. B. Edixhoven, G. van der Geer, and B. Moonen. Abelian varieties. Book project, 2012. http://van-der-geer.nl/~gerard/AV.pdf.
11. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023*, volume 13940 of *Lecture Notes in Computer Science*, pages 345–375. Springer, 2023.
12. Mireille Fouquet, Josep M. Miret, and Javier Valera. Distorting the volcano. *Finite Fields and Their Applications*, 49:108–125, 2018.
13. Steven D. Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS J. Comput. Math.*, 2:118–138, 1999.
14. Steven D. Galbraith. Climbing and descending tall isogeny volcanos. *Research in Number Theory*, 11(7), 2025.
15. G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers (5th edition)*. Oxford, 1979.
16. Sorina Ionica and Antoine Joux. Pairing the volcano. *Math. Comput.*, 82(281):581–603, 2013.
17. Bruce W Jordan, Allan G Keeton, Bjorn Poonen, Eric M Rains, Nicholas Shepherd-Barron, and John T Tate. Abelian varieties isogenous to a power of an elliptic curve. *Compositio Mathematica*, 154(5):934–959, 2018.
18. Ernst Kani. The number of curves of genus two with elliptic differentials. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1997:122 – 93, 1997.
19. D. R. Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.
20. Joseph Macula and Katherine E. Stange. Extending class group action attacks via sesquilinear pairings. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024*, volume 15486 of *Lecture Notes in Computer Science*, pages 371–395. Springer, 2024.
21. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 448–471. Springer, 2023.

22. Josep M. Miret, Ramiro Moreno, Daniel Sadornil, Juan Tena, and Magda Valls. An algorithm to compute volcanoes of 2-isogenies of elliptic curves over finite fields. *Applied Mathematics and Computation*, 176(2):739–750, 2006.
23. Hiroshi Onuki. On oriented supersingular elliptic curves. *Finite Fields Their Appl.*, 69:101777, 2021.
24. Thomas Pornin. Optimized discrete logarithm computation for faster square roots in finite fields. Cryptology ePrint Archive, Paper 2023/828, 2023.
25. Damien Robert. Some applications of higher dimensional isogenies to elliptic curves (preliminary version). *IACR Cryptol. ePrint Arch.*, page 1704, 2022.
26. Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 472–503. Springer, 2023.
27. Damien Robert. The module action for isogeny based cryptography. IACR Eprint 2024/1556, October 2024.
28. Damien Robert. On the efficient representation of isogenies (a survey). IACR Eprint 2024/1071, June 2024.
29. Katherine E Stange. Sesquilinear pairings on elliptic curves. *arXiv preprint arXiv:2405.14167*, 2024.
30. John Tate. Endomorphisms of abelian varieties over finite fields. *Inventiones mathematicae*, 2:134–144, 1966.

A Tate-Weil-Cartier pairings

The goal of this section is to revisit the construction of self pairings from [5].

Consider a commutative diagram of isogenies of abelian varieties:

$$\begin{array}{ccc} A_1 & \xrightarrow{\sigma_1} & A_2 \\ \downarrow{\scriptstyle \psi_1} & & \downarrow{\scriptstyle \psi_2} \\ B_1 & \xrightarrow{\sigma_2} & B_2 \end{array}$$

We know that the Weil-Cartier pairing [10, Ch. 11] is non degenerate: $e_{\sigma_1} : A_1[\sigma_1] \times \hat{A}_2[\hat{\sigma}_1] \to \mathbb{G}_m$, where we denote by $A_1[\sigma_1]$ the kernel of σ_1. By analogy with the standard Tate pairing, we can define a Tate-Weil-Cartier pairing by restricting to the subgroup $A_1[\sigma_1, \psi_1] = A_1[\sigma_1] \cap A_1[\psi_1]$ of $A_1[\sigma_1]$.

Theorem A.1. *There is a well defined Tate-Weil-Cartier pairing:*

$$T_{\psi_1}^{\sigma_1} : A_1[\sigma_1, \psi_1] \times \hat{A}_2[\hat{\sigma}_1]/\hat{\psi}_2(\hat{B}_2[\hat{\sigma}_2]) \to \mathbb{G}_m$$

which can be computed for $P_1 \in A_1[\sigma_1, \psi_1]$ and $Q_2 \in \hat{A}_2[\hat{\sigma}_1]$ by

$$T_{\psi_1}^{\sigma_1}(P_1, Q_2) = e_{\sigma_1}(P_1, Q_2) = e_{\psi_1}(P_1, \hat{\sigma}_2 Q') \tag{3}$$

where $\hat{\psi}_2 Q' = Q$.

Proof. Let

$$A_1[\sigma_1, \psi_1]^{\perp} = \{P \in \hat{A}_2[\hat{\sigma}_1] : e_{\sigma_1}(Q, P) = 1 \ \forall \ Q \in A_1[\sigma_1, \psi_1]\}$$

be the orthogonal of $A_1[\sigma_1, \psi_1]$ in $\hat{A}_2[\hat{\sigma}_1]$ for the Weil-Cartier pairing. We then have a well defined pairing:

$$A_1[\sigma_1, \psi_1] \times \hat{A}_2[\hat{\sigma}_1]/A_1[\sigma_1, \psi_1]^\perp \to \mathbb{G}_m$$

It is not hard to see that $\hat{\psi}_2(\hat{B}_2[\hat{\sigma}_2])$ is in this orthogonal, because $e_{\sigma_1}(P_1, \hat{\psi}_2 Q_2) = e_{\psi_2 \circ \sigma_1}(P_1, Q_2) = e_{\sigma_2 \circ \psi_1}(P_1, Q_2) = e_{\sigma_2}(\psi_1 P_1, Q_2) = 1$, since $\psi_1(P_1) = 0$ when $P_1 \in A_1[\sigma_1, \psi_1]$. Hence $T^{\sigma_1}_{\psi_1}$ is well defined.

We can use the compatibility of the Weil-Cartier pairings with isogenies to prove Eq. (3). Indeed, we have $\hat{\psi}_1 \hat{\sigma}_2 Q' = \hat{\sigma}_1 \hat{\psi}_2 Q' = 0$, so $e_{\psi_1}(P_1, \hat{\sigma}_2 Q')$ is well defined. And we compute: $e_{\psi_1}(P_1, \hat{\sigma}_2 Q') = e_{\sigma_2 \psi_1}(P_1, Q') = e_{\psi_2 \sigma_1}(P_1, Q') = e_{\sigma_1}(P_1, \hat{\psi}_2 Q') = e_{\sigma_1}(P_1, Q_2)$. □

Remark A.1.

- The non degenerate pairing on the left $A_1[\sigma_1, \psi_1] \times \hat{A}_2[\hat{\sigma}_1]/\hat{\psi}_2(\hat{B}_2[\hat{\sigma}_2]) \to \mathbb{G}_m$ is also non degenerate on the right whenever $\hat{\psi}_2(\hat{B}_2[\hat{\sigma}_2])$ is the full orthogonal, e.g., when $\#\hat{B}_2[\hat{\sigma}_2] = \#\hat{A}_2[\hat{\sigma}_1]$ and $\#\hat{B}_2[\hat{\sigma}_2, \hat{\psi}_2] = \#A_1[\sigma_1, \psi_1]$.
- The proof of Eq. (3) shows that $Q' \in B_2[\sigma_2 \psi_1]$ and that $T^{\sigma_1}_{\psi_1}(P_1, Q_2) = e_{\sigma_2 \psi_1}(Q') = e_{\sigma_2 \psi_1}(Q')$ which shows that $T^{\sigma_1}_{\psi_1}$ is also induced by the Weil-Cartier pairing $e_{\sigma_2 \psi_1}$, which highlight the symmetric role of the ψ_i, σ_i.
- Still by the compatibility of the Weil-Cartier pairing and isogenies, whenever σ_1 is a d-isogeny, that is we have a contragradient isogeny $\tilde{\sigma}_1$ satisfying $\sigma_1 \circ \tilde{\sigma}_1 = \tilde{\sigma}_1 \circ \sigma_1 = [d]$, we have: $e_{\sigma_1}(P, Q) = e_d(P, Q') = e_d(P', Q)$ where $\sigma_1(Q') = Q$ and $\tilde{\sigma}_1(P') = P$.

Example A.1.

- Take $E_1/\mathbb{F}_q$ an elliptic curve defined over a finite field $\mathbb{F}_q$, $\psi_1 = \psi_2 = \psi : E_1 \to E_2$ a rational isogeny, and $\sigma_i = \hat{\pi}_q - 1$ on E_i. Since $E[\hat{\pi}_q - 1] = E[\pi_q - q]$ and $E[\pi_q - 1] = E(\mathbb{F}_q)$, we obtain a non degenerate pairing on the left $E_1[\pi_q - q, \psi] \times E_1(\mathbb{F}_q)/\hat{\psi}(E_2(\mathbb{F}_q)) \to \mathbb{G}_m$. We remark that if $E_1[\psi] \subset E_1[m]$ with $q \equiv 1 \mod m$, then $E_1[\pi_q - q, \psi] = E_1(\mathbb{F}_q)[\psi]$.
  As a further special case, taking $\psi = [m]$ (where m satisfy the above condition), we obtain a non degenerate pairing on the left $e : E_1(\mathbb{F}_q)[m] \times E_1(\mathbb{F}_q)/m(E_1(\mathbb{F}_q)) \to \mathbb{G}_m$, which can be computed as $e(P, Q) = e_{\hat{\pi_q} - 1}(P, Q) = e_m(P, (\pi_q - 1)Q')$ where $mQ' = Q$. We recover the standard Tate pairing.
- Take $E/\mathbb{F}_q$ an elliptic curve, and R a primitive orientation on E by a quadratic imaginary order of discriminant Δ_R. Let ω_R be the canonical imaginary element of Definition 3.2. Since $\hat{\omega} = -\omega$, we have a non degenerate Cartier-Weil pairing $e_{\omega_R} : E[\omega_R] \times E[\omega_R] \to \mathbb{G}_m$. Since $E[\omega_R]$ is cyclic, we see that e_{ω_R} gives a cyclic self pairing.
  Now if $N(\omega_R) = m_1 m_2$, then we can construct a commutative diagram as above, by taking $\sigma_1 = \omega_R, \psi_1 = [m_1], \psi_2 = \hat{\omega_R}, \sigma_2 = [m_2]$, to obtain a non degenerate pairing on the left (hence also on the right) $E[\omega_R, m_1] \times E[\omega_R]/\omega_R E[m_2] \to \mathbb{G}_m$, which can be computed as $e_\omega(P, Q) = e_{m_1}(P, m_2 Q')$ where $\omega_R Q' = Q$.

Let P_0 be a $\mathbb{Z}[\omega_R]$-generator of $E[m_1 m_2]$. Then $Q_0 = \omega_R P_0$ is a generator of $E[\omega_R]$, $P = m_2 P_0$ is a $\mathbb{Z}[\omega_R]$-generator of $E[m]$, and $Q = \omega_R P = m_2 \omega_R P_0$ a generator of $E[\omega_R, m_1]$. By the result above, we find that $e_\omega(Q, Q) = e_{m_1}(\omega_R P, P)$. We recover the distorted pairing from Theorem 4.1 (up to a sign), and this is the self pairing built in [5].

8 Rational isogeny evaluation

Publication data. Gustavo Banegas, Valerie Gilchrist, Anaëlle Le Dévéhat, Benjamin Smith. Fast and Frobenius: Rational Isogeny Evaluation over Finite Fields. LatinCrypt 2023.

My Contribution. In this work all four coauthors contributed equally in the theoretical development, code, experiments, and writing.

Fast and Frobenius: Rational Isogeny Evaluation over Finite Fields

Gustavo Banegas[1], Valerie Gilchrist[2], Anaëlle Le Dévéhat[3], Benjamin Smith[3]

[1] Qualcomm France SARL, Valbonne, France
[2] Université Libre de Bruxelles and FRIA, Brussels, Belgium
[3] Inria and Laboratoire d'Informatique de l'École polytechnique, Institut Polytechnique de Paris, Palaiseau, France

Abstract. Consider the problem of efficiently evaluating isogenies $\phi : \mathcal{E} \to \mathcal{E}/H$ of elliptic curves over a finite field $\mathbb{F}_q$, where the kernel $H = \langle G \rangle$ is a cyclic group of odd (prime) order: given $\mathcal{E}$, G, and a point (or several points) P on $\mathcal{E}$, we want to compute $\phi(P)$. This problem is at the heart of efficient implementations of group-action- and isogeny-based post-quantum cryptosystems such as CSIDH. Algorithms based on Vélu's formulæ give an efficient solution when the kernel generator G is defined over $\mathbb{F}_q$, but for general isogenies G is only defined over some extension $\mathbb{F}_{q^k}$, even though $\langle G \rangle$ as a whole (and thus ϕ) is defined over the base field $\mathbb{F}_q$; and the performance of Vélu-style algorithms degrades rapidly as k grows. In this article we revisit isogeny evaluation with a special focus on the case where $1 \leq k \leq 12$. We improve Vélu-style evaluation for many cases where $k = 1$ using special addition chains, and combine this with the action of Galois to give greater improvements when $k > 1$.

1 Introduction

Faced with the rising threat of quantum computing, demand for quantum-secure, or post-quantum, cryptographic protocols is increasing. Isogenies have emerged as a useful candidate for post-quantum cryptography thanks to their generally small key sizes, and the possibility of implementing post-quantum group actions which offer many simple post-quantum analogues of classical discrete-log-based algorithms (see e.g. [26]).

A major drawback of isogeny-based cryptosystems is their relatively slow performance compared with many other post-quantum systems. In this paper, we improve evaluation times for isogenies of many prime degrees $\ell > 3$ given a generator of the kernel; these computations are the fundamental building blocks of most isogeny-based cryptosystems. Specifically, we propose simple alternative

* Authors listed in alphabetical order: see `https://www.ams.org/profession/leaders/CultureStatement04.pdf`. This work was funded in part by a FRIA grant by the National Fund for Scientific Research (F.N.R.S.) of Belgium, by the French Agence Nationale de la Recherche through ANR CIAO (ANR-19-CE48-0008), and by a *Plan France 2030* grant managed by the Agence Nationale de la Recherche (ANR-22-PETQ-0008). Date of this document: 2025-03-18.

differential addition chains to enumerate points of (subsets of) the kernel more efficiently. This speeds up many ℓ-isogeny computations over the base field by a factor depending on ℓ, and also permits a full additional factor-of-k speedup for ℓ-isogenies over $\mathbb{F}_q$ whose kernel generators are defined over an extension $\mathbb{F}_{q^k}$.

Our techniques have constructive and destructive applications. First, accelerating basic isogeny computations can speed up isogeny-based cryptosystems. The methods in §4 apply for many $\ell > 3$, so they would naturally improve the performance of commutative isogeny-based schemes such as CSIDH [5], and CSI-FiSh [4] and its derivatives (such as [12] and [14]), which require computing many ℓ-isogenies for various primes ℓ. They may also improve the performance of other schemes like SQISign [16], which computes many ℓ-isogenies in its signing process. (We discuss applications further in §6.)

In §5 we focus on rational isogenies with irrational kernels; our methods there could further accelerate the improvements of [13] for Couveignes–Rostovtsev–Stolbunov key exchange (CRS) and related protocols of Stolbunov [11,24,27,28]. This is a small step forward on the road to making CRS a practical "ordinary" fallback for CSIDH in the event of new attacks making specific use of the full supersingular isogeny graph (continuing the approach of [6], for example).

Our results also have applications in cryptanalysis: the best classical and quantum attacks on commutative isogeny-based schemes involve computing massive numbers of group actions, each comprised of a large number of ℓ-isogenies (see e.g. [3] and [8]). Any algorithm that reduces the number of basic operations per ℓ-isogeny will improve the effectiveness of these attacks.

Proof-of-concept implementations of our algorithms in SageMath are available at `https://github.com/vgilchri/k-velu`. The scripts include operation-counting code to verify the counts claimed in this article.

Disclaimer. In this paper, we quantify potential speedups by counting finite field operations. Real-world speed increases depend on many additional variables including parameter sizes; the application context; implementation choices; specificities of the runtime platform (including architecture, vectorization, and hardware acceleration); and the availability of optimized low-level arithmetic.

2 Background

We work over (extensions of) the base field $\mathbb{F}_q$, where q is a power of a prime $p > 3$. The symbol ℓ always denotes a prime $\neq p$. In our applications, $3 < \ell \ll p$.

Elliptic curves. For simplicity, elliptic curves are supposed to be in a general Weierstrass form $\mathcal{E} : y^2 = f(x)$. Our algorithms focus on *Montgomery models*

$$\mathcal{E} : By^2 = x(x^2 + Ax + 1) \qquad \text{where} \qquad B(A^2 - 4) \neq 0\,.$$

but our results extend easily to other models such as twisted Edwards and short Weierstrass models. The multiplication-by-m map is denoted by $[m]$. The q-power Frobenius endomorphism is $\pi : (x, y) \mapsto (x^q, y^q)$.

Field operations. While the curve $\mathcal{E}$ will always be defined over $\mathbb{F}_q$, we will often work with points defined over $\mathbb{F}_{q^k}$ for $k \geq 1$. We write $\mathbf{M}$, $\mathbf{S}$, and $\mathbf{a}$ for the cost of multiplication, squaring, and adding (respectively) in $\mathbb{F}_{q^k}$. We write $\mathbf{C}$ for the cost of multiplying an element of $\mathbb{F}_{q^k}$ by an element of $\mathbb{F}_q$ (typically a curve constant, or an evaluation-point coordinate). Note that $\mathbf{C} \approx (1/k)\mathbf{M}$ (when k is not too large). Later, we will write $\mathbf{F}$ for the cost of evaluating the Frobenius map on $\mathbb{F}_{q^k}$; see §5.1 for discussion on this.

x-only arithmetic. Montgomery models are designed to optimize x-only arithmetic (see [20] and [10]). The `xADD` operation is

$$\texttt{xADD} : (x(P), x(Q), x(P-Q)) \longmapsto x(P+Q)\,;$$

it can be computed at a cost of $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$ using the formulæ

$$\begin{cases} X_+ = Z_- \left[(X_P - Z_P)(X_Q + Z_Q) + (X_P + Z_P)(X_Q - Z_Q)\right]^2 \,, \\ Z_+ = X_- \left[(X_P - Z_P)(X_Q + Z_Q) - (X_P + Z_P)(X_Q - Z_Q)\right]^2 \end{cases} \tag{1}$$

where $(X_P : Z_P)$, $(X_Q : Z_Q)$, $(X_+ : Z_+)$, and $(X_- : Z_-)$ are the x-coordinates $x(P)$, $x(Q)$, $x(P+Q)$, and $x(P-Q)$, respectively (so $x(P) = \frac{X_P}{Z_P}$, and so on).

The `xDBL` operation is

$$\texttt{xDBL} : x(P) \longmapsto x([2]P)\,;$$

it can be computed at a cost of $2\mathbf{M} + 2\mathbf{S} + \mathbf{C} + 4\mathbf{a}$ using the formulæ

$$\begin{cases} X_{[2]P} = (X_P + Z_P)^2 (X_P - Z_P)^2 \,, \\ Z_{[2]P} = (4X_P Z_P)((X_P - Z_P)^2 + ((A+2)/4)(4X_P Z_P)) \,. \end{cases} \tag{2}$$

Isogenies. Let $\mathcal{E}_1, \mathcal{E}_2$ be elliptic curves over a finite field $\mathbb{F}_q$. An isogeny $\phi : \mathcal{E}_1 \to \mathcal{E}_2$ is a non-constant morphism mapping the identity point of $\mathcal{E}_1$ to the identity point of $\mathcal{E}_2$. Such a morphism is automatically a homomorphism. For more details see [25, Chapter 3, §4]. The kernel of ϕ is a finite subgroup of $\mathcal{E}_1$, and vice versa: every finite subgroup $\mathcal{G}$ of $\mathcal{E}_1$ determines a separable *quotient isogeny* $\mathcal{E}_1 \to \mathcal{E}_1/\mathcal{G}$. The *kernel polynomial* of ϕ is

$$D(X) := \prod_{P \in S} (X - x(P))$$

where $S \subset \mathcal{G}$ is any subset satisfying

$$S \cap -S = \emptyset \qquad \text{and} \qquad S \cup -S = \mathcal{G} \setminus \{0\}\,. \tag{3}$$

Every separable isogeny $\phi : \mathcal{E}_1 \to \mathcal{E}_2$ defined over $\mathbb{F}_q$ can be represented by a rational map in the form

$$\phi : (x, y) \longmapsto \big(\phi_x(x), \phi_y(x, y)\big) \tag{4}$$

with

$$\phi_x(x) = \frac{N(x)}{D(x)^2} \qquad \text{and} \qquad \phi_y(x, y) = c \cdot y \frac{d\phi_x}{dx}(x) \tag{5}$$

where D is the kernel polynomial of ϕ, N is a polynomial derived from D, and c is a normalizing constant in $\mathbb{F}_q$.

Vélu's formulæ. Given a curve $\mathcal{E}$ and a finite subgroup $\mathcal{G} \subset \mathcal{E}$, Vélu [29] gives explicit formulæ for the rational functions that define a separable isogeny $\phi : \mathcal{E} \to \mathcal{E}' := \mathcal{E}/\mathcal{G}$ with kernel $\mathcal{G}$, as well as the resulting codomain curve $\mathcal{E}'$. Although the quotient curve $\mathcal{E}'$ and the isogeny ϕ are defined up to isomorphism, Vélu's formulæ construct the unique *normalized* isogeny (i.e. with $c = 1$ in (5)). See Kohel's thesis [18, §2.4] for a modern treatment of Vélu's results.

3 Evaluating isogenies

Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_q$, and let $\langle G \rangle$ be a subgroup of prime order ℓ (where ℓ is not equal to the field characteristic p). We suppose $\langle G \rangle$ is defined over $\mathbb{F}_q$; then, the quotient isogeny $\phi : \mathcal{E} \to \mathcal{E}/\langle G \rangle$ is also defined over $\mathbb{F}_q$.

When we say $\langle G \rangle$ is defined over $\mathbb{F}_q$, this means $\langle G \rangle$ is *Galois stable*: that is, $\pi(\langle G \rangle) = \langle G \rangle$ (where π is the q-power Frobenius endomorphism). We will mostly be concerned with algorithms taking $x(G)$ as an input, so it is worth noting that

$$x(G) \in \mathbb{F}_{q^{k'}} \qquad \text{where} \qquad k' := \begin{cases} k & \text{if } k \text{ is odd}\,, \\ k/2 & \text{if } k \text{ is even}\,. \end{cases}$$

The set of projective x-coordinates of the nonzero kernel points is

$$\mathcal{X}_G := \big\{ (X_P : Z_P) = x(P) : P \in \langle G \rangle \setminus \{0\} \big\} \subset \mathbb{P}^1(\mathbb{F}_{q^{k'}})\,;$$

each X_P/Z_P corresponds to a root of the kernel polynomial $D(X)$, and vice versa. If $\#\langle G \rangle$ is an odd prime ℓ, then $\#\mathcal{X}_G = (\ell - 1)/2$.

3.1 The isogeny evaluation problem

We want to evaluate the isogeny $\phi : \mathcal{E} \to \mathcal{E}/\langle G \rangle$. More precisely, we want efficient solutions to the problem of Definition 1:

Definition 1 (Isogeny Evaluation). *Given an elliptic curve $\mathcal{E}$ over $\mathbb{F}_q$, a list of points $(P_1, \ldots, P_n)$ in $\mathcal{E}(\mathbb{F}_q)$, and a finite subgroup $\mathcal{G}$ of $\mathcal{E}$ corresponding to the separable isogeny $\phi : \mathcal{E} \to \mathcal{E}/\mathcal{G}$, compute $(\phi(P_1), \ldots, \phi(P_n))$.*

In most cryptographic applications n is relatively small, especially compared to ℓ. We do *not* assume the codomain curve $\mathcal{E}/\mathcal{G}$ is known; if required, an equation for $\mathcal{E}/\mathcal{G}$ can be interpolated through the image of well-chosen evaluation points.

For each separable isogeny ϕ of degree d defined over $\mathbb{F}_q$, there exists a sequence of primes $(\ell_1, \ldots, \ell_n)$ and a sequence of isogenies $(\phi_1, \ldots, \phi_n)$, all defined over $\mathbb{F}_q$, such that $\phi_n \circ \cdots \circ \phi_1$ and

- $\phi_i = [\ell_i]$ (the non-cyclic case) or
- ϕ_i has cyclic kernel of order ℓ_i.

The kernel of ϕ_1 is $\ker\phi \cap \mathcal{E}[\ell_1]$, and so on. The maps $[\ell_i]$ can be computed in $O(\log \ell_i)$ $\mathbb{F}_q$-operations, so we reduce quickly to the case where ϕ has prime degree ℓ, assuming the factorization of d is known (as it is in our applications).

In general, the isogeny evaluation problem can be reduced to evaluating the map $\alpha \mapsto D(\alpha)$, where D is the kernel polynomial and α is in $\mathbb{F}_q$ or some $\mathbb{F}_q$-algebra (see e.g. [2, §4]). The polynomial D need not be explicitly computed.

3.2 The Costello–Hisil algorithm

The Costello–Hisil algorithm [9], generalized in [23], is the state-of-the-art for evaluating isogenies of Montgomery models. This algorithm is a variation of Vélu's formulæ working entirely on the level of x-coordinates, using the fact that for an ℓ-isogeny ϕ with kernel $\langle G\rangle$, the rational map on x-coordinates is

$$\phi_x(x) = x\cdot\left(\prod_{i=1}^{(\ell-1)/2}\left(\frac{x\cdot x([i]G)-1}{x-x([i]G)}\right)\right)^2. \tag{6}$$

Moving to projective coordinates $(U:V)$ such that $x = U/V$ and using the fact that $\mathcal{X}_G = \{(x([i]G):1) : 1 \le i \le (\ell-1)/2\}$, Eq. (6) becomes

$$\phi_x\big((U:V)\big) = (U':V'), \quad \begin{cases} U' = U\big[\prod_{(X_Q:Z_Q)\in\mathcal{X}_G}(UX_Q - VZ_Q)\big]^2, \\ V' = V\big[\prod_{(X_Q:Z_Q)\in\mathcal{X}_G}(UZ_Q - VX_Q)\big]^2. \end{cases} \tag{7}$$

Algorithm 1 (from [9]) and Algorithm 2 (our space-efficient variant) compute ϕ_x at a series of input points using an efficient evaluation of the expressions in (7). For the moment, we assume that we have subroutines

- `KernelPoints` (see §4): given $(X_G : Z_G)$, returns $\mathcal{X}_G$ as a list.
- `KernelRange` (see §4): a *generator* coroutine which, given $(X_G : Z_G)$, constructs and yields the elements of $\mathcal{X}_G$ to the caller one by one.
- `CrissCross` [9, Algorithm 1]: takes $(\alpha,\beta,\gamma,\delta)$ in $\mathbb{F}_{q^k}^4$ and returns $(\alpha\delta+\beta\gamma, \alpha\delta-\beta\gamma)$ in $\mathbb{F}_{q^k}^2$ at a cost of $2\mathbf{M}+2\mathbf{a}$.

4 Accelerating Vélu: faster iteration over the kernel

Let $\mathcal{E}/\mathbb{F}_q$ be an elliptic curve, and let G be a point of prime order ℓ in $\mathcal{E}$. For simplicity, in this section we will assume that G is defined over $\mathbb{F}_q$, but all of the results here apply when G is defined over an extension $\mathbb{F}_{q^k}$: in that case $\mathbf{M}$, $\mathbf{S}$, and $\mathbf{a}$ represent operations in the extension field $\mathbb{F}_{q^k}$, while $\mathbf{C}$ represents multiplication of an element of $\mathbb{F}_{q^k}$ by a curve constant of the subfield $\mathbb{F}_q$ (which is roughly k times cheaper than $\mathbf{M}$). We return to the case where $k > 1$ in §5.

Algorithm 1: Combines Algorithms 3 and 4 from [9] to evaluate an ℓ-isogeny of Montgomery models at a list of input points. The total cost is $2n\ell\mathbf{M} + 2n\mathbf{S} + ((n+1)(\ell+1) - 2)\mathbf{a}$, *plus* the cost of `KernelPoints`.

Input: The x-coordinate $(X_G : Z_G)$ of a generator G of the kernel of an ℓ-isogeny ϕ, and a list of evaluation points $((U_i : V_i) : 1 \le i \le n)$
Output: The list of images $((U_i' : V_i') = \phi_x((U_i : V_i)) : 1 \le i \le n)$

$((X_1, Z_1), \ldots, (X_{(l-1)/2}, Z_{(l-1)/2})) \leftarrow$ `KernelPoints`$((X_G : Z_G))$ `// See §4`
for $1 \le i \le (\ell-1)/2$ **do**
  $(\hat{X}_i, \hat{Z}_i) \leftarrow (X_i + Z_i, X_i - Z_i)$ `// 2a`
for $i = 1$ ***to*** n **do**
  $(\hat{U}_i, \hat{V}_i) \leftarrow (U_i + V_i, U_i - V_i)$ `// 2a`
  $(U_i', V_i') \leftarrow (1, 1)$
  for $j = 1$ ***to*** $(\ell-1)/2$ **do**
    $(t_0, t_1) \leftarrow$ `CrissCross`$(\hat{X}_j, \hat{Z}_j, \hat{U}_i, \hat{V}_i)$ `// 2M + 2a`
    $(U_i', V_i') \leftarrow (t_0 \cdot U_i', t_1 \cdot V_i')$ `// 2M`
  $(U_i', V_i') \leftarrow (U_i \cdot (U_i')^2, V_i \cdot (V_i')^2)$ `// 2M + 2S`
return $((U_1', V_1'), \ldots, (U_n', V_n'))$

Algorithm 2: A generator-based version of Algorithm 1, with much lower space requirements when $\ell \gg n$. The total cost is $2n\ell\mathbf{M} + 2n\mathbf{S} + (2n + (\ell-1)(n+1))\mathbf{a}$, *plus* the cost of a full run of `KernelRange`.

Input: The x-coordinate $(X_G : Z_G)$ of a generator G of the kernel of an ℓ-isogeny ϕ, and a list of evaluation points $((U_i : V_i) : 1 \le i \le n)$
Output: The list of images $((U_i' : V_i') = \phi_x((U_i : V_i)) : 1 \le i \le n)$

for $1 \le i \le n$ **do**
  $(\hat{U}_i, \hat{V}_i) \leftarrow (U_i + V_i, U_i - V_i)$ `// 2a`
  $(U_i', V_i') \leftarrow (1, 1)$
for $(X : Z)$ ***in*** `KernelRange`$((X_G : Z_G))$ **do** `// See §4`
  $(\hat{X}, \hat{Z}) \leftarrow (X + Z, X - Z)$ `// 2a`
  for $1 \le i \le n$ **do**
    $(t_0, t_1) \leftarrow$ `CrissCross`$(\hat{X}, \hat{Z}, \hat{U}_i, \hat{V}_i)$ `// 2M + 2a`
    $(U_i', V_i') \leftarrow (t_0 \cdot U_i', t_1 \cdot V_i')$ `// 2M`
for $1 \le i \le n$ **do**
  $(U_i', V_i') \leftarrow (U_i \cdot (U_i')^2, V_i \cdot (V_i')^2)$ `// 2M + 2S`
return $((U_1', V_1'), \ldots, (U_n', V_n'))$

4.1 Kernel point enumeration and differential addition chains

We now turn to the problem of enumerating the set $\mathcal{X}_G$. This process, which we call *kernel point enumeration*, could involve constructing the entire set (as in `KernelPoints`) or constructing its elements one by one (for `KernelRange`).

For $\ell = 2$ and 3, there is nothing to be done because $\mathcal{X}_G = \{(X_G : Z_G)\}$; so from now on we consider the case $\ell > 3$.

We allow ourselves two curve operations for kernel point enumeration: `xADD` and `xDBL`. In §5, where G is assumed to be defined over a nontrivial extension of the base field, we will also allow the Frobenius endomorphism.

Every algorithm constructing a sequence of elements of $\mathcal{X}_G$ using a series of `xADD` and `xDBL` instructions corresponds to a *modular differential addition chain*.

Definition 2. *A **Modular Differential Addition Chain (MDAC)** for a set $S \subset \mathbb{Z}/\ell\mathbb{Z}$ is a sequence of integers $(c_0, c_1, c_2, \ldots, c_n)$ such that*

1. *every element of S is represented by some $c_i \pmod{\ell}$,*
2. *$c_0 = 0$ and $c_1 = 1$, and*
3. *for each $1 < i \leq n$ there exist $0 \leq j(i), k(i), d(i) < i$ such that $c_i \equiv c_{j(i)} + c_{k(i)} \pmod{\ell}$ and $c_{j(i)} - c_{k(i)} \equiv c_{d(i)} \pmod{\ell}$.*

Algorithms to enumerate $\mathcal{X}_G$ using `xADD` and `xDBL` correspond to MDACs $(c_0, \ldots, c_n)$ for $\{1, \ldots, (\ell-1)/2\}$: the algorithm starts with $x([c_0]G) = x(0) = (1 : 0)$ and $x([c_1]G) = x(G) = (X_G : Z_G)$, then computes each $x([c_i]G)$ using

$$x([c_i]G) = \begin{cases} \texttt{xADD}(x([c_{j(i)}]G), x([c_{k(i)}]G), x([c_{d(i)}]G)) & \text{if } d(i) \neq 0\,, \\ \texttt{xDBL}([c_{j(i)}]G) & \text{if } d(i) = 0\,. \end{cases}$$

4.2 Additive kernel point enumeration

The classic approach is to compute $\mathcal{X}_G$ using repeated `xADD`s. Algorithm 3 is Costello and Hisil's `KernelPoints` [9, Algorithm 2], corresponding to the MDAC $(0, 1, 2, 3, \ldots, (\ell-1)/2)$ computed by repeatedly adding 1 (except for 2 which is computed by doubling 1). The simplicity of this MDAC means that Algorithm 3 converts to a `KernelRange` with a small internal state: to generate the next $(X_{i+1} : Z_{i+1})$, we only need the values of $(X_i : Z_i)$, $(X_{i-1} : Z_{i-1})$, and $(X_1 : Z_1)$.

4.3 Replacing `xADD`s with `xDBL`s

Comparing x-only operations on Montgomery curves, replacing an `xADD` with an `xDBL` trades $2\mathbf{M}$ and $2\mathbf{a}$ for $1\mathbf{C}$. We would therefore like to replace as many `xADD`s as possible in our kernel enumeration with `xDBL`s.

As a first attempt, we can replace Line 4 of Algorithm 3 with

$$(X_i : Z_i) \leftarrow \begin{cases} \texttt{xDBL}((X_{i/2} : Z_{i/2})) & \text{if } i \text{ is even}, \\ \texttt{xADD}\big((X_{i-1} : Z_{i-1}), (X_G : Z_G), (X_{i-2}, Z_{i-2})\big) & \text{if } i \text{ is odd}. \end{cases}$$

But applying this trick systematically requires storing many more intermediate values, reducing the efficiency of `KernelRange`. It also only replaces half of the `xADD`s with `xDBL`s, and it turns out that we can generally do much better.

Algorithm 3: Basic kernel point enumeration by repeated addition.
Uses exactly 1 `xDBL` and $(\ell-5)/2$ `xADD` operations (for prime $\ell > 3$).

Input: The x-coordinate $(X_G : Z_G) = x(G)$ of a point G of order ℓ in $\mathcal{E}(\mathbb{F}_q)$
Output: $\mathcal{X}_G$ as a list

$(X_1 : Z_1) \leftarrow (X_G : Z_G)$
$(X_2 : Z_2) \leftarrow \texttt{xDBL}\big((X_G : Z_G)\big)$
for $i = 3$ ***to*** $(\ell-1)/2$ **do** `// Invariant:` $(X_i : Z_i) = x([i]G)$
  $(X_i : Z_i) \leftarrow \texttt{xADD}\big((X_{i-1} : Z_{i-1}), (X_G : Z_G), (X_{i-2}, Z_{i-2})\big)$
return $\big((X_1 : Z_1), \ldots, (X_{(\ell-1)/2} : Z_{(\ell-1)/2})\big)$

4.4 Multiplicative kernel point enumeration

We can do better for a large class of ℓ by considering the quotient

$$M_\ell := \big(\mathbb{Z}/\ell\mathbb{Z}\big)^\times / \langle \pm 1 \rangle \,.$$

(note: M_ℓ is a quotient of the *multiplicative* group.) For convenience, we write

$$m_\ell := \#M_\ell = (\ell-1)/2 \,.$$

We can now reframe the problem of enumerating $\mathcal{X}_G$ as the problem of enumerating a complete set of representatives for M_ℓ. The MDAC of Algorithm 3 computes the set of representatives $\{1, 2, \ldots, m_\ell\}$, but for the purposes of enumerating $\mathcal{X}_G$, *any* set of representatives will do. Example 1 is particularly useful.

Example 1. Suppose 2 generates M_ℓ. This is the case if 2 is a primitive element modulo ℓ—that is, if 2 has order $(\ell-1)$ modulo ℓ—but also if 2 has order $(\ell-1)/2$ modulo ℓ and $\ell \equiv 3 \pmod 4$. In this case

$$M_\ell = \{2^i \bmod \ell : 0 \le i < m_\ell\} \,,$$

so $(0, 1, 2, 4, 8, \ldots, 2^{m_\ell})$ is an MDAC for M_ℓ using *only* doubling, and *no* differential additions. The corresponding `KernelPoints` replaces *all* of the `xADD`s in Algorithm 3 with cheaper `xDBL`s, trading $(\ell-5)\mathbf{M} + (\ell-5)\mathbf{a}$ for $(\ell-5)/2$ $\mathbf{C}$. The corresponding `KernelRange` is particularly simple: each element depends only on its predecessor, so the state consists of a single $(X_i : Z_i)$.

How often does this trick apply? The quantitative form of Artin's primitive root conjecture (see [30]) says that $M_\ell = \langle 2 \rangle$ for a little over half of all ℓ. Experimentally, 5609420 of the first 10^7 odd primes ℓ satisfy $M_\ell = \langle 2 \rangle$.

One might generalize Example 1 to other generators of M_ℓ: for example, if $M_\ell = \langle 3 \rangle$, then we could find an MDAC for $\{3^i \bmod \ell : 0 \le i < (\ell-1)/2\}$. But this is counterproductive: x-only tripling is *slower* than differential addition.

4.5 Stepping through cosets

What can we do when $M_\ell \neq \langle 2 \rangle$? A productive generalization is to let

$$A_\ell := \langle 2 \rangle \subseteq M_\ell \quad \text{and} \quad a_\ell := \#A_\ell \,,$$

and to try to compute a convenient decomposition of M_ℓ into cosets of A_ℓ. Within each coset, we can compute elements using repeated `xDBL`s as in Example 1; then, it remains to step from one coset into another using differential additions.

This can be done in a particularly simple way for the primes ℓ such that

$$M_\ell = \langle 2, 3 \rangle \,, \qquad \text{so} \qquad M_\ell = \bigsqcup_{i=0}^{m_\ell/a_\ell - 1} 3^i A_\ell \,. \tag{$*$}$$

We can move from the i-th to the $(i+1)$-th coset using the elementary relations

$$\begin{cases} c \cdot 2^{j+1} + c \cdot 2^j = 3c \cdot 2^j \\ c \cdot 2^{j+1} - c \cdot 2^j = c \cdot 2^j \end{cases} \qquad \text{for all integers } c \text{ and } j \geq 0. \tag{8}$$

In particular, having enumerated $3^i A_\ell$ by repeated doubling, we can compute an element of $3^{i+1} A_\ell$ by applying a differential addition to any two consecutive elements of $3^i A_\ell$ (and the difference is the first of them). Algorithm 4 minimizes storage overhead by using the last two elements of the previous coset to generate the first element of the next one. The `KernelRange` of Algorithm 4 therefore has an internal state of only two x-coordinates—so not only is it faster than the `KernelRange` of Algorithm 3, but it also has a smaller memory footprint.

Algorithm 4: Kernel enumeration for $\ell > 3$ satisfying $(*)$. Cost: $(1 - 1/a_\ell) \cdot m_\ell$ `xDBL`s and $m_\ell/a_\ell - 1$ `xADD`s.

Input: Projective x-coordinate $(X_G : Z_G)$ of the generator G of a cyclic subgroup of order ℓ in $\mathcal{E}(\mathbb{F}_q)$, where ℓ satisfies $(*)$.
Output: $\mathcal{X}_G$ as a list

$(a, b) \leftarrow (a_\ell, m_\ell/a_\ell)$
for $i = 0$ ***to*** $b - 1$ **do** `// Invariant:` $(X_{ai+j} : Z_{ai+j}) = x([3^i 2^{i(a-2)+(j-1)}]G)$
  if $i = 0$ **then**
    $(X_1 : Z_1) \leftarrow (X_G : Z_G)$
  else `// Compute new coset representative`
    $(X_{ai+1} : Z_{ai+1}) \leftarrow \texttt{xADD}\big((X_{ai} : Z_{ai}), (X_{ai-1} : Z_{ai-1}), (X_{ai-1} : Z_{ai-1})\big)$
  for $j = 2$ ***to*** a **do** `// Exhaust coset by doubling`
    $(X_{ai+j} : Z_{ai+j}) \leftarrow \texttt{xDBL}\big((X_{ai+j-1} : Z_{ai+j-1})\big)$
return $\big((X_1 : Z_1), \ldots, (X_{(\ell-1)/2} : Z_{(\ell-1)/2})\big)$

Algorithm 4 performs better the closer a_ℓ is to m_ℓ. In the best case, when $A_\ell = M_\ell$, it uses $m_\ell - 1$ `xDBL`s and no `xADD`s at all. The worst case is when the order of 2 in M_ℓ is as small as possible: that is, $\ell = 2^k - 1$. In this case $a_\ell = k$, and compared with Algorithm 3 we still reduce the `xADD`s by a factor of k.

4.6 The remaining primes

While 1878 of the 2261 odd primes $\ell \leq 20000$ satisfy $(*)$, there are still 383 primes that do not. We can, to some extent, adapt Algorithm 4 to handle these primes, but on a case-by-case basis and with somewhat less satisfactory results.

For example, the CSIDH-512 parameter set specifies 74 isogeny-degree primes

$$\ell = 3, 5, 7, 11, 13, \ldots, 367, 373, \text{ and } 587\,.$$

All but seven of these ℓ satisfy $(*)$: the exceptions are $\ell = 73$, 97, 193, 241, 313, and 337. Table 1 lists a candidate decomposition of M_ℓ for each of these ℓ. In each case, we need to produce an element of either $5A_\ell$ or $7A_\ell$. This can certainly be done using previously-computed elements, but this requires a larger internal state and a more complicated execution pattern, depending on ℓ.

Table 1. Primes ℓ in the CSIDH-512 parameter set that do not satisfy $(*)$.

Prime ℓ	a_ℓ	$[M_\ell : \langle 2, 3\rangle]$	Coset decomposition of M_ℓ	Notes
73	9	2	$M_{73} = A_{73} \sqcup 3A_{73} \sqcup 5A_{73} \sqcup 5 \cdot 3A_{73}$	
97	24	2	$M_{97} = A_{97} \sqcup 5A_{97}$	3 is in A_{97}
193	48	2	$M_{193} = A_{193} \sqcup 5A_{193}$	3 is in A_{193}
241	12	2	$M_{241} = \left(\bigsqcup_{i=0}^{4} 3^i A_{241}\right) \sqcup \left(\bigsqcup_{i=0}^{4} 7 \cdot 3^i A_{241}\right)$	
307	51	3	$M_{307} = A_{307} \sqcup 5A_{307} \sqcup 7A_{307}$	3 is in A_{313}
313	78	2	$M_{313} = A_{313} \sqcup 5A_{313}$	3 is in A_{193}
337	21	2	$M_{337} = \left(\bigsqcup_{i=0}^{3} 3^i A_{337}\right) \sqcup \left(\bigsqcup_{i=0}^{3} 5 \cdot 3^i A_{337}\right)$	

Example 2. Consider $\ell = 97$. Now $3 \equiv 2^{19} \pmod{97}$, so 3 is in A_{97}, and in fact $M_{97} = A_{97} \sqcup 5A_{97}$. To adapt Algorithm 4 to this case, we can still enumerate A_{97} using repeated doubling. Then, we must construct an element of $5A_{97}$ from elements of A_{97}, using a differential addition like $5 \cdot 2^i = 2^{i+2} + 2^i$ (difference $3 \cdot 2^i$) or $5 \cdot 2^i = 2^{i+1} + 3 \cdot 2^i$ (difference 2^i). Each involves near powers of 2 (modulo 97), but also $3 \cdot 2^i \equiv 2^{i+19} \pmod{97}$, which must be stored while enumerating A_{97}. This gives an algorithm using one `xADD` and 48 `xDBL`s, just like Algorithm 4, but with a slightly larger state and a more complicated execution pattern specific to $\ell = 97$. Alternatively, after enumerating A_{97}, we could redundantly recompute $3 = 1 + 2$ (difference 1) to get 5 as $1 + 4$ (difference 3) or $2 + 3$ (difference 1).

Ultimately, there does not seem to be a "one size fits all" generalization of Algorithm 4 for enumerating $\mathcal{X}_G$ without a more complicated state or redundant recomputations. We can get reasonable results for many ℓ not satisfying $(*)$ by finding a good MDAC for $M_\ell/\langle 2, 3\rangle$ and then using Algorithm 4 to exhaust the coset containing each representative but the savings are generally not optimal.

4.7 (In)Compatibility with Vélusqrt

It is natural to ask whether these techniques can be used to further accelerate the Vélusqrt algorithm [2], which evaluates isogenies of large prime degree ℓ in $\widetilde{O}(\sqrt{\ell})$ time (with $O(\sqrt{\ell})$ space). Vélusqrt never explicitly computes all of $\mathcal{X}_G$. Instead, it relies on the existence of a decomposition

$$S := \{1, 3, 5, \ldots, \ell - 2\} = (I + J) \sqcup (I - J) \sqcup K \tag{9}$$

where I, J, and K are sets of integers of size $O(\sqrt{\ell})$ such that the maps $(i, j) \to i + j$ and $(i, j) \to i - j$ are injective with disjoint images. In [2], these sets are

$$\begin{aligned} I &:= \{2b(2i + 1) : 0 \le i < b'\} && (\text{“giant steps”}), \\ J &:= \{2j + 1 : 0 \le j < b\} && (\text{“baby steps”}), \\ K &:= \{4bb' + 1, \ldots, \ell - 4, \ell - 2\} && (\text{“the rest”}), \end{aligned}$$

where $b := \lfloor \sqrt{\ell - 1}/2 \rfloor$ and $b' := \lfloor (\ell - 1)/4b \rfloor$.

The key thing to note here is that this decomposition is essentially additive, and the elements of I, J, and K are in arithmetic progression. Algorithm 4, however, is essentially multiplicative: it works with subsets in geometric progression. We cannot exclude the existence of subsets I, J, and K of size $O(\sqrt{\ell})$ satisfying (9) and which are amenable to enumeration by a variant of Algorithm 4 for some ℓ, but it seems difficult to construct nontrivial and useful examples.

5 Irrational kernel points: exploiting Frobenius

Now suppose G is defined over a nontrivial extension $\mathbb{F}_{q^k}$ of $\mathbb{F}_q$, but $\langle G \rangle$ is defined over the subfield $\mathbb{F}_q$: that is, it is Galois-stable. In particular, the q-power Frobenius endomorphism π of $\mathcal{E}$, which maps points in $\mathcal{E}(\mathbb{F}_{q^k})$ to their conjugates under $\mathrm{Gal}(\mathbb{F}_{q^k}/\mathbb{F}_q)$, maps $\langle G \rangle$ into $\langle G \rangle$, and hence restricts to an endomorphism of $\langle G \rangle$. But since the endomorphisms of $\langle G \rangle$ are $\mathbb{Z}/\ell\mathbb{Z}$, and Frobenius has no kernel (so π is not 0 on $\langle G \rangle$), it must act as multiplication by an eigenvalue $\lambda \neq 0$ on $\langle G \rangle$. The precise value of λ is not important here, but we will use the fact that λ has order k in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ and order k' in $(\mathbb{Z}/\ell\mathbb{Z})^\times/\langle \pm 1 \rangle$.

Now let

$$F_\ell := \langle \lambda \rangle \subseteq M_\ell \qquad \text{and} \qquad c_F := [M_\ell : F_\ell] = m_\ell/k' \,.$$

Let R_0 be a set of representatives for M_ℓ/F_ℓ; set $S_0 := \{[r]G : r \in R_0\}$, and note

$$\#S_0 = (\ell - 1)/k' \,.$$

5.1 The cost of Frobenius

We want to use the Galois action to replace many **M** and **S** with a few **F**. For this to be worthwhile, **F** must be cheap: and it is, even if this is not obvious

given the definition of the Frobenius map on $\mathbb{F}_{q^k}$ as q-th powering. Indeed, we do not compute Frobenius by powering. Instead, we use the fact that Frobenius is $\mathbb{F}_q$-linear, acting as a $k \times k$ matrix (with entries in $\mathbb{F}_q$) on the coefficient vectors of elements in $\mathbb{F}_{q^k}$. The form of this matrix, and the cost of applying it, depends on the basis of $\mathbb{F}_{q^k}/\mathbb{F}_q$. For example:

1. If $k = 2$ and $\mathbb{F}_{q^2} = \mathbb{F}_q(\sqrt{\Delta})$, then Frobenius simply negates $\sqrt{\Delta}$ and the matrix is $\mathrm{diag}(1, -1)$, so $\mathbf{F} \approx 0$.
2. If $\mathbb{F}_{q^k}/\mathbb{F}_q$ is represented with a normal basis, then the matrix represents a cyclic permutation, and again $\mathbf{F} \approx 0$.

Even in the worst case where the basis of $\mathbb{F}_{q^k}/\mathbb{F}_q$ has no special Galois structure, $\mathbf{F}$ is just the cost of multiplying a k-vector by a $k \times k$ matrix over $\mathbb{F}_q$: that is, k^2 multiplications and $k^2 - k$ additions. This is close to the cost of one "schoolbook" $\mathbb{F}_{q^k}$-multiplication; so when $k \leq 12$, we have $\mathbf{F} \approx \mathbf{M}$.

5.2 Galois orbits

Each point $P \in \mathcal{E}(\mathbb{F}_{q^k})$ is contained in a *Galois orbit* containing all the conjugates of P. The kernel subgroup $\langle G \rangle$ breaks up (as a set) into *Galois orbits*: if we write

$$\mathcal{O}_P := \{P, \pi(P), \ldots, \pi^{k-1}(P)\} \qquad \text{for } P \in \mathcal{E}(\mathbb{F}_{q^k}),$$

then

$$\langle G \rangle = \{0\} \sqcup \begin{cases} \bigsqcup_{P \in S_0} \mathcal{O}_P & \text{if } k \text{ is even,} \\ \left(\bigsqcup_{P \in S_0} \mathcal{O}_P\right) \sqcup \left(\bigsqcup_{P \in S_0} \mathcal{O}_{-P}\right) & \text{if } k \text{ is odd.} \end{cases} \tag{10}$$

To get a picture of where we are going, recall from §3 that in general, isogeny evaluation can be reduced to evaluations of the kernel polynomial

$$D(X) := \prod_{P \in S} (X - x(P)),$$

where $S \subset \langle G \rangle$ is any subset such that $S \cap -S = \emptyset$ and $S \cup -S = \langle G \rangle \setminus \{0\}$. The decomposition of (10) can be seen in the factorization of $D(X)$ over $\mathbb{F}_{q^k}$:

$$D(X) = \prod_{P \in S} (X - x(P)) = \prod_{P \in S_0} \prod_{i=0}^{k'-1} (X - x(\pi^i(P))) = \prod_{P \in S_0} \prod_{i=0}^{k'-1} (X - x(P)^{q^i}),$$

and the factors corresponding to each P in S_0 are the irreducible factors of D over $\mathbb{F}_q$. Transposing the order of the products, if we let

$$D_0(X) := \prod_{P \in S_0} (X - x(P))$$

then for α in the base field $\mathbb{F}_q$, we can compute $D(\alpha)$ using

$$D(\alpha) = \mathrm{Norm}(D_0(\alpha)) \quad \text{for all } \alpha \in \mathbb{F}_q$$

where

$$\mathrm{Norm}(x) := \prod_{i=0}^{k'-1} x^{q^i} = x(x(\cdots(x(x)^q)^q\cdots)^q)^q\,,$$

which can be computed for the cost of $(k-1)\mathbf{F}+(k-1)\mathbf{M}$ (some multiplications can be saved with more storage, but for small k this may not be worthwhile).

Similarly, we can rewrite the rational map ϕ_x from (6) as

$$\phi_x(x) = x\cdot\Big[\prod_{P\in S}\Big(\frac{x\cdot x(P)-1}{x-x(P)}\Big)\Big]^2 = x\cdot\Big[\prod_{P\in S_0}\prod_{i=0}^{k'-1}\Big(\frac{x\cdot x(P)^{q^i}-1}{x-x(P)^{q^i}}\Big)\Big]^2.$$

Evaluating ϕ_x at α in $\mathbb{F}_q$, rearranging the products gives

$$\phi_x(\alpha) = \alpha\cdot\Big[\prod_{P\in S_0}\prod_{i=0}^{k'-1}\Big(\frac{\alpha\cdot x(P)^{q^i}-1}{\alpha-x(P)^{q^i}}\Big)\Big]^2 = \alpha\cdot\mathrm{Norm}(\overline{\phi}_x(\alpha))^2\,,$$

where

$$\overline{\phi}_x(X) := \prod_{P\in S_0}\frac{X\cdot x(P)-1}{X-x(P)}\,.$$

Projectively, from (7) we get $\phi_x : (U:V)\mapsto(U':V')$ where

$$\begin{cases} U' = U\cdot\big[\prod_{i=0}^{k'-1}\prod_{P\in S_0}(UX_P^{q^i}-Z_P^{q^i}V)\big]^2,\\ V' = V\cdot\big[\prod_{i=0}^{k'-1}\prod_{P\in S_0}(UZ_P^{q^i}-X_P^{q^i}V)\big]^2,\end{cases}$$

so if we set

$$F(U,V) := \prod_{P\in S_0}(U\cdot X_P - Z_P\cdot V)\quad\text{and}\quad G(U,V) := \prod_{P\in S_0}(U\cdot Z_P - X_P\cdot V)\,,$$

then for α and β in $\mathbb{F}_q$ we get

$$\phi_x((\alpha:\beta)) = (\alpha':\beta') := \big(\alpha\cdot\mathrm{Norm}(F(\alpha,\beta))^2 : \beta\cdot\mathrm{Norm}(G(\alpha,\beta))^2\big)\,.$$

5.3 Enumerating representatives for the Galois orbits.

We now need to enumerate a set S_0 of representatives for the Galois orbits modulo ±1 or, equivalently, a set of representatives R_0 for the cosets of F_ℓ in M_ℓ. Given an MDAC driving enumeration of the coset representatives, there are obvious adaptations of Algorithms 1 and 2 to this extension field case. Rather than iterating over all of the kernel x-coordinates, we just iterate over a subset representing the cosets of $\mathbb{F}_\ell$, and then compose with the norm.

Concretely, in Algorithm 2, we should

1. Replace `KernelRange` in Line 4 with a generator driven by an efficient MDAC for M_ℓ/F_ℓ;
2. Replace Line 10 with $(U_i', V_i') \leftarrow (U_i\cdot\mathrm{Norm}(U_i')^2, V_i\cdot\mathrm{Norm}(V_i')^2)$.

First, we can consider Algorithm 3: that is, enumerating M_ℓ/F_ℓ by repeated addition. Unfortunately, we do not have a nice bound on the length of this MDAC: the coset representatives may not be conveniently distributed over M_ℓ, so we could end up computing a lot of redundant points.

Example 3. Consider the "naive" S_0 comprised of the minimal elements (up to negation) in each Galois orbit. We computed the percentage of primes $3 \leq \ell < 10^4$ where an optimal MDAC (without redundant values) exists for this S_0:

k	1	2	3	4	5	6	7	8	9	10	11	12
%	100	100	100	100	84	86	76	67	60	56	45	42

For an example of what can go wrong, take $(\ell, k) = (89, 11)$. In this case, we get $R_0 = \{1, 3, 5, 13\}$; the shortest MDAC is $(0, 1, 2, 3, 5, 8, 13)$, which enumerates the kernel using one `xDBL` operation and six `xADD` operations, but requires the computation of two intermediate points not used in the final result.

But when we say that the coset representatives are not conveniently distributed over M_ℓ, we mean convenient with respect to addition. If we look at M_ℓ multiplicatively, then the path to efficient MDACs is clearer.

If $M_\ell = \langle 2, \lambda \rangle$ then we can take $R_0 = \{2^i : 0 \leq i < c_F\}$, which brings us to the 2-powering MDAC of Example 1—except that we stop after $c_F - 1$ `xDBL`s. We thus reduce the number of `xDBL`s by a factor of $\approx k'$, at the expense of two norm computations. This MDAC actually applies to more primes ℓ here than in §4, because we no longer need 2 to generate all of M_ℓ; we have λ to help. (In fact, the suitability of this MDAC depends not only on ℓ, but also on k.)

We can go further if we assume

$$M_\ell = \langle 2, 3, \lambda \rangle \,. \tag{$**$}$$

To simplify notation, we define

$$a_{\ell,k} := [\langle 2, \lambda \rangle : F_\ell] \,, \qquad b_{\ell,k} := [\langle 2, 3, \lambda \rangle : \langle 2, \lambda \rangle] = c_F / a_{\ell,k} \,.$$

Algorithm 5 is a truncated version of Algorithm 4 for computing S_0 instead of $\mathcal{X}_G$ when $(**)$ holds. Algorithm 6 uses Algorithm 5 to evaluate an ℓ-isogeny over $\mathbb{F}_q$ with kernel $\langle G \rangle$ at n points of $\mathcal{E}(\mathbb{F}_q)$, where $x(G)$ is in $\mathbb{F}_{q^{k'}}$ with $k' > 1$.

Table 2 compares the total costs of Algorithms 6 and 5 with Algorithms 1 and 3. In both algorithms, we can take advantage of the fact that many of the multiplications have one operand in the smaller field $\mathbb{F}_q$: notably, the multiplications involving coordinates of the evaluation points. In the context of isogeny-based cryptography (where curve constants look like random elements of $\mathbb{F}_q$), this means that in Algorithm 1, we can replace the $2\mathbf{M} + 2\mathbf{a}$ in Line 8 and the $2\mathbf{M} + 2\mathbf{S}$ in Line 10 with $2\mathbf{C} + 2\mathbf{a}$ and $2\mathbf{C} + 2\mathbf{S}$, respectively. Table 3 gives examples of the resulting costs for various (ℓ, k) with a single evaluation point.

Algorithm 5: Compute S_0 when $(**)$ holds. Cost: $b_{\ell,k} - 1$ `xADD`s and $(c_F - b_{\ell,k})$ `xDBL`s, or $(2c_F + 2b_{\ell,k} + 4)\mathbf{M} + (2c_F - 2)\mathbf{S} + (c_F - b_{\ell,k})\mathbf{C} + (4c_F + 4b_{\ell,k} - 6)\mathbf{a}$

Input: Projective x-coordinate $(X_G : Z_G)$ of the generator G of a cyclic subgroup of order ℓ in $\mathcal{E}(\mathbb{F}_{q^k})$, where ℓ satisfies $M_\ell = \langle 2, 3, \lambda \rangle$.
Output: S_0 as a list

Function `SZeroPoints`$((X_G : Z_G))$
  $(a, b) \leftarrow (a_{\ell,k}, b_{\ell,k})$
  for $i = 0$ ***to*** $b - 1$ **do** `// Invariant:` $\mathsf{x}_{ai+j} = x([3^i 2^{i(a-2)+(j-1)}]G)$
    if $i = 0$ **then**
      $\mathsf{x}_1 \leftarrow (X_G : Z_G)$
    else `// Compute new coset representative`
      $\mathsf{x}_{ai+1} \leftarrow \mathtt{xADD}(\mathsf{x}_{ai}, \mathsf{x}_{ai-1}, \mathsf{x}_{ai-1})$
    for $j = 2$ ***to*** a **do** `// Exhaust coset by doubling`
      $\mathsf{x}_{ai+j} \leftarrow \mathtt{xDBL}(\mathsf{x}_{ai+j-1})$
  return $(\mathsf{x}_1, \ldots, \mathsf{x}_{c_F})$

Algorithm 6: Isogeny evaluation using `SZeroPoints` and Frobenius. Cost: $2(c_F + k - 1)n\mathbf{M} + 2n\mathbf{S} + 2(cF + 1)n\mathbf{C} + 2c_F(n+1)\mathbf{a} + 2(k-1)n\mathbf{F}$ *plus* the cost of `SZeroPoints`.

Input: The x-coordinate $(X_G : Z_G)$ of a generator G of the kernel of an ℓ-isogeny ϕ, and a list of evaluation points $((U_i : V_i) : 1 \le i \le n)$
Output: The list of images $((U_i' : V_i') = \phi_x((U_i : V_i)) : 1 \le i \le n)$

$((X_1, Z_1), \ldots, (X_{c_F}, Z_{c_F})) \leftarrow$ `SZeroPoints`$((X_G : Z_G))$ `// Algorithm 5`
for $1 \le i \le c_F$ **do**
  $(\hat{X}_i, \hat{Z}_i) \leftarrow (X_i + Z_i, X_i - Z_i)$ `// 2a`
for $i = 1$ ***to*** n **do**
  $(\hat{U}_i, \hat{V}_i) \leftarrow (U_i + V_i, U_i - V_i)$ `// 2a`
  $(U_i', V_i') \leftarrow (1, 1)$
  for $j = 1$ ***to*** c_F **do**
    $(t_0, t_1) \leftarrow$ `CrissCross`$(\hat{X}_j, \hat{Z}_j, \hat{U}_i, \hat{V}_i)$ `// 2C + 2a`
    $(U_i', V_i') \leftarrow (t_0 \cdot U_i', t_1 \cdot V_i')$ `// 2M`
  $(U_i', V_i') \leftarrow (\mathtt{Norm}(U_i'), \mathtt{Norm}(V_i'))$ `// 2(k' - 1)M + 2(k' - 1)F`
  $(U_i', V_i') \leftarrow (U_i \cdot (U_i')^2, V_i \cdot (V_i')^2)$ `// 2C + 2S`
return $((U_1', V_1'), \ldots, (U_n', V_n'))$

Table 2. ℓ-isogeny evaluation comparison for kernels $\langle G \rangle$ defined over $\mathbb{F}_q$ but with $x(G) \in \mathbb{F}_{q^{k'}}$. Here, **C** denotes multiplications of elements of $\mathbb{F}_{q^{k'}}$ by elements of $\mathbb{F}_q$.

	Costello–Hisil (Algorithms 1 and 3)	This work (Algorithm 6)
M	$(\ell-1)n + 2\ell - 8$	$2(c_F + k' - 1)n + 2c_F + 2b_{\ell,k} + 4$
S	$2n + \ell - 3$	$2n + 2c_F - 2$
C	$(\ell+1)n + 1$	$2(c_F + 1)n + c_F - b_{\ell,k}$
a	$(n+1)(\ell+1) + 3\ell + 17$	$2c_F(n+1) + 4c_F + 4b_{\ell,k} - 6$
F	0	$2(k'-1)n$

Table 3. Examples of costs for evaluating an ℓ-isogeny at a single point over $\mathbb{F}_q$, with $x(G) \in \mathbb{F}_{q^{k'}}$, using Costello–Hisil (Algorithm 1 with 3, in white) and Algorithm 6 (in gray). For these k, it is reasonable to use the approximation $\mathbf{F} \approx \mathbf{M}$ (see §5.1).

$\ell = 13$						$\ell = 19$						$\ell = 23$					
k'	**M**	**S**	**C**	**a**	**F**	k'	**M**	**S**	**C**	**a**	**F**	k'	**M**	**S**	**C**	**a**	**F**
any	30	12	15	54	0	any	48	18	21	84	0	any	60	22	25	104	0
1	22	12	19	46	0	1	34	18	28	70	0	1	42	22	34	86	0
3	10	4	7	14	4	3	14	6	10	22	4	11	22	2	4	6	20
						9	18	2	4	6	16						

6 Applications to key exchange

Our algorithms could be applied in any cryptosystem involving isogenies of prime degree $\ell > 3$. We focus on key exchanges like CSIDH [5] here, but similar remarks apply for other schemes such as SQISign [16,17], SeaSign [15], and CSI-FiSh [4].

6.1 CSIDH and constant-time considerations

CSIDH is a post-quantum non-interactive key exchange based on the action of the class group of the imaginary quadratic order $\mathbb{Z}[\sqrt{-p}]$ on the set of supersingular elliptic curves $\mathcal{E}/\mathbb{F}_p$ with $\mathrm{End}_{\mathbb{F}_p}(\mathcal{E}) \cong \mathbb{Z}[\sqrt{-p}]$. The action is computed via compositions of ℓ_i-isogenies for a range of small primes $(\ell_1, \ldots, \ell_m)$.

CSIDH works over prime fields $\mathbb{F}_p$, so the methods of §5 do not apply; but Algorithm 4 may speed up implementations at least for the ℓ_i satisfying $(*)$. (We saw in §4.6 that 67 of the 74 primes ℓ_i in the CSIDH-512 parameter set met $(*)$).

The true speedup depends on two factors. The first is the number of evaluation points. Costello and Hisil evaluate at a 2-torsion point other than $(0,0)$ in order to interpolate the image curve. The constant-time CSIDH of [19] evaluates at one more point (from which subsequent kernels are derived)—that is, $n = 2$; [21] uses $n = 3$; [7] discusses $n > 3$. For large n, the cost of Algorithm 1 overwhelms kernel enumeration, but our results may still make a simple and interesting improvement when n is relatively small.

The second factor is the organisation of primes into batches for constant-time CSIDH implementations. CTIDH [1] hides the degree ℓ using the so-called

matryoshka property: ℓ_i-isogeny evaluation is a sub-computation of ℓ_j-isogeny computation whenever $\ell_i < \ell_j$ using Algorithms 1 and 3. Organising primes into similar-sized batches, we can add dummy operations to disguise smaller-degree isogenies as isogenies of the largest degree in their batch.

Our Algorithm 4 has a limited matryoshka property: ℓ_i-isogenies are sub-computations of ℓ_j-isogenies if $a_{\ell_i} \leq a_{\ell_k}$ and $m_{\ell_i}/a_{\ell_i} \leq m_{\ell_j}/a_{\ell,j}$. For constant-time implementations, it would make more sense to make all primes in a batch satisfying $(*)$ a sub-computation of an algorithm using the maximum a_ℓ and maximum m_ℓ/a_ℓ over ℓ in the batch. Redistributing batches is a delicate matter with an important impact on efficiency; therefore, while our work improves the running time for a fixed ℓ, its impact on batched computations remains uncertain, and ultimately depends on specific parameter choices.

6.2 CRS key exchange

The historical predecessors of CSIDH, due to Couveignes [11] and Rostovtsev and Stolbunov [24,27,28], are collectively known as CRS. Here the class group of an quadratic imaginary order $\mathcal{O}$ acts on an isogeny (sub)class of elliptic curves $\mathcal{E}$ with $\mathrm{End}(\mathcal{E}) \cong \mathcal{O}$. CRS performance was greatly improved in [13] using Vélu-style isogeny evaluation, but this requires finding ordinary isogeny classes over $\mathbb{F}_p$ with rational ℓ_i-torsion points over $\mathbb{F}_{q^{k_i}}$ with k_i as small as possible for as many ℓ_i as possible.

One such isogeny class over a 512-bit prime field is proposed in [13, §4]. The curves ℓ-isogenies with kernel generators over $\mathbb{F}_p$ for $\ell = 3$, 5, 7, 11, 13, 17, 103, 523, and 821, and over $\mathbb{F}_{p^k}$ for $\ell = 19$, 29, 31, 37, 61, 71, 547, 661, 881, 1013, 1181, 1321, and 1693. These "irrational" ℓ are an interesting basis of comparison for our algorithms: Table 4 shows that there are substantial savings to be had.

References

1. Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. CTIDH: faster constant-time CSIDH. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4):351–387, 2021.
2. Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. In Steven D. Galbraith, editor, *Proceedings of the Fourteenth Algorithmic Number Theory Symposium*, pages 39–55. Mathematics Sciences Publishers, 2020. `https://eprint.iacr.org/2020/341`.
3. Daniel J. Bernstein, Tanja Lange, Chloe Martindale, and Lorenz Panny. Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 409–441. Springer, 2019.
4. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019*

Table 4. Costello–Hisil (Algorithms 1 and 3, in white) vs. Algorithm 6 (in gray) for the CRS parameters with $k > 1$ proposed in [13]. We omit $(\ell, k) = (1321, 5)$, since in this case $M_\ell \neq \langle 2, 3, \lambda \rangle$. Here **M**, **S**, **a**, and **F** refer to operations on elements of $\mathbb{F}_{q^{k'}}$, while **C** denotes multiplications of elements of $\mathbb{F}_{q^{k'}}$ by elements of $\mathbb{F}_q$.

k	ℓ	$a_{\ell,k}$	$b_{\ell,k}$	**M**	**S**	**C**	**a**	**F**
3	19	3	1	18n + 30	2n + 16	20n + 1	20n + 64	0
				10n + 4	2n + 4	8n + 2	8n + 14	4n
	661	110	1	660n + 1314	2n + 658	662n + 1	662n + 2632	0
				224n + 218	2n + 218	222n + 109	222n + 656	4n
4	1013	23	11	1012n + 2018	2n + 1010	1014n + 1	1014n + 4040	0
				48n + 524	2n + 504	48n + 242	48n + 1074	2n
	1181	59	5	1180n + 2354	2n + 1178	1182n + 1	1182n + 4712	0
				120n + 596	2n + 588	120n + 290	120n + 1302	2n
5	31	1	3	30n + 54	2n + 28	32n + 1	32n + 112	0
				10n + 8	2n + 4	4n	4n + 14	8n
	61	6	1	60n + 114	2n + 58	62n + 1	62n + 232	0
				20n + 10	2n + 10	14n + 5	14n + 32	8n
7	29	2	1	28n + 50	2n + 26	30n + 1	30n + 104	0
				16n + 2	2n + 2	6n + 1	6n + 8	12n
	71	5	1	70n + 134	2n + 68	72n + 1	72n + 272	0
				22n + 8	2n + 8	12n + 4	12n + 26	12n
	547	39	1	546n + 1086	2n + 544	548n + 1	548n + 2176	0
				90n + 76	2n + 76	80n + 38	80n + 230	12n
8	881	55	2	880n + 1754	2n + 878	882n + 1	882n + 3512	0
				116n + 220	2n + 218	112n + 108	112n + 548	6n
9	37	2	1	36n + 66	2n + 34	38n + 1	38n + 136	0
				20n + 2	2n + 2	6n + 1	6n + 8	16n
	1693	94	1	1692n + 3378	2n + 1690	1694n + 1	1694n + 6760	0
				204n + 186	2n + 186	190n + 93	190n + 560	16n

- *25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 227–247. Springer, 2019.
5. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Peyrin and Galbraith [22], pages 395–427.
6. Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 523–548. Springer, 2020.
7. Jesús-Javier Chi-Domínguez and Francisco Rodríguez-Henríquez. Optimal strategies for CSIDH. *Adv. Math. Commun.*, 16(2):383–411, 2022.
8. Jesús-Javier Chi-Domínguez, Andre Esser, Sabrina Kunzweiler, and Alexander May. Low memory attacks on small key CSIDH. In Mehdi Tibouchi and XiaoFeng Wang, editors, *Applied Cryptography and Network Security*, pages 276–304, Cham, 2023. Springer Nature Switzerland.
9. Craig Costello and Hüseyin Hisil. A simple and compact algorithm for SIDH with arbitrary degree isogenies. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *Lecture Notes in Computer Science*, pages 303–329. Springer, 2017.
10. Craig Costello and Benjamin Smith. Montgomery curves and their arithmetic. *Journal of Cryptographic Engineering*, Mar 2017.
11. Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Paper 2006/291, 2006. `https://eprint.iacr.org/2006/291`.
12. Daniele Cozzo and Nigel P. Smart. Sashimi: Cutting up csi-fish secret keys to produce an actively secure distributed signing protocol. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 169–186, Cham, 2020. Springer International Publishing.
13. Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In Peyrin and Galbraith [22], pages 365–394.
14. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *Public-Key Cryptography – PKC 2023*, pages 345–375, Cham, 2023. Springer Nature Switzerland.
15. Luca De Feo and Steven D. Galbraith. Seasign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III*, volume 11478 of *Lecture Notes in Computer Science*, pages 759–789. Springer, 2019.
16. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 64–93. Springer, 2020.

17. Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the Deuring correspondence - towards practical and secure SQISign signatures. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 659–690. Springer, 2023.
18. David R. Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California at Berkeley, 1996. `http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf`.
19. Michael Meyer, Fabio Campos, and Steffen Reith. On lions and elligators: An efficient constant-time implementation of CSIDH. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019, Chongqing, China, May 8-10, 2019 Revised Selected Papers*, volume 11505 of *Lecture Notes in Computer Science*, pages 307–325. Springer, 2019.
20. Peter L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.
21. Hiroshi Onuki, Yusuke Aikawa, Tsutomu Yamazaki, and Tsuyoshi Takagi. A constant-time algorithm of CSIDH keeping two points. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 103-A(10):1174–1182, 2020.
22. Thomas Peyrin and Steven Galbraith, editors. *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
23. Joost Renes. Computing Isogenies Between Montgomery Curves Using the Action of $(0, 0)$. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography*, pages 229–247, Cham, 2018. Springer International Publishing.
24. Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Paper 2006/145, 2006. `https://eprint.iacr.org/2006/145`.
25. Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Springer-Verlag, New York, NY, 2. aufl. edition, 2009.
26. Benjamin Smith. Pre- and post-quantum Diffie-Hellman from groups, actions, and isogenies. In Lilya Budaghyan and Francisco Rodríguez-Henríquez, editors, *Arithmetic of Finite Fields - 7th International Workshop, WAIFI 2018, Bergen, Norway, June 14-16, 2018, Revised Selected Papers*, volume 11321 of *Lecture Notes in Computer Science*, pages 3–40. Springer, 2018.
27. Anton Stolbunov. Reductionist security arguments for public-key cryptographic schemes based on group action. *Norsk informasjonssikkerhetskonferanse (NISK)*, pages 97–109, 2009.
28. Anton Stolbunov. Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. *Advances in Mathematics of Communications*, 4(2):215–235, 2010.
29. Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes rendus hebdomadaires des séances de l'Académie des sciences, Série A*, 273:238–241, 7 1971.
30. Samuel S. Wagstaff, Jr. Pseudoprimes and a generalization of Artin's conjecture. *Acta Arithmetica*, 41:141–150, 1982.

A Computing a kernel generator

One task that poses a challenge is to find a point $G \in \mathcal{E}(\mathbb{F}_{q^k})$. In this section, we will illustrate an efficient method for computing a point with the necessary properties for use in the isogeny evaluation.

A.1 The subgroup H_k.

To compute a rational isogeny, our first step will be to sample a random point $P \in \mathcal{E}(\mathbb{F}_{q^k})$ of order ℓ. For this, letting $N_k := \#\mathcal{E}(\mathbb{F}_{q^k})$, one could sample a random point P, and compute $P_\ell = [N_k/\ell]P$. Then P_ℓ is either 0 or a point of order ℓ. If the order of P_ℓ is not ℓ, one tries again with a new choice of P.

Remark 1. In the special case that ℓ^2 divides $\#\mathcal{E}(\mathbb{F}_{q^k})$, we instead choose $N_k = exp(\mathbb{F}_{q^k})$, the exponent of the group order. We do this to avoid having a cofactor, N_k/ℓ, that "kills" certain torsion points.

In our context, we are assured that the P_ℓ we are looking for is *not* in $\mathcal{E}(\mathbb{F}_q)$, or indeed in any $\mathcal{E}(\mathbb{F}_{q^i})$, for any proper divisor i of k. We can therefore save some effort by sampling P_ℓ from the genuinely "new" subgroup of $\mathcal{E}(\mathbb{F}_{q^k})$.

Recall that $\mathcal{E}(\mathbb{F}_{q^i}) = \ker(\pi^i - [1])$ for each $i > 0$. For each $k > 0$, then we define an endomorphism

$$\eta_k := \Phi_k(\pi) \in \mathrm{End}(\mathcal{E})$$

where $\Phi_k(X)$ is the k-th cyclotomic polynomial (that is, the minimal polynomial over $\mathbb{Z}$ of the primitive k-th roots of unity in $\overline{\mathbb{F}}$). The subgroup

$$H_k := \ker(\eta_k) \subset \mathcal{E}(\mathbb{F}_{q^k})$$

satisfies

$$\mathcal{E}(\mathbb{F}_{q^k}) = H_k \oplus \sum_{i|k, i\neq k} \mathcal{E}(\mathbb{F}_{q^i}).$$

The key fact is that in our situation, $\mathcal{E}[\ell](\mathbb{F}_{q^k}) \subset H_k$.

Generating elements of $\mathcal{E}[\ell](\mathbb{F}_{q^k})$. We always have $\Phi_k(X) \mid X^k - 1$, so for each $k > 0$ there is an endomorphism

$$\delta_k := (\pi^k - [1])/\eta_k \in \mathrm{End}(\mathcal{E})\,,$$

and $\delta_k(\mathcal{E}(\mathbb{F}_{q^k})) \subset H_k$. We can therefore sample a point P_ℓ in $\mathcal{E}[\ell](\mathbb{F}_{q^k})$ by computing

$$P_\ell = [h_k/\ell]\delta_k(P) \qquad \text{where} \qquad h_k := \#H_k$$

and P is randomly sampled from $\mathcal{E}(\mathbb{F}_{q^k})$.

Table 5 lists the first few values of h_k and δ_k. We see that evaluating δ_k amounts to a few Frobenius operations (which are almost free, depending on the field representation) and a few applications of the group law, so this approach

saves us a factor of at least $1/k$ in the loop length of the scalar multiplication (compared wih computing P_ℓ as $[N_k/\ell]P$), but for highly composite k we save much more.

The value $\varphi(k)$ of the Euler totient function plays an important role. We have $h_k = q^{\varphi(k)} + o(q^{\varphi(k)})$, so computing $[h_k/\ell]$ instead of $[N_k/\ell]$ allows us to reduce the loop length of basic scalar multiplication algorithms from $k \log_2 q$ to $\varphi(k) \log_2 q$, which is particularly advantageous when k is highly composite.

The action of Frobenius on H_k. The Frobenius endomorphism π commutes with η_k, and therefore restricts to an endomorphism of H_k. If $G \subset H_k$ is a subgroup of prime order ℓ and fixed by π, then π will act on G as multiplication by an integer eigenvalue λ (defined modulo ℓ). Since $\eta_k = \Phi_k(\pi) = 0$ on H_k by definition, we know that λ is a k-th root of unity in $\mathbb{Z}/\ell\mathbb{Z}$.

Scalar multiplication with Frobenius. Now, $H_k \cong \mathbb{Z}/d_k\mathbb{Z} \times \mathbb{Z}/e_k\mathbb{Z}$, where $d_k \mid e_k$ and (by the rationality of the Weil pairing) $d_k \mid q^k - 1$. Typically, d_k is very small compared with e_k. If $\ell \nmid d_k$, then we can replace H_k with the cyclic subgroup $H'_k := [d_k]H_k$, and h_k with $h'_k := e_k/d_k$. Now, π induces an endomorphism of H'_k, and therefore acts as multiplication by an eigenvalue λ defined modulo h'_k.

We want to compute $[c_k]P$ for P in H'_k, where $c_k := h'_k/\ell$. Since $\Phi_k(\pi) = 0$, the eigenvalue λ is a root of Φ_k (i.e., a primitive k-th root of unity) modulo h'_k. We can compute $a_0, \ldots, a_{k-1}$ such that

$$c_k \equiv \sum_{i=0}^{k-1} a_i \lambda^i \pmod{h'_k},$$

with each coefficient $a_i \approx (h'_k)^{1/\varphi(k)}$ in $O(q)$, and then

$$[c_k]P = \sum_{i=0}^{k-1} [a_i]\pi^i(P).$$

If we precompute the various sums of conjugates of P,then we can compute $[c_k]$ using a multiscalar multiplication algorithm with a loop of length only $\log_2 q$. This might be particularly interesting in the cases where $\varphi(k) = 2$ (which corresponds to GLV multiplication) or 4.

Example 4. Consider $k = 3$: we have $\mathcal{E}(\mathbb{F}_{q^3}) \cong \mathcal{E}(\mathbb{F}_q) \oplus H_3$, and $\#H_3 = N_3/N_1$.

We first note that π^3 fixes the points in $\mathcal{E}(\mathbb{F}_{q^3})$, so $\pi^3 - [1] = [0]$ on $\mathcal{E}(\mathbb{F}_{q^3})$. By similar logic, the regular Frobenius map will fix the points in $\mathcal{E}(\mathbb{F}_q)$, meaning $\pi - [1] = [0]$ holds only for points contained entirely in the $\mathcal{E}(\mathbb{F}_q)$ portion of $\mathcal{E}(\mathbb{F}_{q^3})$. Therefore, by computing $P_H = (\pi - 1)P$, we are "killing" the $\mathcal{E}(\mathbb{F}_q)$ part of P, leaving only the part lying in the subgroup H_3. This computation is easy enough to do, and so now we need only compute $P_\ell = [N_3/N_1/\ell]P_H$, thereby saving us about a third of the multiplications.

Table 5. The first few values of h_k and δ_k.

k	h_k	δ_k	$\varphi(k)$
1	1	$[1]$	
2	$N_2/N_1 = q + O(\sqrt{q})$	$\pi - [1]$	1
3	$N_3/N_1 = q^2 + O(q^{3/2})$	$\pi - [1]$	2
4	$N_4/N_2 = q^2 + O(q)$	$\pi^2 - [1]$	2
5	$N_5/N_1 = q^4 + O(q^{7/2})$	$\pi - [1]$	4
6	$(N_6N_1)/(N_2N_3) = q^2 + O(q^{3/2})$	$(\pi + [1])(\pi^3 - [1])$	2
7	$N_7/N_1 = q^6 + O(q^{7/2})$	$\pi - [1]$	6
8	$N_8/N_4 = q^4 + O(q^2)$	$\pi^4 - [1]$	4
9	$N_9/N_3 = q^6 + O(q^{4/2})$	$\pi^3 - [1]$	6
10	$(N_{10}N_1)/(N_2N_5) = q^4 + O(q^{7/2})$	$(\pi + [1])(\pi^5 - [1])$	4
11	$N_{11}/N_1 = q^{10} + O(q^{19/2})$	$\pi - [1]$	10
12	$(N_{12}N_2)/(N_4N_6) = q^4 + O(q^3)$	$(\pi^2 + [1])(\pi^6 - [1])$	4

The "twist trick". When k is even, if we use x-only scalar multiplication, then the following lemma allows us to work over $\mathbb{F}_{q^{k/2}}$ instead of $\mathbb{F}_{q^k}$. In the case $k = 2$, this is known as the "twist trick".

Lemma 1. *If k is even, then every point P in H_k has $x(P)$ in $\mathbb{F}_{q^{k/2}}$.*

Proof. If k is even, then η_k divides $\pi^{k/2}+1$, so $\pi^{k/2}$ acts as -1 on $H_k = \ker(\eta_k)$: that is, if P is in H_k, then $\pi^{k/2}(P) = -P$, so $x(P)$ is in $\mathbb{F}_{q^{k/2}}$.

Example 5. Consider $k = 6$. We take a random point R in $E(\mathbb{F}_{q^6})$, and compute $R' := \pi^3(R) - R$, then $P := \pi(R') + R'$; now $P = \delta_6(R)$ is in H_6, and $x(P)$ is in $\mathbb{F}_{q^3}$. We have $h_6 = N_1^2 - (q+1)N_1 + q^2 - q + 1 \approx q^2$, and we need to compute $x([c_{\ell,6}]P)$ where $c_{\ell,6} := h_6/\ell$. Since $x(P)$ is in $\mathbb{F}_{q^3}$, we can do this using x-only arithmetic and the Montgomery ladder working entirely over $\mathbb{F}_{q^3}$.

The improvements outlined in this section are summarized in Algorithm 7.

Algorithm 7: Computation of Kernel Generator.

Input: $\mathcal{E}$ an elliptic curve defined over $\mathbb{F}_q$, ℓ an integer, k such that $\mathbb{F}_{q^k}$ contains an ℓ-torsion point.
Output: P, a point on $\mathcal{E}(\mathbb{F}_{q^k})$ of order ℓ.

$P_\ell \leftarrow (0:1:0)$
repeat
  $P \leftarrow$ `RandomPoint`$(\mathcal{E}(\mathbb{F}_{q^k}))$
  $c \leftarrow h_k/\ell$ `// `h_k` taken from Table 5.`
  $P' \leftarrow \delta_k(P)$ `// `δ_k` taken from Table 5.`
  $P_\ell \leftarrow [c]P'$
until $P_\ell = (0:1:0)$
return P_ℓ

9 Supersingularity testing

Publication data. Gustavo Banegas, Valerie Gilchrist, Benjamin Smith. Efficient supersingularity testing over $\mathbb{F}_p$ and CSIDH key validation. MathCrypt 2022.

My Contribution. In this work all three coauthors contributed equally in the theoretical development and writing.

Efficient supersingularity testing over $\mathbb{F}_p$ and CSIDH key validation

Gustavo Banegas[1], Valerie Gilchrist[2,1,*], Benjamin Smith[1]

[1]Inria and Laboratoire d'Informatique de l'École polytechnique, Institut Polytechnique de Paris, Palaiseau, France
[2]University of Waterloo, Canada

Abstract *Many public-key cryptographic protocols, notably non-interactive key exchange (NIKE), require incoming public keys to be validated to mitigate some adaptive attacks. In CSIDH, an isogeny-based post-quantum NIKE, a key is deemed legitimate if the given Montgomery coefficient specifies a supersingular elliptic curve over the prime field. In this work, we survey the current supersingularity tests used for CSIDH key validation, and implement and measure two new alternative algorithms. Our implementation shows that we can determine supersingularity substantially faster, and using less memory, than the state-of-the-art.*

Keywords: Isogenies, Key validation, Supersingularity, Elliptic Curves

1 INTRODUCTION

The security of many public-key cryptosystems assumes that public keys are honestly generated: that is, that public keys have not been manipulated by adversaries. *Key validation* is the process of determining whether an incoming public key was plausibly constructed following the protocol.

The simplest example of this is in non-interactive key exchange (NIKE). Consider classic static Diffie–Hellman in a finite field: the system parameters fix a prime modulus p and a generator g in $\mathbb{F}_p$ for a subgroup $G = \langle g \rangle \subset \mathbb{F}_p^\times$ of prime order r. Alice samples a long-term secret integer a, and binds to the corresponding public key $A = g^a$. An honest Bob computes his keypair $(B = g^b, b)$, and the shared secret is $S = A^b = B^a$. However, if G is a proper subgroup of $\mathbb{F}_p^\times$, then a dishonest Bob can choose some h in $\mathbb{F}_p^\times \setminus G$, of order $s \mid (p-1)/r$, and transmit the malformed public key $B' = B \cdot h$. The shared secret as computed by Alice is now $S' = (B')^a = S \cdot h^a$, while Bob derives the original $S = A^b$. The success or failure of subsequent encrypted communication tells Bob whether $S = S'$, and hence whether a is $0 \pmod{s}$.

To avoid leaking information on her long-term private key to adaptive adversaries, then, Alice must validate incoming public keys as being honestly generated. In the example above, this amounts to checking that Bob's public key really is an element of G; this can be done by checking that $B^r = 1$. In elliptic-curve Diffie–Hellman key exchange, key validation amounts to an analogous scalar multiplication by a (tiny or trivial) cofactor, *plus* a simple check that Bob's public key B really does encode a point on the curve.

Moving now to the post-quantum setting, the best-established NIKE candidate is CSIDH [4], a key exchange scheme based on the action of the ideal class group of $\mathbb{Z}[\sqrt{-p}]$ on the isogeny class of supersingular elliptic curves over $\mathbb{F}_p$ (recall that $\mathcal{E}/\mathbb{F}_p$ is *supersingular* if $\#\mathcal{E}(\mathbb{F}_p) = p + 1$; otherwise, it is *ordinary*). The CSIDH group action has also been used to construct other post-quantum public-key cryptosystems, including signatures [2, 8, 9], threshold schemes [10], and oblivious transfer [16].

Validating CSIDH public keys is therefore important for long-term post-quantum security. The fundamental problem is: we are given an element A in $\mathbb{F}_p$ corresponding to an elliptic curve with a *Montgomery model*

$$\mathcal{E}_A/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)\,, \tag{1}$$

and we must determine if $\mathcal{E}_A$ is in the orbit of the group action. By [4, Proposition 8], to validate A it suffices to

1. check that $A^2 \neq 4$ (that is, $\mathcal{E}_A$ is an elliptic curve and not a singular cubic), a trivial task; and
2. check that $\mathcal{E}_A$ is supersingular, a nontrivial and mathematically interesting task.

Key validation is not the bottleneck in CSIDH key exchange—it typically takes under 5% of the runtime—but it is still a critical problem to be solved efficiently. The main issue is supersingularity testing over $\mathbb{F}_p$, and we will focus entirely on this problem, ignoring the rest of the CSIDH cryptosystem and its derivatives (we refer the reader to [4, 5, 1] for further details and discussion). The only relevant detail is that in CSIDH,

$$p = 4 \prod_{i=1}^{n} \ell_i - 1$$

*Corresponding Author: Valerie Gilchrist, vgilchrist@uwaterloo.ca

*Author list in alphabetical order; see https://www.ams.org/profession/leaders/CultureStatement04.pdf. This research was funded in part by the French Agence Nationale de la Recherche through ANR CIAO (ANR-19-CE48-0008) and the European Commission through H2020 SPARTA, https://www.sparta.eu/. Date of this document: 2025-03-18.

where the ℓ_i are small primes: this ensures that supersingular curves over $\mathbb{F}_p$ have rational points of order ℓ_i for $1 \le i \le n$, which can be used to compute the group action for ideals of smooth norm in a particularly efficient way. In particular, $p \equiv 3 \pmod 4$.

In this work we consider four algorithms for testing supersingularity over $\mathbb{F}_p$ in the context of CSIDH key validation. The algorithms are listed in Table 1. We have implemented and benchmarked each of them for the 512-bit prime p defined by the CSIDH-512 parameter set; our practical results are also summarized in Table 1.

Table 1: Supersingularity testing algorithms for elliptic curves over $\mathbb{F}_p$. For Algorithms 2 and 3, $p + 1 = 4 \prod_{i=1}^{n} \ell_i$. (The experimental setup for the performance measurements with CSIDH-512 parameters is detailed below.)

Test algorithm	Asymptotics		Supersingular input (valid keys)			Non-supersingular input (invalid keys)		
	Time ($\mathbb{F}_p$-ops)	Space ($\mathbb{F}_p$-elts)	MCycles: Mean	Median	Stack (B)	MCycles: Mean	Median	Stack (B)
Alg. 2 (Random point)	$O(n \log p)$	$O(1)$	63.4	62.2	2890	65.3	62.9	2890
Alg. 3 (Product tree)	$O((\log n)(\log p))$	$O(\log n)$	6.7	6.1	4344	1.7	1.6	3896
Alg. 5 (Sutherland)	$O(\log^2 p)$	$O(1)$	35.4	35.1	2696	0.8	0.4	2696
Alg. 6 (Doliskani)	$O(\log p)$	$O(1)$	4.5	4.7	3280	2.9	2.8	3264

Algorithms 2 and 3, detailed in 3, are elementary algorithms based on determining the order of a random point on $\mathcal{E}_A$. These are the algorithms used in existing CSIDH implementations, and we include them as a baseline for comparison. These are the only supersingularity tests here that rely on the special form of CSIDH primes (they require the factorization of $p + 1$).

Algorithm 5, described in 4, is our variant of Sutherland's test [22] for curves over $\mathbb{F}_{p^2}$, adapted and optimized for Montgomery models over $\mathbb{F}_p$. This test is based on distinguishing between the 2-isogeny graph structures of supersingular and ordinary elliptic curves over $\mathbb{F}_{p^2}$. While the asymptotic complexity of this algorithm is quadratic in $\log p$ (the others are linear or quasi-linear), it has very good constants, and we find that it performs surprisingly well in practice. It is extremely easy to implement, and requires very little memory. It is also the only one of the four tests considered here that always produces definitive proof of supersingularity.

Algorithm 6, described in 5, is our variant of Doliskani's test [12] for curves over $\mathbb{F}_{p^2}$, which uses Polynomial Identity Testing to distinguish between the p-th division polynomials of supersingular and ordinary curves. We have adapted and optimized this algorithm for Montgomery models over $\mathbb{F}_p$, drastically simplifying the division polynomial computation. This algorithm is particularly simple: it only requires a single scalar multiplication of a point over $\mathbb{F}_{p^2}$, followed by an easy field exponentiation. Algorithm 6 is a Monte Carlo algorithm with one-sided error: it may declare an ordinary curve supersingular, but the probability of this is in $O(1/p)$, which is virtually zero for cryptographic p: for the 512-bit p of CSIDH, a false-accept rate of 2^{-512} is more than covered by the claimed security level.

We will see that Algorithm 6 is the simplest and fastest supersingularity test for CSIDH public key validation.

Implementation and experimental results We implemented our algorithms in C for the 512-bit prime p of the CSIDH-512 parameter set, which was built from $n = 74$ small primes ℓ_i. For $\mathbb{F}_p$-arithmetic, we used the assembly code from the CTIDH library [1][1]. For $\mathbb{F}_{p^2}$-arithmetic, we used the "tricks" from [20], which we reproduce for easy reference in Appendix A. The implementation of Algorithm 3 was taken directly from the CTIDH library. Algorithms 2, 5, and 6 are our own implementations.

We ran our experiments on an Intel i7-10610U processor running at 4.90 GHz with TurboBoost and SpeedStep disabled, running Arch Linux with kernel 5.15.41-1-lts and GCC 12.1.0. Cycles were measured using the `bench` utility provided in the CSIDH code package. For our experiments, we generated 500 valid (supersingular) curves and 500 invalid (ordinary) curves. Table 1 presents the average and median runtime for each algorithm (in millions of cycles), and the maximum stack use (in bytes) for a complete run of the algorithm (including subroutines). We note that the algorithm used to compute square roots in $\mathbb{F}_{p^2}$ inside Algorithm 5 uses several temporary variables, which increases the stack footprint; this might be further optimized.

Elliptic curve models We focus on elliptic curves with Montgomery models, which are ubiquitous in isogeny-based cryptography. However, we discuss extensions of our methods to more general elliptic curves over $\mathbb{F}_p$ in 6.

Notation Throughout, p denotes a (large) odd prime, and q is a power of p. For every integer $m > 0$, we write

$$\mathrm{len}(m) := \lfloor \log_2 m \rfloor + 1 \qquad \text{(i.e., the bitlength of } m\text{)}.$$

[1]We used version `20210523`, available from `http://ctidh.isogeny.org/software.html`.

2 MONTGOMERY ARITHMETIC

We assume that the reader is familiar with basic elliptic curve arithmetic (see e.g. [21] for background). However, Algorithm 6 requires some fine detail on the Montgomery ladder algorithm [18], which is also a subroutine of Algorithms 2, 3, and CSIDH itself, so we take a moment to recall it here. (For further detail, see [7].)

In practice, most isogeny-based cryptosystems (including CSIDH) work with *Montgomery models*

$$\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)\,.$$

We call the parameter A the *Montgomery coefficient*. We write $\oplus$ and $\ominus$ for addition and subtraction on $\mathcal{E}_A$.

Let P and Q be points on $\mathcal{E}_A$. Any three of $x(P)$, $x(Q)$, $x(P \ominus Q)$, and $x(P \oplus Q)$ determines the fourth, so we can define a *differential addition*

$$\texttt{xADD} : (x(P), x(Q), x(P \ominus Q)) \longmapsto x(P \oplus Q)$$

and a pseudo-doubling operation

$$\texttt{xDBL}_A : x(P) \longmapsto x([2]P)\,.$$

We can evaluate these maps on affine representatives for projective points as follows. Given points P and Q on $\mathcal{E}_A$, we write $(X_P : Z_P) = x(P)$, $(X_Q : Z_Q) = x(Q)$, $(X_\oplus : Z_\oplus) = x(P \oplus Q)$, and $(X_\ominus : Z_\ominus) = x(P \ominus Q)$ (recall that $x((X : Y : Z)) = (X : Z)$ if $Z \neq 0$, and $(1 : 0)$ if $Z = 0$). Now we can compute $\texttt{xADD}$ using the formulæ

$$\begin{cases} X_\oplus = Z_\ominus \cdot [U + V]^2\,, \\ Z_\oplus = X_\ominus \cdot [U - V]^2 \end{cases} \qquad \text{where} \qquad \begin{cases} U = (X_P - Z_P)(X_Q + Z_Q)\,, \\ V = (X_P + Z_P)(X_Q - Z_Q)\,. \end{cases} \tag{2}$$

Note that if we replace (X_P, Z_P) and (X_Q, Z_Q) with the projectively equivalent $(\lambda_P X_P, \lambda_P Z_P)$ and $(\lambda_Q X_Q, \lambda_Q Z_Q)$ in (2), then $(X_\oplus, Z_\oplus)$ becomes $((\lambda_P\lambda_Q)^2 X_\oplus, (\lambda_P\lambda_Q)^2 Z_\oplus)$.

Similarly, writing $(X_{[2]P} : Y_{[2]P} : Z_{[2]P})$ for $[2]P$, we can compute $\texttt{xDBL}_A$ using

$$\begin{cases} X_{[2]P} = R \cdot S\,, \\ Z_{[2]P} = T \cdot (S + \frac{A+2}{4} T) \end{cases} \qquad \text{where} \qquad \begin{cases} R = (X_P + Z_P)^2\,, \\ S = (X_P - Z_P)^2\,, \\ T = 4X_P \cdot Z_P = Q - R\,. \end{cases} \tag{3}$$

If we replace (X_P, Z_P) with $(\lambda_P X_P, \lambda_P Z_P)$ in (3), then $(X_{[2]P}, Z_{[2]P})$ becomes $(\lambda_P^4 X_{[2]P}, \lambda_P^4 Z_{[2]P})$.

Algorithm 1 is the *Montgomery ladder*, which efficiently computes the map $(m, x(P)) \mapsto x([m]P)$.

Algorithm 1: The Montgomery ladder on the x-line $\mathbb{P}^1$ under $\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)$

Input: $A \in \mathbb{F}_q$, $m = \sum_{i=0}^{\beta-1} m_i 2^i$, and $(X_P, Z_P) \in \mathbb{F}_q^2$ with $X_P Z_P \neq 0$
Output: (X_m, Z_m) and (X_{m+1}, Z_{m+1}) in $\mathbb{F}_p(u)^2$ such that $(X_m : Z_m) = x([m]P)$ and $(X_{m+1} : Z_{m+1}) = x([m+1]P)$ where P is a point on $\mathcal{E}_A$ with $x(P) = (X_P : Z_P)$

Function `Ladder`$(A, m, (X_P, Z_P))$
  $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow ((1,0), (X_P, Z_P))$ // $(1:0) = x(0)$ and $(X_P : Z_P) = x(P)$
  for i *in* $(\beta - 1, \ldots, 0)$ **do** // Invariant: $\mathsf{R}_0 = x([\lfloor m/2^i \rfloor]P)$ and $\mathsf{R}_1 = x([\lfloor m/2^i \rfloor + 1]P)$
    if $m_i = 0$ **then**
      $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow (\texttt{xDBL}_A(\mathsf{R}_0), \texttt{xADD}(\mathsf{R}_0, \mathsf{R}_1, (X_P, Z_P)))$
    else
      $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow (\texttt{xADD}(\mathsf{R}_0, \mathsf{R}_1, (X_P, Z_P)), \texttt{xDBL}_A(R_1))$
  return R_0 *and optionally* R_1 // $\mathsf{R}_0 = x([m]P)$ and $\mathsf{R}_1 = x([m+1]P)$

3 ELEMENTARY SUPERSINGULARITY TESTS

We begin by considering the two supersingularity tests that have been proposed for use with CSIDH [4]. These elementary tests are included as a point of reference when comparing performance with our new algorithms below; for detailed discussion, see [4, 5 and 8].

The goal of these two tests is to try to exhibit a point of order $N \mid (p+1)$ with $N > 4\sqrt{p}$; then, the only multiple of N in the Hasse interval is $p+1$, so we can conclude that the curve has order $p+1$ and is therefore supersingular.

On the other hand, if we find a point whose order can be shown to *not* divide $p+1$, then we can immediately declare the curve ordinary. To efficiently determine (a divisor of) the order, we need to know the factorization of $p+1$, which in the case of CSIDH is $4\prod_{i=1}^{n}\ell_i$ with the ℓ_i very small.

Algorithm 2 proceeds in the simplest way: let u be a random element of $\mathbb{F}_p \setminus \{0\}$. Now u is the x-coordinate of a point P in $\mathcal{E}(\mathbb{F}_{p^2})$, which has exponent $p+1$. For each of the primes ℓ_i, we compute $Q_i = [(p+1)/\ell_i]P$.

- If $Q_i = 0$, then we learn nothing from ℓ_i;
- if $Q_i \neq 0$ but $[\ell_i]Q_i = 0$ then we know that ℓ_i divides the order of P;
- if $[\ell_i]Q_i \neq 0$ then the order of P cannot divide $p+1$, so we know the curve is ordinary.

Once we have accumulated enough ℓ_i that $\prod_i \ell_i > 4\sqrt{p}$, we can stop and declare the curve supersingular.

Algorithm 2 is quite simple to follow and has low memory requirements, but its expected runtime is rather slow. Indeed, each ℓ_i that we treat entails two `Ladder` calls, with $m = (p+1)/\ell_i$ and ℓ_i, and this adds up to approximately the cost of a `Ladder` with $m = p+1$. The asymptotic runtime is therefore $O(n \log p)$ $\mathbb{F}_p$-operations.

For a more concrete perspective: to minimise the total runtime before exceeding $4\sqrt{p}$ (or declaring ordinariness), we treat the ℓ_i from largest to smallest. If the curve is supersingular (which is the worst case for runtime), then we will need a little under half of the ℓ_i; that is, we expect an effort equivalent of around $n/2$ full-length `Ladder` calls over $\mathbb{F}_p$. Of course, in practice we can compute these point multiplications using precalculated optimal differential addition chains, rather than the ladder, but this is only a small improvement.

It is possible, though *extremely* improbable for cryptographic-size p, for Algorithm 2 to fail and return $\perp$. (This would imply that the random point has very small order.) If this happens, then we can simply re-run Algorithm 2 with a new random u.

Algorithm 2: Supersingularity testing for $\mathcal{E}/\mathbb{F}_p : y^2 = x(x^2+Ax+1)$ via random point multiplication [4, Algorithm 1]. Assumes the factorization $p+1 = 4\prod_{i=1}^{n}\ell_i$ is known, with $\ell_n > \cdots > \ell_1$.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False** or $\perp$

```
1  Function IsSupersingular(A)
2      u ← Random(F_p \ {0})
3      N ← 1
4      for i in (n, ..., 1) do
5          (X, Z) ← Ladder(A, (p+1)/ℓ_i, (u, 1))
6          (X', Z') ← Ladder(A, ℓ_i, (X, Z))
7          if Z' ≠ 0 then
8              return False
9          if Z ≠ 0 then
10             N ← N · ℓ_i
11         if N > 4√p then
12             return True
13     return ⊥
```

Algorithm 3 is a simple version of Algorithm 2 exploiting the fact that the various scalar multiplications are products of the same small primes, so we can compute them more efficiently using a classic product-tree structure. The algorithm proposed in [4, 8] traverses the product tree breadth-first. Algorithm 3, which is essentially the algorithm currently used in the CSIDH and CTIDH reference implementations, does this depth-first instead, handling the leaves corresponding to the largest ℓ_i first (the depth-first approach saves a lot of memory, and a little time too).

The product-tree approach is essentially a space-time tradeoff: we mutualise much of the effort of the scalar multiplications in the basic Algorithm 2, at the cost of storing the internal nodes of the product tree on the path to the current leaf. The depth of the tree is $\lceil \log_2 n \rceil$, so we have an asymptotic time complexity of $O(k \log n)$ $\mathbb{F}_p$-operations and a space complexity of $O(\log n)$ $\mathbb{F}_p$-elements.

Remark 1. *We warn the reader that the tests in this section do* not *work, or generalise, for elliptic curves over $\mathbb{F}_{p^2}$: supersingular curves over $\mathbb{F}_{p^2}$ do not have points of sufficiently large order to conclude on the group order.*

Algorithm 3: Supersingularity testing for $\mathcal{E}/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)$ via random point multiplication with an implicit product tree. Assumes the factorization $p + 1 = 4\prod_{i=1}^{n} \ell_i$ is known.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**
Function `IsSupersingular(`A`)`
  Function `OrderRec(`A, (X_Q, Z_Q), L, U, m`)`
    if $U - L = 1$ **then** `// At this point,` $Q = [(p+1)/\ell_U]P$
      if $Z_Q = 0$ **then** `// In this case, we learn nothing`
        return m
      $(X_Q, Z_Q) \leftarrow$ `Ladder(`$A, \ell_U, (X_Q, Z_Q)$`)`
      if $Z_Q = 0$ **then** `//` ℓ_U `divides the order of` P
        return $\ell_U \cdot m$
      else `//` $[p+1]P \neq 0$`: not supersingular!`
        return 0
    $(X_L, Z_L) \leftarrow$ `Ladder(`$A, \prod_{i=L+1}^{\lfloor (U+L)/2 \rfloor} \ell_i, (X_Q, Z_Q)$`)`
    $m \leftarrow$ `OrderRec(`$A, (X_L, Z_L), \lfloor (U+L)/2 \rfloor, U, m$`)`
    if $m > 4\sqrt{p}$ **then**
      return m
    $(X_R, Z_R) \leftarrow$ `Ladder(`$A, \prod_{i=\lfloor (U+L)/2+1 \rfloor}^{R} \ell_i, (X_Q, Z_Q)$`)`
    $m \leftarrow$ `OrderRec(`$A, (X_R, Z_R), L, \lfloor (U+L)/2 \rfloor, m$`)`
    return m
  $u \leftarrow$ `Random(`$\mathbb{F}_p$`)`
  $m \leftarrow$ `OrderRec(`$A, (4u, 1), 0, n, 1$`)`
  if $m = 0$ **then**
    return False
  else if $m > 4\sqrt{p}$ **then**
    return True
  else
    return $\perp$

4 ISOGENY VOLCANOES AND SUTHERLAND'S TEST

Our first non-elementary supersingularity test is Sutherland's algorithm [22], which actually detects supersingularity over $\mathbb{F}_{p^2}$. As such, we will need a slightly more evolved perspective on supersingularity before describing the algorithm. The facts stated here without proof are all covered in [21] (for supersingularity), and [15] and [13] (for isogeny graphs and volcanoes).

4.1 SUPERSINGULARITY IN GENERAL

Let $\mathcal{E} : y^2 = x^3 + a_2x^2 + a_4x + a_6$ be an elliptic curve over $\mathbb{F}_{p^e}$, and let $\pi : (x, y) \mapsto (x^{p^e}, y^{p^e})$ be the p^e-power Frobenius endomorphism. Like all endomorphisms, π satsifies a quadratic *characteristic polynomial* in the form

$$\chi_\pi(X) = X^2 - tX + p^e \,.$$

The integer t is called the *trace* of Frobenius, or simply the trace of $\mathcal{E}$; Hasse's theorem tells us that $|t| \leq 2p^{e/2}$. Since the $\mathbb{F}_{p^e}$-rational points of $\mathcal{E}$ are precisely the points fixed by π, we have

$$\#\mathcal{E}(\mathbb{F}_{p^e}) = \chi_\pi(1) = p^e - t + 1 \,.$$

We say that $\mathcal{E}$ is supersingular if $p \mid t$; otherwise, $\mathcal{E}$ is ordinary. (In particular, in the case $e = 1$, Hasse's theorem implies that $\mathcal{E}$ is supersingular if and only if $t = 0$, if and only if $\#\mathcal{E}(\mathbb{F}_p) = p + 1$.) Equivalently, $\mathcal{E}$ is supersingular if $\mathcal{E}[p^r](\overline{\mathbb{F}}_p) = 0$ for all $r > 0$; for ordinary curves, $\mathcal{E}[p^r](\overline{\mathbb{F}}_p) \cong \mathbb{Z}/p^e\mathbb{Z}$.

If $\mathcal{E}$ is supersingular, then its multiplication-by-p endomorphism $[p]$ is purely inseparable, hence isomorphic to the p^2-power Frobenius isogeny, which is therefore isomorphic to an endomorphism. Hence, if $\mathcal{E}$ is supersingular, then the j-invariant of $\mathcal{E}$ must be in $\mathbb{F}_{p^2}$, and $\mathcal{E}$ is $\mathbb{F}_{p^e}$-isomorphic to a supersingular curve over $\mathbb{F}_{p^2}$. Thus, supersingularity testing over $\mathbb{F}_{p^e}$ reduces immediately to supersingularity testing over $\mathbb{F}_{p^2}$.

The ring $\mathbb{Z}[\pi]$ forms a subring of the endomorphism ring $\mathrm{End}(\mathcal{E})$. If $\mathcal{E}$ is ordinary, then $\mathrm{End}(\mathcal{E})$ is an order in the quadratic imaginary field $\mathbb{Q}(\pi) \cong \mathbb{Q}(\sqrt{t^2 - 4p^e})$. If $\mathcal{E}$ is supersingular, then $\mathrm{End}(\mathcal{E})$ is a maximal order in a quaternion algebra ramified at p and ∞.

4.2 2-ISOGENY GRAPHS

An *isogeny* $\phi : \mathcal{E} \to \mathcal{E}'$, is a non-zero morphism of elliptic curves. We say ϕ is a *d-isogeny* if the associated extension of function fields has degree d (we are only concerned with 2-isogenies in this paper).

If $\phi : \mathcal{E} \to \mathcal{E}'$ is a d-isogeny, then there exists a dual d-isogeny $\hat{\phi} : \mathcal{E}' \to \mathcal{E}$. The composition of two isogenies is another isogeny, and every elliptic curve has an isogeny to itself (the identity map, for example). Isogeny is therefore an equivalence relation: the set of all elliptic curves over $\mathbb{F}_{p^e}$ decomposes into *isogeny classes*. Isogenous curves have the same trace; in particular, supersingularity is preserved by isogeny.

The 2-isogeny graph of elliptic curves over $\mathbb{F}_{p^2}$ is the graph whose vertices are isomorphism classes of elliptic curves over $\mathbb{F}_{p^2}$, and where there is an edge between two vertices corresponding to (the isomorphism class of) each 2-isogeny between elliptic curves representing the vertices. While this graph is technically directed, away from the vertices of j-invariant 0 and 1728 it behaves exactly like an undirected graph.

The 2-isogeny graphs containing ordinary and supersingular curves have very different structures.

- The 2-isogeny graph of supersingular curves over $\mathbb{F}_{p^2}$ is a connected 3-regular graph.
- The 2-isogeny graph of ordinary curves over $\mathbb{F}_{p^2}$ decomposes into a set of *volcanoes*, each formed by a (possibly trivial) cycle whose vertices form the roots of a forest of regular binary trees, all of the same height.

Figure 1 illustrates an example of a 2-isogeny volcano. Following the terminology of [13], the cycle (at the top) is called the *crater*, and the leaves of the trees form the *floor*.

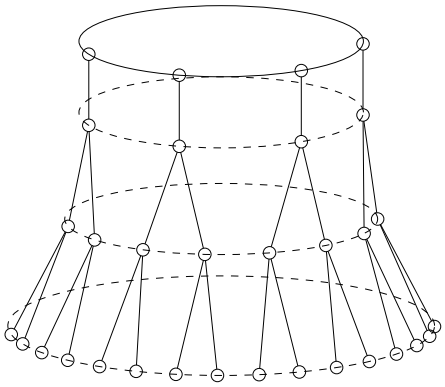


Figure 1: Example isogeny volcano.

As shown in [15], the volcano structure can be interpreted in terms of endomorphism rings. Let $\mathcal{E}/\mathbb{F}_{p^2}$ be an ordinary curve representing a vertex in a 2-isogeny volcano, let t be the trace of $\mathcal{E}$, and let $\Delta := t^2 - 4p^2$; then the algebra $\mathbb{Q}(\pi)$ generated by the Frobenius endomorphism π is isomorphic to the quadratic imaginary field $K := \mathbb{Q}(\sqrt{\Delta})$. Let Δ_0 be the fundamental discriminant of Δ, so $\Delta = c^2\Delta_0$ for some $c > 0$, which is the *conductor* (or index) of $\mathbb{Z}[\pi]$ in the maximal order $\mathcal{O}_K$ of K.

Now, if $\mathcal{E}$ is on the crater, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2 \cong \mathcal{O}_K \otimes \mathbb{Z}_2$; if $\mathcal{E}$ is on the floor, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2 \cong \mathbb{Z}[\pi] \otimes \mathbb{Z}_2$. In between, if $\mathcal{E}$ is d levels down from the crater, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2$ has index 2^d in $\mathcal{O}_K \otimes \mathbb{Z}_2$; that is, the level d gives the 2-valuation of the conductor of $\mathrm{End}(\mathcal{E})$ in $\mathcal{O}_K$. The height h of the volcano is therefore bounded by the 2-valuation of c, which can never be greater than $\log_2(\sqrt{|t^2 - 4p^2|}) \leq \log_2(\sqrt{4p^2}) = \log_2 p + 1$.

4.3 SUPERSINGULARITY TESTING WITH ISOGENY VOLCANOES

Sutherland's supersingularity test [22] determines the supersingularity of an elliptic curve $\mathcal{E}/\mathbb{F}_{p^2}$ by exploring the 2-isogeny graph around $\mathcal{E}$ and determining whether it is a volcano or not. If it is, then $\mathcal{E}$ must be ordinary; otherwise, it is supersingular.

We explore the graph using *modular polynomials*. The (classical) modular polynomial of level 2 is

$$\begin{aligned}\Phi_2(J_1, J_2) = J_1^3 + J_2^3 - J_1^2J_2^2 + 1488(J_1^2J_2 + J_1J_2^2) - 162000(J_1^2 + J_2^2) \\ + 40773375J_1J_2 + 8748000000(J_1 + J_2) - 157464000000000\,.\end{aligned}$$

This polynomial has the property that

$$\text{there exists a 2-isogeny } \mathcal{E}_1 \to \mathcal{E}_2 \iff \Phi_2(j(\mathcal{E}_1), j(\mathcal{E}_2)) = 0\,.$$

Hence, given a vertex $j(\mathcal{E})$ in the 2-isogeny graph, we can compute its neighbours by computing the roots of the cubic polynomial $\Phi_2(j_0, X)$ in $\mathbb{F}_{p^2}$.

Algorithm 4 is Sutherland's algorithm for curves $\mathcal{E}/\mathbb{F}_{p^2}$. We start three non-backtracking walks in the graph from the vertex of $\mathcal{E}$. If the graph is a volcano, then one of the walks must head towards the floor; if we find the floor (that is, a vertex with only one neighbour), then we can declare the curve ordinary. On the other hand, we know the maximal distance to the floor, if it exists; so if no walk finds the floor in less than $\log_2 p + 1$ steps, then we can declare the curve supersingular.

Initialising the three walks requires factoring the cubic $\Phi_2(j(\mathcal{E}), X)$. In every proceeding step, we can avoid backtracking by dividing the evaluated modular polynomial by $X - j'$, where j' is the j-invariant of the previous vertex. Therefore, each subsequent step in a walk requires factoring only a quadratic, which can be done easily via the quadratic formula; the cost of each step is therefore dominated by the computation of a square root in $\mathbb{F}_{p^2}$.

Algorithm 4: Sutherland's supersingularity test for elliptic curves over $\mathbb{F}_{p^2}$.

Input: $j \in \mathbb{F}_{p^2}$ (the j-invariant of an elliptic curve $\mathcal{E}/\mathbb{F}_{p^2}$)
Output: **True** or **False**

Function `IsSupersingular(`j`)`
  $\Phi \leftarrow$ `ClassicalModularPolynomial(`2`)` $\bmod p$ — `// `$\Phi(X,Y) \in \mathbb{F}_p[X,Y]$
  $f \leftarrow$ `Evaluate(`$\Phi, (j, X)$`)` — `// `$f(X) \in \mathbb{F}_{p^2}[X]$`, degree 3`
  $J \leftarrow$ `Roots(`$f, \mathbb{F}_{p^2}$`)` — `// Roots may be repeated`
  if $\#J < 3$ **then**
    return False
  $(j_1, j_2, j_3) \leftarrow J$
  $(j_1', j_2', j_3') \leftarrow (j, j, j)$
  for $0 \le s \le \lfloor \log_2 p \rfloor$ **do** — `// 2-isogeny walk steps`
    for $1 \le i \le 3$ **do** — `// Three parallel walks`
      $f_i \leftarrow$ `Evaluate(`$\Phi, (j_i, X)$`)` — `// `$f_i(X) \in \mathbb{F}_{p^2}[X]$`, degree 3`
      $f_i \leftarrow f_i/(X - j_i')$ — `// No backtracking`
      $J_i \leftarrow$ `Roots(`$f_i, \mathbb{F}_{p^2}$`)` — `// Roots may be repeated`
      if $\#J_i \neq 2$ **then** — `// We have hit the volcano floor`
        return False
      $(j_i, j_i') \leftarrow$ `Random(`J_i`)`, j_i — `// Next step`
  return True

4.4 AN IMPROVED VOLCANO TEST FOR CURVES OVER PRIME FIELDS

Sutherland suggests a speed-up for the case that the given curve is defined over $\mathbb{F}_p$—which is exactly the CSIDH setting. The key fact is that if we consider the subgraph of the supersingular 2-isogeny graph supported on vertices with j-invariants in $\mathbb{F}_p$, then this part looks like a particularly stumpy volcano:[2] that is, it consists of a cycle, with descending trees of height at most 1 (see [11] for a detailed treatment of this graph). When we allow vertices over $\mathbb{F}_{p^2}$, then we obtain the usual complete 3-regular graph. On the other hand, if $\mathcal{E}/\mathbb{F}_p$ is an ordinary curve, then it is in the upper part of the volcano of curves over $\mathbb{F}_{p^2}$, because the ring generated by the p^2-power Frobenius is (obviously) a subring of the ring generated by the p-power Frobenius.

Using these facts, when running Sutherland's algorithm from a vertex in $\mathbb{F}_p$, the "descending" path can quickly be determined: as soon as we encounter a j-invariant in $\mathbb{F}_{p^2}$ (which, in the supersingular case, will happen within two steps) we know that this is the only possibility for a "descending" walk, so we can continue that walk and stop the two others. This simple modification translates into a rough 3× speedup when the input is in $\mathbb{F}_p$.

In fact, we can do even better. Sutherland bounds the length of a maximal descending walk by $\log_2 p + 1$, which is the maximum height of the 2-isogeny volcano containing any ordinary curve over $\mathbb{F}_{p^2}$. But as we noted above,

[2] In fact, this description corresponds to the graph whose vertices correspond to $\mathbb{F}_p$-isomorphism classes of supersingular curves over $\mathbb{F}_p$; using j-invariants corresponds to quotienting this structure by the involution mapping each curve to its quadratic twist, but in practice this makes no difference to the operation or correctness of the algorithm.

when working with curves defined over $\mathbb{F}_p$, we can quickly step into the $\mathbb{F}_{p^2}$-part of the 2-isogeny graph, and then continue there. Lemma 1 shows that the length of a "descending" walk in the $\mathbb{F}_{p^2}$-part of the graph is bounded by $\frac{1}{2}\log_2 p + 1$: that is, essentially half of Sutherland's bound. Combined with the modification above, this yields a $\approx 6\times$ speedup over the general algorithm.

Lemma 1. *Let $\mathcal{E}$ be an ordinary elliptic curve over $\mathbb{F}_p$. The height of the $\mathbb{F}_{p^2}$-part of the 2-isogeny volcano containing $\mathcal{E}$ is at most $\frac{1}{2}\lfloor\log_2 p\rfloor + 1$.*

Proof. Let π_p be the p-power Frobenius endomorphism of $\mathcal{E}$, and t_p its trace. The height of the $\mathbb{F}_{p^2}$-part of the 2-isogeny volcano containing $\mathcal{E}$ is the 2-valuation of the conductor of $\mathbb{Z}[\pi_p^2]$ in $\mathbb{Z}[\pi_p]$. The discriminants of $\mathbb{Z}[\pi_p]$ and $\mathbb{Z}[\pi_p^2]$ are $\Delta_1 = t_p^2 - 4p$ and $\Delta_2 = (t_p^2 - 2p)^2 - 4p^2 = t_p^2\Delta_1$, respectively, so the relative conductor is $|t_p|$. But $|t_p| < 2\sqrt{p}$ by Hasse's theorem; hence, the 2-valuation of $|t_p|$ is bounded above by $\log_2(2\sqrt{p}) = \frac{1}{2}\log_2 p + 1$. □

Our second improvement is to streamline the isogeny step computations. Rather than computing roots of evaluations of the classical modular polynomial Φ_2, we can use Lemma 2 to compute explicit 2-isogenies.

Lemma 2. *Let $\mathcal{E}/\mathbb{F}_{p^2}$ be an elliptic curve defined by an equation $y^2 = x(x^2 + a_2x + a_4)$. Set $D := a_2^2 - 4a_4$, and choose a square root $\delta := \sqrt{D}$ in $\overline{\mathbb{F}}_p$. Then $\alpha := -(a_2 - \delta)/2$ is a root of $x^2 + a_2x + a_4$, there is a quotient isogeny*

$$\varphi : \mathcal{E} \longrightarrow \mathcal{E}/\langle(\alpha, 0)\rangle \cong \mathcal{E}' : y^2 = x(x^2 + a_2'x + a_4') \qquad \textit{where} \qquad \begin{cases} a_2' = a_2 - 3\delta\,, \\ a_4' = a_2(a_2 + \delta)/2 - a_4 \end{cases} \tag{4}$$

defined over $\mathbb{F}_{p^2}(\delta)$, and the kernel of its dual isogeny $\widehat{\varphi}$ is generated by $(0, 0)$.

Proof. Follows on composing the normalized 2-isogeny given by Vélu's formulæ (see [23] or [15, 2.4]) with the map $(x, y) \mapsto (x + \delta, y)$. □

We can now define our variant of Sutherland's algorithm. The subroutine `SqrtFp2`, given α in $\mathbb{F}_{p^2}$ (or $\mathbb{F}_p$), returns a square root of α in $\mathbb{F}_{p^2}$ if one exists, or $\perp$ if not. When $p \equiv 3 \pmod 4$—as in CSIDH, and many other isogeny-based cryptosystems—we can use the efficient arithmetic for $\mathbb{F}_{p^2}$ from Appendix A.

Algorithm 5: Modified Sutherland supersingularity test for Montgomery curves over $\mathbb{F}_p$.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**

Function `IsSupersingular(`A`)`
  if $p \not\equiv 3 \pmod 4$ **then** `// No supersingular curve over` $\mathbb{F}_p$ `has a Montgomery model`
    return False
  if $A = 0$ **then** `//` j`-invariant 1728: always supersingular when` $p \equiv 3 \pmod 4$
    return True
  $(a_2, D) \leftarrow (A, A^2 - 4)$
  $\delta \leftarrow$ `SqrtFp2(`D`)` `//` D `is in` $\mathbb{F}_p$`, so` δ `cannot be` $\perp$
  if $\delta \in \mathbb{F}_p$ **then** `//` D `is a square in` $\mathbb{F}_p$
    return False
  for $0 \le i \le \frac{1}{2}\lfloor\log_2 p\rfloor + 1$ **do**
    $(a_2, D) \leftarrow (a_2 - 3\delta, 8(D - \delta \cdot a_2))$
    $\delta \leftarrow$ `SqrtFp2(`D`)`
    if $\delta = \perp$ **then** `//` D `is not a square in` $\mathbb{F}_{p^2}$
      return False
  return True

Proposition 1. *Given a Montgomery coefficient A in $\mathbb{F}_p$ corresponding to an elliptic curve $\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)$, Algorithm 5 returns* **True** *if and only if $\mathcal{E}$ is supersingular. It requires $O(\log^2 p)$ $\mathbb{F}_p$-operations (in the worst case, which is when $\mathcal{E}$ is supersingular) and $O(1)$ $\mathbb{F}_p$-elements worth of space.*

Proof. First, recall that every Montgomery model has cardinality divisible by 4, and therefore a Montgomery model can only be supersingular if $p \equiv 3 \pmod 4$ (since otherwise $4 \nmid (p + 1)$). We can immediately dispose of the special case $A = 0$: this curve has j-invariant 0, and is always supersingular when $p \equiv 3 \pmod 4$.

For the general case where $A \neq 0$, let α and $1/\alpha$ be the roots of $x^2 + Ax + 1$ in $\mathbb{F}_{p^2}$. Having a Montgomery model implies that $\mathcal{E}_A$ is on the floor of the $\mathbb{F}_p$-volcano [4, Prop. 8]; there is only one $\mathbb{F}_p$-rational 2-isogeny (its kernel is $(0, 0)$), and this isogeny must be ascending. In particular, if $\mathcal{E}_A$ is supersingular then $x^2 + Ax + 1$ can have no roots in $\mathbb{F}_p$; hence, if $A^2 - 4$ is a square in $\mathbb{F}_p$, then $\mathcal{E}_A$ is ordinary and we can return **False** at Line 9.

Otherwise, Lemma 2 shows that we can continue with a non-backtracking walk into the $\mathbb{F}_{p^2}$-part of the 2-isogeny graph by iterating the mapping $(a_2, D = a_2^2 - 4a_4) \mapsto (a_2', D' = (a_2')^2 - 4a_4')$ defined by (4), starting with $(a_2, D) = (A, A^2 - 4)$. The $\mathbb{F}_{p^2}$-rationality of each step depends on whether or not D is a square in $\mathbb{F}_{p^2}$; the choice between each pair of descending isogenies is made arbitrarily, according to which of the two square roots is returned by `SqrtFp2`. Algorithm 5 repeats this process until we either hit the floor of $\mathbb{F}_{p^2}$-rationality and conclude that $\mathcal{E}_A$ is ordinary, or go beyond the maximal path length specified by Lemma 1 and conclude that $\mathcal{E}_A$ is supersingular.

Algorithm 5 is therefore correct. It requires (in the worst case) one square root in $\mathbb{F}_p$ and $\frac{1}{2}\log_2 p + 2$ iterations of a loop consisting of a square root in $\mathbb{F}_{p^2}$ plus a handful of $\mathbb{F}_{p^2}$-operations. Each square root costs $O(\log p)$ $\mathbb{F}_p$-operations using the Tonelli–Shanks algorithm, so we have a total complexity of $O(\log^2 p)$ $\mathbb{F}_p$-operations. □

5 DIVISION POLYNOMIALS AND DOLISKANI'S TEST

Doliskani gives an interesting probabilistic supersingularity test in [12], based on Polynomial Identity Testing and properties of division polynomials. To the best of our knowledge, this algorithm has not yet been used in practice. We will provide some important algorithmic improvements that will ultimately make this the most efficient of our supersingularity tests.

5.1 DIVISION POLYNOMIALS AND SUPERSINGULARITY

Recall that that for $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\}\,.$$

For $\mathcal{E} : y^2 = x(x^2 + Ax + 1)$, the first few division polynomials are

$$\psi_{\mathcal{E},0} = 0\,, \qquad \psi_{\mathcal{E},1} = 1\,, \qquad \psi_{\mathcal{E},2} = 2y\,,$$

then (for Montgomery models)

$$\psi_{\mathcal{E},3} = 3x^4 + 4Ax^3 + 6x^2 - 1\,, \qquad \psi_{\mathcal{E},4} = 4y(x^2 - 1)(x^4 + 2Ax^3 + 6x^2 + 2Ax + 1)\,.$$

The higher-degree polynomials satisfy the standard recurrences

$$\psi_{\mathcal{E},2m} = (\psi_{\mathcal{E},m-1}^2\psi_{\mathcal{E},m}\psi_{\mathcal{E},m+2} - \psi_{\mathcal{E},m-2}\psi_{\mathcal{E},m}\psi_{\mathcal{E},m+1}^2)/\psi_{\mathcal{E},2} \qquad \text{for } m \geq 3\,, \tag{5}$$

$$\psi_{\mathcal{E},2m+1} = \psi_{\mathcal{E},m}^3\psi_{\mathcal{E},m+2} - \psi_{\mathcal{E},m-1}\psi_{\mathcal{E},m+1}^3 \qquad \text{for } m \geq 2\,. \tag{6}$$

We see that $\psi_{\mathcal{E},m}^2$ is a polynomial in $\mathbb{F}_p[A][x]$ for any $m > 0$, and indeed $\psi_{\mathcal{E},m}$ is in $\mathbb{F}_p[A][x]$ for odd m.

Lemma 3. *The p-th division polynomial satisfies $\psi_{\mathcal{E},p}(x) = \widetilde{\psi}_{\mathcal{E},p}(x)^p$ where $\widetilde{\psi}_{\mathcal{E},p}(x)$ is either a polynomial of degree $(p-1)/2$ if $\mathcal{E}$ is ordinary, or ± 1 if $\mathcal{E}$ is supersingular. In particular,*

$$\psi_{\mathcal{E},p}^2(x) = 1 \iff \mathcal{E} \text{ is supersingular.}$$

Proof. See Theorem 3.1 and Propositions 3.3 and 3.4 of [14] (where the sign in the ± 1 for the supersingular case is made completely explicit, though we only need the fact that $\psi_{\mathcal{E},p}^2(x) = 1$ here). □

Example 1. *Let $\mathcal{E} : y^2 = x(x^2 + Ax + 1)$ be the generic Montgomery model over $\mathbb{F}_7(A)$. We find*

$$\psi_{\mathcal{E},7}(x) = \widetilde{\psi}_{\mathcal{E},7}(x)^7 \qquad \text{where} \qquad \widetilde{\psi}_{\mathcal{E},7}(x) = A(A^2-1)\big(x^3 - A(A^2-2)x^2 + 2(A^2+1)x\big) - 1\,.$$

In particular, if $A \in \{-1, 0, 1\}$ then $\widetilde{\psi}_{\mathcal{E},7} = -1$ and $\mathcal{E}$ is supersingular; otherwise, $\deg \widetilde{\psi}_{\mathcal{E},7} = 3$ and $\mathcal{E}$ is ordinary.

Doliskani's test applies basic Polynomial Identity Testing to check the supersingularity criterion of Lemma 3. The algorithm presented in [12] samples a uniformly random u in $\mathbb{F}_{p^2}$, and computes $\widetilde{\psi}_{\mathcal{E},p}(u)^2$ as $\psi_{\mathcal{E},p}(u)^2$. If this yields $\psi_{\mathcal{E},p}(u) = 1$, then the Schwartz–Zippel lemma (see e.g. [19]) tells us that $\psi_{\mathcal{E},p}^2(x) = 1$, and $\mathcal{E}$ is supersingular, with probability $1 - (\deg \widetilde{\psi}_{\mathcal{E},p})/\#\mathbb{F}_{p^2} = 1 - ((p-1)/2)/p^2 \approx 1 - 1/2p$.

5.2 EFFICIENT EVALUATION OF DIVISION POLYNOMIALS

To evaluate $\psi_{\mathcal{E},p}(u)$, Doliskani proposes an algorithm based on the standard recurrences of (5) and (6), which resembles a vectorial addition chain where the vector tracks the values of nine consecutively-indexed division polynomials. This algorithm seems to be folklore, but Cheng gives a detailed analysis in [6].

We can significantly improve the efficiency of division polynomial evaluation by using the link between division polynomials and scalar multiplication. Recall that

$$[m](x,y) = \left(\frac{\phi_{\mathcal{E},m}(x)}{\psi_{\mathcal{E},m}(x)^2}, \frac{\omega_{\mathcal{E},m}(x,y)}{\psi_{\mathcal{E},m}(x)^3}\right) \quad \text{where} \quad \phi_{\mathcal{E},m}(x) := x\psi_{\mathcal{E},m}(x)^2 - \psi_{\mathcal{E},m+1}(x)\psi_{\mathcal{E},m-1}(x) \tag{7}$$

and $\omega_{\mathcal{E},m}(x,y) = (4y)^{-1}\big(\psi_{\mathcal{E},m+2}(x)\psi^2_{\mathcal{E},m-1}(x) - \psi_{\mathcal{E},m-2}(x)\psi^2_{\mathcal{E},m+1}(x)\big)$ (though we will not need ω_m in what follows). Hence, if $(u : v : 1)$ is a point on $\mathcal{E}_A$ and we use the Montgomery ladder to compute the (X, Z)-coordinates of $[p](u : v : 1)$, then (7) tells us that the X and Z coordinates are $\phi_{\mathcal{E},p}(u)$ and $\psi^2_{\mathcal{E},p}(u)$, respectively, *up to a common projective factor*—which we can predict and remove. Proposition 2 makes this explicit for general m.

Proposition 2. *On input A, m, and* $(x, 1)$*, Algorithm 1 (the Montgomery ladder) returns*

$$(X_m, Z_m) = (\phi_{\mathcal{E},m}(x) \cdot f_m(x), \psi^2_{\mathcal{E},m}(x) \cdot f_m(x)) \quad \textit{where} \quad f_m(x) := (4x)^{m(2^{\operatorname{len}(m)}-m)}\,. \tag{8}$$

Proof. Let P be a point on $\mathcal{E}_A$ with x-coordinate $x \neq 0$. We saw in 2 that $\texttt{Ladder}(A, m, (x, 1))$ returns $\mathsf{R}_0 = (X_m, Z_m)$ and $\mathsf{R}_1 = (X_{m+1}, Z_{m+1})$ such that $X_m/Z_m = x([m]P)$ and $X_{m+1}/Z_{m+1} = x([m + 1]P)$, so (7) implies

$$\begin{aligned}
\mathsf{R}_0 &= (X_m, Z_m) = \big(\phi_{\mathcal{E},m}(x) \cdot f_m(x), \psi^2_{\mathcal{E},m}(x) \cdot f_m(x)\big)\,,\\
\mathsf{R}_1 &= (X_{m+1}, Z_{m+1}) = \big(\phi_{\mathcal{E},m+1}(x) \cdot g_m(x), \psi^2_{\mathcal{E},m+1}(x) \cdot g_m(x)\big)
\end{aligned}$$

for some projective factors $f_m(x)$ and $g_m(x)$. We claim that

$$f_m(x) = (4x)^{F_m} \quad \text{and} \quad g_m(x) = (4x)^{G_m} \quad \text{where} \quad \begin{cases} F_m := m(2^{\operatorname{len}(m)} - m)\,, \\ G_m := (m + 1)(2^{\operatorname{len}(m)} - (m + 1))\,. \end{cases} \tag{9}$$

We proceed by induction. First, the base case: on input $(A, 1, (x, 1))$, Algorithm 1 outputs

$$\begin{aligned}
\mathsf{R}_0 &= (4x^2, 4x) = (\phi_{\mathcal{E}_A,1}(x) \cdot 4x, \psi_{\mathcal{E},1}(x) \cdot 4x)\,,\\
\mathsf{R}_1 &= (x^4 - 2x^2 + 1, 4x^3 + 4Ax^2 + 4x) = (\phi_{\mathcal{E},2}(x) \cdot 1, \psi^2_{\mathcal{E},2}(x) \cdot 1)\,,
\end{aligned}$$

which is exactly (9) with $m = 1$. For the inductive step: looking at the differential addition and doubling formulæ in (2) and (3), respectively, we see that the projective factors are defined by mutually recursive relationships:

$$f_{2m}(x) = f_m^4(x)\,, \qquad f_{2m+1}(x) = g_{2m}(x) = 4x f_m^2(x) g_m^2(x)\,, \qquad g_{2m+1}(x) = g_m^4(x)\,. \tag{10}$$

Substituting (9) into (10), it suffices to show that

$$F_{2k} = 4F_k\,, \quad G_{2k} = 2F_k + 2G_k + 1 = F_{2k+1}\,, \quad \text{and} \quad G_{2k+1} = 4G_k\,.$$

Noting that $\operatorname{len}(2k) = \operatorname{len}(k) + 1$, we find that

$$F_{2k} = 2k(2^{\operatorname{len}(2k)} - 2k) = 2k(2 \cdot 2^{\operatorname{len}(k)} - 2k) = 4k(2^{\operatorname{len}(k)} - k) = 4F_k$$

and similarly

$$G_{2k+1} = (2k + 2)(2^{\operatorname{len}(2k)} - (2k + 2)) = 2(k + 1)(2 \cdot 2^{\operatorname{len}(k)} - 2(k + 1)) = 4(k + 1)(2^{\operatorname{len}(k)} - (k + 1)) = 4G_k\,.$$

Since $F_{2k+1} = G_{2k}$ by definition, it remains to show that $G_{2k} = 2F_k + 2G_k + 1$, and indeed

$$\begin{aligned}
G_{2k} &= (2k + 1)(2^{\operatorname{len}(2k)} - (2k + 1))\\
&= (2k + 1)(2 \cdot 2^{\operatorname{len}(k)} - 2k) - 2k - 1\\
&= 2k(2^{\operatorname{len}(k)} - k) + 2(k + 1)(2^{\operatorname{len}(k)} - (k + 1)) + 1\\
&= 2F_k + 2G_k + 1\,,
\end{aligned}$$

as required. □

When $m = p$, we can use the fact that $\alpha^p = \alpha$ for α in $\mathbb{F}_p$ to replace (8) with the simpler formula

$$f_p(u) = (4\bar{u})^{2^{\mathrm{len}(p)}}/(4u) \qquad \text{where} \qquad \bar{u} = u^p\,. \tag{11}$$

If $p \equiv 3 \pmod 4$ and we realise $\mathbb{F}_{p^2}$ as $\mathbb{F}_p(i)$, then $\overline{a+bi} = a - bi$, so computing $\bar{u} = u^p$ is almost free.

Computing $\psi^2_{\mathcal{E},m}(u)$ using Proposition 2 requires $\approx 19\,\mathrm{len}(m)$ arithmetic operations (each bit of m requires one 18-operation ladder step, plus one squaring in the final exponentiation). This compares very favourably with Cheng's algorithm, which requires $\approx 72\,\mathrm{len}(m)$ operations to compute $\psi_{\mathcal{E},m}(u)$, and more storage (see [6, 4]).

5.3 DOLISKANI'S TEST REVISITED

Doliskani's original test from [12] samples a value u uniformly at random from $\mathbb{F}_{p^2}$, and returns **True** if and only if $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ and $\psi^2_{\mathcal{E},p}(u) = 1$. Algorithm 6 combines Doliskani's test with our efficient algorithm for evaluating division polynomials. Remarkably, we can replace the computation of $\psi^2_{\mathcal{E},p}(u), \psi_{\mathcal{E},p+1}(u)$, and $\psi_{\mathcal{E},p-1}(u)$ with a single application of the Montgomery ladder over $\mathbb{F}_{p^2}$.

Algorithm 6: Doliskani's PIT supersingularity test with fast ladder-based division polynomial evaluation

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**

```
1 Function IsSupersingular(A)
2     u ← Random(F_{p^2} \ {0})
3     (X_p, Z_p) ← Ladder(A, p, (u, 1))
4     if X_p ≠ u · Z_p then                    // Order of (u : * : 1) does not divide p ± 1
5         return False
6     if 4u · Z_p = (4ū)^{2^len(p)} then       // Polynomial Identity Testing:  Z_p = ψ²_{E,p}(u) f_p(u)
7         return True
8     else
9         return False
```

Proposition 3. *Let A in $\mathbb{F}_p$ be the Montgomery coefficient of an elliptic curve $\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)$.*

- *If $\mathcal{E}_A$ is supersingular, then Algorithm 6 returns* **True***.*
- *If $\mathcal{E}_A$ is ordinary, then Algorithm 6 returns* **True** *with probability $1/(2p+2)$, and* **False** *otherwise.*

Algorithm 6 requires one random element of $\mathbb{F}_{p^2}$, $O(\log p)$ $\mathbb{F}_p$-operations and $O(1)$ $\mathbb{F}_p$-elements worth of space.

Proof. As mentioned above, Doliskani's test samples u uniformly at random from $\mathbb{F}_{p^2}$, and returns **True** if and only if $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ and $\psi^2_{\mathcal{E},p}(u) = 1$. We make a small change here, requiring $u \neq 0$ to ensure that the Montgomery ladder returns valid output on input $(u, 1)$.

Let $v \in \mathbb{F}_{p^4}$ be such that $P = (u : v : 1)$ is a point in $\mathcal{E}_A(\mathbb{F}_{p^4})$. If $\mathcal{E}_A/\mathbb{F}_p$ is supersingular, then $\mathcal{E}_A(\mathbb{F}_{p^4}) \cong (\mathbb{Z}/(p^2-1)\mathbb{Z})^2$, so the condition $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ amounts to checking that P has order dividing $p^2 - 1$. But we can equivalently check that $[p]P = \pm P$, which is what Line 4 does.

It remains to check that $\psi^2_{\mathcal{E},p}(u) = 1$, which holds if and only if $\widetilde{\psi}^2_{\mathcal{E},p}(\bar{u}) = 1$. The probability that $\widetilde{\psi}^2_{\mathcal{E},p}(\bar{u}) = 1$ while $\widetilde{\psi}_{\mathcal{E},p}(x) \neq \pm 1$ (that is, while $\mathcal{E}_A$ is ordinary) is $\deg \widetilde{\psi}_{\mathcal{E},p}/\#(\mathbb{F}_{p^2} \setminus \{0\}) = ((p-1)/2)/(p^2-1) = 1/(2p+2)$ by the Schwartz–Zippel lemma. We compute $\psi^2_{\mathcal{E},p}(u)$ using Proposition 2 and (11). This requires one application of the Montgomery ladder over $\mathbb{F}_{p^2}$ with a scalar of length $\mathrm{len}(p) = \lceil \log_2 p \rceil$, followed by $\mathrm{len}(p)$ squarings in $\mathbb{F}_p$; both cost $O(\log p)$ $\mathbb{F}_p$-operations, storing only $O(1)$ $\mathbb{F}_p$-elements. □

Remark 2. *The projective factor f_p of Proposition 2 depends on the ladder algorithm as presented in Algorithm 1 and the differential addition and doubling formulæ in 2. If the ladder is initialised differently, or replaced with an optimal differential addition chain, or if alternative formulæ are used, then f_p should be redefined accordingly.*

Remark 3. *Doliskani's test is not limited to curves over $\mathbb{F}_p$: indeed, it was designed for curves over $\mathbb{F}_{p^2}$. Our algorithm and implementation makes little use of the fact that the input is defined over $\mathbb{F}_p$ (this slightly simplifies multiplication by curve constants, but this makes only a slight difference to the runtime): it should have essentially the same cost for inputs in $\mathbb{F}_{p^2}$, and the same probability of correctness.*

6 OTHER ELLIPTIC CURVE MODELS

Looking beyond the Montgomery models used in CSIDH, the supersingularity tests described here can all be easily adapted to other elliptic curve models over $\mathbb{F}_p$, such as short Weierstrass curves or (twisted) Edwards curves. Such adaptations may be required for key validation in other isogeny-based systems such as CSURF [3], but they might also be useful for purely number-theoretic calculations.

Elementary tests The random-point tests of Algorithms 2 and 3 work essentially unchanged for other curve models, once the Montgomery ladder has been replaced with a suitable high-speed scalar multiplication routine, and tests like $Z \neq 0$ replaced with comparisons against the neutral element of the group.

Sutherland's test Sutherland's original supersingularity test over $\mathbb{F}_{p^2}$ (Algorithm 4) works for any curve model. Our modified version over $\mathbb{F}_p$ (Algorithm 5) requires extensive changes for alternative curve models, though the underlying algorithm is very similar:

- Lines 2 and 4 are Montgomery-specific, and must be replaced (or omitted) according to the curve model.
- Lines 6 through 14 depend on the Montgomery-specific formulæ from Lemma 2, which should be replaced with an appropriate analogue for the given curve model.
- Lines 7 through 9 use the fact that supersingular Montgomery curves are necessarily on the floor of the $\mathbb{F}_p$-volcano for $\ell = 2$. This is not true for more general curves: for example, the curves in CSURF [3] are supposed to be on the surface. These lines should be replaced with an examination of the neighbourhood of the starting vertex to find a neighbour on the floor, if it exists, before proceeding as usual.

Doliskani's test Our modification of Doliskani's test (Algorithm 6) extends to other projective curve models. The Montgomery ladder must be replaced with a suitable high-speed scalar multiplication, and as we noted in Remark 2, the projective factor must be adapted to fit that scalar multiplication algorithm and the curve arithmetic that it uses.

7 CONCLUSION

This paper improves two supersingularity tests, originally due to Sutherland and Doliskani, and compares them with the state-of-the-art in the context of CSIDH public-key validation.

Our modification of Sutherland's algorithm specialized to prime fields reduces the running time and space; while it does not change the asymptotic complexity, it does improve the constant hidden by the big-O. It performs relatively slowly for valid keys (though it is still quite practical for CSIDH-512 parameters), but it rejects invalid keys much faster than the other tests: it is therefore probably more useful in computational number theory (where we mostly encounter ordinary input) than in CSIDH key validation (where we mostly expect supersingular input from honest parties). In any case, it has the advantages of low memory and definitive proof of supersingularity.

Our modification of Doliskani's algorithm shows a more significant improvement due to our new method for evaluating squared division polynomials. This algorithm achieved a substantial speedup over the alternatives for valid CSIDH-512 keys, while remaining competitive on invalid keys. It uses less memory than the currently-used product-tree algorithm, and is far simpler to implement correctly. We therefore suggest that this algorithm is a better choice for key validation in CSIDH and similar isogeny-based protocols.

Our benchmarks all used the CSIDH-512 parameter set, with a 512-bit prime p. We expect that our algorithms and results will be relevant for larger primes, but more work needs to be done to optimize these cases.

REFERENCES

[1] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. "CTIDH: faster constant-time CSIDH". In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.4 (2021), pp. 351–387. DOI: 10.46586/tches.v2021.i4.351-387.

[2] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. "CSI-FiSh: Efficient Isogeny Based Signatures Through Class Group Computations". In: *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I*. Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11921. Lecture Notes in Computer Science. Springer, 2019, pp. 227–247. DOI: 10.1007/978-3-030-34578-5_9.

[3] Wouter Castryck and Thomas Decru. "CSIDH on the Surface". In: *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020, Paris, France, April 15-17, 2020, Proceedings*. Ed. by Jintai Ding and Jean-Pierre Tillich. Vol. 12100. Lecture Notes in Computer Science. Springer, 2020, pp. 111–129. DOI: 10.1007/978-3-030-44223-1_7.

[4] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. "CSIDH: An Efficient Post-Quantum Commutative Group Action". In: *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*. Ed. by Thomas Peyrin and Steven D. Galbraith. Vol. 11274. Lecture Notes in Computer Science. Springer, 2018, pp. 395–427. DOI: 10.1007/978-3-030-03332-3_15.

[5] Jorge Chávez-Saab, Jesús-Javier Chi-Domınguez, Samuel Jaques, and Francisco Rodrıguez-Henrıquez. "The SQALE of CSIDH: Square-root Vélu quantum-resistant isogeny action with low exponents". In: *Journal of Cryptographic Engineering* (2021). DOI: 10.1007/s13389-021-00271-w.

[6] Qi Cheng. "Straight-line programs and torsion points on elliptic curves". In: *Computational Complexity* 12 (2003), pp. 150–161. DOI: 10.1007/s00037-003-0180-0.

[7] Craig Costello and Benjamin Smith. "Montgomery curves and their arithmetic: The case of large characteristic fields". In: *Journal of Cryptographic Engineering* 8.3 (2018), pp. 227–240. DOI: 10.1007/s13389-017-0157-6.

[8] Luca De Feo and Steven D. Galbraith. "SeaSign: Compact Isogeny Signatures from Class Group Actions". In: *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. Lecture Notes in Computer Science. Springer, 2019, pp. 759–789. DOI: 10.1007/978-3-030-17659-4_26.

[9] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. "SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies". In: *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12491. Lecture Notes in Computer Science. Springer, 2020, pp. 64–93. DOI: 10.1007/978-3-030-64837-4_3.

[10] Luca De Feo and Michael Meyer. "Threshold Schemes from Isogeny Assumptions". In: *Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12111. Lecture Notes in Computer Science. Springer, 2020, pp. 187–212. DOI: 10.1007/978-3-030-45388-6_7.

[11] Christina Delfs and Steven D. Galbraith. "Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$". In: *Des. Codes Cryptogr.* 78.2 (2016), pp. 425–440. DOI: 10.1007/s10623-014-0010-1.

[12] Javad Doliskani. "On division polynomial PIT and supersingularity". In: *Appl. Algebra Eng. Commun. Comput.* 29.5 (2018), pp. 393–407. DOI: 10.1007/s00200-018-0349-z.

[13] Mireille Fouquet and François Morain. "Isogeny Volcanoes and the SEA Algorithm". In: *Algorithmic Number Theory. ANTS 2002*. Ed. by Claus Fieker and David R. Kohel. Vol. 2369. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 276–291. ISBN: 978-3-540-45455-7. DOI: 10.1007/3-540-45455-1_23.

[14] Josep González. "On the p-th division polynomial". In: *Journal of Number Theory* 233 (2022), pp. 285–300. DOI: 10.1016/j.jnt.2021.06.011.

[15] David R. Kohel. "Endomorphism rings of elliptic curves over finite fields". http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf. PhD thesis. University of California at Berkeley, 1996.

[16] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. "Compact, Efficient and UC-Secure Isogeny-Based Oblivious Transfer". In: *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12696. Lecture Notes in Computer Science. Springer, 2021, pp. 213–241. DOI: 10.1007/978-3-030-77870-5_8.

[17] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. CRC press, 2018.

[18] Peter L. Montgomery. "Speeding the Pollard and elliptic curve methods of factorization". In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. DOI: 10.1090/S0025-5718-1987-0866113-7.

[19] Jacob T. Schwartz. "Fast probabilistic algorithms for verification of polynomial identities". In: *J. ACM* 27.4 (1980), pp. 701–717. DOI: 10.1145/322217.322225.

[20] Michael Scott. "A note on the calculation of some functions in finite fields: Tricks of the trade". IACR ePrint 2020/1497, https://ia.cr/2020/1497. 2020.

[21] Joseph H. Silverman. *The arithmetic of elliptic curves*. 2nd edition. New York, NY: Springer-Verlag, 2009. ISBN: 0387094938.

[22] Andrew V. Sutherland. "Identifying supersingular elliptic curves". In: *LMS Journal of Computation and Mathematics* 15 (2012), pp. 317–325. DOI: 10.1112/S1461157012001106.

[23] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences, Série A* 273 (July 1971), pp. 238–241.

A FINITE FIELD ARITHMETIC

Square roots in $\mathbb{F}_p$. In our case we have $p \equiv 3 \mod 4$, so the classic Tonelli–Shanks algorithm (see e.g. [17, 3.51]) reduces to computing $t = a^{(p+1)/4}$ for $a \in \mathbb{F}_p$ (which can be done using a precomputed optimal chain of squares and multiplications). If $ta^2 = a$ then t is a square root of a; otherwise, a is not a square in $\mathbb{F}_p$.

Representing $\mathbb{F}_{p^2}$. Since $p \equiv 3 \pmod 4$, we can realise $\mathbb{F}_{p^2}$ as $\mathbb{F}_p(i)$, where $i^2 = -1$. Elements x of $\mathbb{F}_{p^2}$ are encoded as the pair of elements (x_r, x_i) of $\mathbb{F}_p$ (the "real" and "imaginary" parts) such that $x = x_r + x_i \cdot i$.

Addition. Given two elements $a, b \in \mathbb{F}_{p^2}$, we can compute $c = a + b$ at the cost of two additions in $\mathbb{F}_p$ using

$$c_r = a_r + b_r \mod p\,, \qquad c_i = a_i + b_i \mod p\,.$$

Subtraction. Similarly, we can compute $c = a - b$ at the cost of two subtractions in $\mathbb{F}_p$ using

$$c_r = a_r - b_r \mod p\,, \qquad c_i = a_i - b_i \mod p\,.$$

Multiplication. We can compute $c = a \cdot b$ for 4 multiplications, 1 addition, and 1 subtraction in $\mathbb{F}_p$ using

$$c_r = a_r \cdot b_r - a_i \cdot b_i \mod p\,, \qquad c_i = a_i \cdot b_r + a_r \cdot b_i \mod p\,.$$

A Karatsuba-style method computes c with 3 multiplications, 3 additions, and 2 subtractions: first we compute

$$t_0 := a_r \cdot b_r\,, \quad t_1 := a_i \cdot b_i\,, \quad t_2 := a_r + a_i\,, \quad t_3 := b_r + b_i\,, \quad t_4 := t_2 \cdot t_3\,, \quad t_5 := t_0 + t_1\,,$$

and then

$$c_r = t_0 - t_1 \mod p\,, \qquad c_i = t_4 - t_5 \mod p\,.$$

Square roots in $\mathbb{F}_{p^2}$. Algorithm 7 computes square roots in $\mathbb{F}_{p^2}$ following the method of [20] with some minor corrections. The exponentiations in Lines 4 and 11 are done using an optimal precomputed addition chain.

Algorithm 7: Square root in $\mathbb{F}_{p^2} = \mathbb{F}_p(i)$ where $p \equiv 3 \pmod 4$ and $i = \sqrt{-1}$, as in [20].

Input: x in $\mathbb{F}_p(i)$
Output: r in $\mathbb{F}_p(i)$ such that $r^2 = x$, if it exists; otherwise $\perp$

Function `SqrtFp2(`x`)`
- $a + bi \leftarrow x$ // $a, b \in \mathbb{F}_p$
- $\delta \leftarrow a^2 + b^2$ // δ = norm of x
- $\lambda \leftarrow \delta^{(p-3)/4}$ // λ = progenitor of δ
- $\rho \leftarrow \delta \cdot \lambda$ // ρ is candidate square root of δ
- **if** $\rho^2 \neq \delta$ **then** // δ not square in $\mathbb{F}_p \implies x$ not square in $\mathbb{F}_{p^2}$
  - **return** $\perp$
- $\gamma \leftarrow (a + \rho)/2$
- **if** $\gamma = 0$ **then** // Can happen when $b = 0$ and $\rho = -a$
  - $\gamma = -\rho$ // Now $\gamma = (a - \rho)/2$
- $\mu \leftarrow \gamma^{(p-3)/4}$ // μ = progenitor of γ
- $\sigma \leftarrow \gamma \cdot \mu$ // σ = candidate square root of γ
- $\gamma^{-1} \leftarrow \sigma \cdot \mu^3$ // True inverse of γ
- $\tau \leftarrow \sigma \cdot \gamma^{-1}$ // τ = candidate square root of γ^{-1}
- $\omega \leftarrow (b/2) \cdot \tau$
- **if** $\sigma^2 = \gamma$ **then**
  - **return** $\sigma + \omega i$
- **else** // $-\sigma = \sqrt{-\gamma}$ and $\tau = \sqrt{-\gamma^{-1}}$
  - **return** $\omega - \sigma i$

Bibliography

[1] Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 699–728, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

[2] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 411–439, Daejeon, South Korea, December 7–11, 2020. Springer, Cham, Switzerland.

[3] Navid Alamati, Hart Montgomery, Sikhar Patranabis, and Pratik Sarkar. Two-round adaptively secure MPC from isogenies, LPN, or CDH. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part II*, volume 13091 of *Lecture Notes in Computer Science*, pages 305–334, Singapore, December 6–10, 2021. Springer, Cham, Switzerland.

[4] Bill Allombert, Jean-François Biasse, Jonathan Komada Eriksen, Péter Kutas, Chris Leonardi, Aurel Page, Renate Scheidler, and Márton Tot Bagi. PEARL-SCALLOP: Parameter extension applicable in real-life SCALLOP. In *IACR international conference on public-key cryptography*, 2025.

[5] Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. CSI-SharK: CSI-FiSh with sharing-friendly keys. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *ACISP 23: 28th Australasian Conference on Information Security and Privacy*, volume 13915 of *Lecture Notes in Computer Science*, pages 471–502, Brisbane, QLD, Australia, July 5–7, 2023. Springer, Cham, Switzerland.

[6] Karim Baghery, Daniele Cozzo, and Robi Pedersen. An isogeny-based ID protocol using structured public keys. In Maura B. Paterson, editor, *18th IMA International Conference on Cryptography and Coding*, volume 13129 of *Lecture Notes*

in Computer Science, pages 179–197, Virtual Event, December 14–15, 2021. Springer, Cham, Switzerland.

[7] Marco Baldi, Alessandro Barenghi, Luke Beckwith, Jean-Francois Biasse, Andre Esser, Kris Gaj, Kamyar Mohajerani, Gerardo Pelosi, Edoardo Persichetti, Markku-Juhani O. Saarinen, Paolo Santini, and Robert Wallace. LESS: Linear equivalence siganture scheme, 2023.

[8] Alessandro Barenghi, Jean-François Biasse, Tran Ngo, Edoardo Persichetti, and Paolo Santini. Advanced signature functionalities from the code equivalence problem. *International Journal of Computer Mathematics: Computer Systems Theory*, 7(2):112–128, 2022.

[9] Alessandro Barenghi, Jean-François Biasse, Edoardo Persichetti, and Paolo Santini. On the computational hardness of the code equivalence problem in cryptography. *Adv. Math. Commun.*, 17(1):23–55, 2023.

[10] Michele Battagliola, Giacomo Borin, Giovanni Di Crescenzo, Alessio Meneghetti, and Edoardo Persichetti. Enhancing threshold group action signature schemes: Adaptive security and scalability improvements. In Ruben Niederhagen and Markku-Juhani O. Saarinen, editors, *Post-Quantum Cryptography - 16th International Workshop, PQCrypto 2025, Taipei, Taiwan, April 8-10, 2025, Proceedings, Part I*, volume 15577 of *Lecture Notes in Computer Science*, pages 129–161. Springer, 2025.

[11] Michele Battagliola, Giacomo Borin, Alessio Meneghetti, and Edoardo Persichetti. Cutting the GRASS: Threshold GRoup action signature schemes. In Elisabeth Oswald, editor, *Topics in Cryptology – CT-RSA 2024*, volume 14643 of *Lecture Notes in Computer Science*, pages 460–489, San Francisco, CA, USA, May 6–9, 2024. Springer, Cham, Switzerland.

[12] Michele Battagliola, Giacomo Borin, Alessio Meneghetti, and Edoardo Persichetti. Cutting the GRASS: threshold group action signature schemes. In Elisabeth Oswald, editor, *Topics in Cryptology - CT-RSA 2024 - Cryptographers' Track at the RSA Conference 2024, San Francisco, CA, USA, May 6-9, 2024, Proceedings*, volume 14643 of *Lecture Notes in Computer Science*, pages 460–489. Springer, 2024.

[13] Benjamin Bencina, Alessandro Budroni, Jesús-Javier Chi-Domínguez, and Mukul Kulkarni. Properties of lattice isomorphism as a cryptographic group action. In Markku-Juhani O. Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Oxford, UK, June 12-14, 2024, Proceedings, Part I*, volume 14771 of *Lecture Notes in Computer Science*, pages 170–201. Springer, 2024.

[14] Ward Beullens. Not enough LESS: An improved algorithm for solving code equivalence problems over $\mathbb{F}_q$. In Orr Dunkelman, Michael J. Jacobson, Jr., and Colin O'Flynn, editors, *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography*, volume 12804 of *Lecture Notes in Computer Science*, pages 387–403, Halifax, NS, Canada (Virtual Event), October 21-23, 2020. Springer, Cham, Switzerland.

[15] Ward Beullens. Graph-theoretic algorithms for the alternating trilinear form equivalence problem. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 101–126, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[16] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part II*, volume 13276 of *Lecture Notes in Computer Science*, pages 95–126, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

[17] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 464–492, Daejeon, South Korea, December 7–11, 2020. Springer, Cham, Switzerland.

[18] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 227–247, Kobe, Japan, December 8–12, 2019. Springer, Cham, Switzerland.

[19] Jean-François Biasse, Giacomo Micheli, Edoardo Persichetti, and Paolo Santini. LESS is more: Code-based signatures without syndromes. In Abderrahmane Nitaj and Amr M. Youssef, editors, *AFRICACRYPT 20: 12th International Conference on Cryptology in Africa*, volume 12174 of *Lecture Notes in Computer Science*, pages 45–65, Cairo, Egypt, July 20–22, 2020. Springer, Cham, Switzerland.

[20] Gaetan Bisson. Computing endomorphism rings of elliptic curves under the GRH. *J. Math. Cryptol.*, 5(2):101–114, 2012.

[21] Markus Bläser, Zhili Chen, Dung Hoang Duong, Antoine Joux, Tuong Ngoc Nguyen, Thomas Plantard, Youming Qiao, Willy Susilo, and Gang Tang. On digital signatures based on group actions: QROM security and ring signatures.

In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 227–261, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland.

[22] Gilles Brassard and Moti Yung. One-way group actions. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 94–107. Springer, 1990.

[23] Alessandro Budroni, Jesús-Javier Chi-Domínguez, Giuseppe D'Alconzo, Antonio J. Di Scala, and Mukul Kulkarni. Don't use it twice! Solving relaxed linear equivalence problems. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part VIII*, volume 15491 of *Lecture Notes in Computer Science*, pages 35–65, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

[24] Alessandro Budroni, Jesús-Javier Chi-Domínguez, and Ermes Franch. Don't use it twice: Reloaded! on the lattice isomorphism group action. Cryptology ePrint Archive, Paper 2025/516, 2025.

[25] Wouter Castryck and Thomas Decru. CSIDH on the surface. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 111–129, Paris, France, April 15–17, 2020. Springer, Cham, Switzerland.

[26] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 423–447, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[27] Wouter Castryck, Marc Houben, Simon-Philipp Merz, Marzio Mula, Sam van Buuren, and Frederik Vercauteren. Weak instances of class group action based cryptography via self-pairings. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 762–792, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[28] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Cham, Switzerland.

[29] Mingjie Chen, Yi-Fu Lai, Abel Laval, Laurane Marco, and Christophe Petit. Malleable commitments from group actions and zero-knowledge proofs for circuits based on isogenies. In Anupam Chattopadhyay, Shivam Bhasin, Stjepan Picek, and Chester Rebeiro, editors, *Progress in Cryptology - INDOCRYPT 2023: 24th International Conference in Cryptology in India, Part I*, volume 14459 of *Lecture Notes in Computer Science*, pages 221–243, Goa, India, December 10–13, 2023. Springer, Cham, Switzerland.

[30] Mingjie Chen, Antonin Leroux, and Lorenz Panny. SCALLOP-HD: Group action from 2-dimensional isogenies. In Qiang Tang and Vanessa Teague, editors, *PKC 2024: 27th International Conference on Theory and Practice of Public Key Cryptography, Part III*, volume 14603 of *Lecture Notes in Computer Science*, pages 190–216, Sydney, NSW, Australia, April 15–17, 2024. Springer, Cham, Switzerland.

[31] Clémence Chevignard, Guilhem Mureau, Thomas Espitau, Alice Pellet-Mary, Heorhii Pliatsok, and Alexandre Wallet. A reduction from hawk to the principal ideal problem in a quaternion algebra. *IACR Cryptol. ePrint Arch.*, page 287, 2025.

[32] Andrew M. Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *J. Mathematical Crytology*, 8(1):1–29, 2014.

[33] Andrew M Childs and Wim Van Dam. Quantum algorithm for a generalized hidden shift problem. *arXiv preprint quant-ph/0507190*, 2005.

[34] Tung Chou, Ruben Niederhagen, Edoardo Persichetti, Lars Ran, Tovohery Hajatiana Randrianarisoa, Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Matrix equivalence digital signature, 2023.

[35] Tung Chou, Ruben Niederhagen, Edoardo Persichetti, Tovohery Hajatiana Randrianarisoa, Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Take your meds: digital signatures from matrix code equivalence. In *International conference on cryptology in Africa*, pages 28–52. Springer, 2023.

[36] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Paper 2006/291, 2006.

[37] Alain Couvreur, Thomas Debris-Alazard, and Philippe Gaborit. On the hardness of code equivalence problems in rank metric. *CoRR*, abs/2011.04611, 2020.

[38] David A. Cox. *Primes of the form $x^2 + ny^2$: Fermat, Class Field Theory, and Complex Multiplication*. AMS Chelsea Publishing/American Mathematical Society, third edition, 2022.

[39] Giuseppe D'Alconzo and Antonio J Di Scala. Representations of group actions and their applications in cryptography. *Finite Fields and Their Applications*, 99:102476, 2024.

[40] Giuseppe D'Alconzo, Andrea Flamini, and Andrea Gangemi. Non-interactive commitment from non-transitive group actions. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part VII*, volume 14444 of *Lecture Notes in Computer Science*, pages 222–252, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

[41] Giuseppe D'Alconzo, Andrea Flamini, Alessio Meneghetti, and Edoardo Signorini. A framework for group action-based multi-signatures and applications to less, meds, and ALTEQ. *IACR Cryptol. ePrint Arch.*, page 1691, 2024.

[42] Giuseppe D'Alconzo, Andrea Flamini, Alessio Meneghetti, and Edoardo Signorini. A framework for group action-based multi-signatures and applications to less, meds, and ALTEQ. In Tibor Jager and Jiaxin Pan, editors, *Public-Key Cryptography - PKC 2025 - 28th IACR International Conference on Practice and Theory of Public-Key Cryptography, Røros, Norway, May 12-15, 2025, Proceedings, Part II*, volume 15675 of *Lecture Notes in Computer Science*, pages 99–133. Springer, 2025.

[43] Pierrick Dartois, Jonathan Komada Eriksen, Tako Boris Fouotsa, Arthur Herlédan Le Merdy, Riccardo Invernizzi, Damien Robert, Ryan Rueger, Frederik Vercauteren, and Benjamin Wesolowski. PEGASIS: Practical effective class group action using 4-dimensional isogenies. In *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference*, 2025.

[44] Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 345–375, Atlanta, GA, USA, May 7–10, 2023. Springer, Cham, Switzerland.

[45] Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 365–394, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Cham, Switzerland.

[46] Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of Public*

Key Cryptography, Part II, volume 12111 of *Lecture Notes in Computer Science*, pages 187–212, Edinburgh, UK, May 4–7, 2020. Springer, Cham, Switzerland.

[47] Cyprien Delpech de Saint Guilhem and Robi Pedersen. New proof systems and an OPRF from CSIDH. In Qiang Tang and Vanessa Teague, editors, *PKC 2024: 27th International Conference on Theory and Practice of Public Key Cryptography, Part III*, volume 14603 of *Lecture Notes in Computer Science*, pages 217–251, Sydney, NSW, Australia, April 15–17, 2024. Springer, Cham, Switzerland.

[48] Whitfield Diffie and Martin E. Hellman. *New Directions in Cryptography*, page 365–390. Association for Computing Machinery, New York, NY, USA, 1 edition, 2022.

[49] Javad Doliskani. On division polynomial PIT and supersingularity. *Appl. Algebra Eng. Commun. Comput.*, 29(5):393–407, 2018.

[50] Léo Ducas, Eamonn W. Postlethwaite, Ludo N. Pulles, and Wessel P. J. van Woerden. Hawk: Module LIP makes lattice signatures fast, compact and simple. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part IV*, volume 13794 of *Lecture Notes in Computer Science*, pages 65–94, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

[51] Léo Ducas and Wessel van Woerden. On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 643–673. Springer, 2022.

[52] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Group action key encapsulation and non-interactive key exchange in the QROM. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part II*, volume 13792 of *Lecture Notes in Computer Science*, pages 36–66, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

[53] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Generic models for group actions. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 406–435, Atlanta, GA, USA, May 7–10, 2023. Springer, Cham, Switzerland.

[54] Dung Hoang Duong, Xuan Thanh Khuc, Youming Qiao, Willy Susilo, and Chuanqi Zhang. Blind signatures from cryptographic group actions. Cryptology ePrint Archive, Paper 2025/397, 2025.

[55] Dung Hoang Duong, Xuan Thanh Khuc, Youming Qiao, Willy Susilo, and Chuanqi Zhang. Blind signatures from cryptographic group actions. Cryptology ePrint Archive, Paper 2025/397, 2025.

[56] Luca De Feo. Mathematics of isogeny based cryptography. *CoRR*, abs/1711.04062, 2017.

[57] Joshua A. Grochow and Youming Qiao. On the complexity of isomorphism problems for tensors, groups, and polynomials I: tensor isomorphism-completeness. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPIcs*, pages 31:1–31:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[58] Helmut Hasse. Zur theorie der abstrakten elliptischen funktionenkörper iii. die struktur des meromorphismenrings. die riemannsche vermutung. 1936.

[59] Minki Hhan, Tomoyuki Morimae, and Takashi Yamakawa. From the hardness of detecting superpositions to cryptography: Quantum public key encryption and commitments. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part I*, volume 14004 of *Lecture Notes in Computer Science*, pages 639–667, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[60] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34, Tapei, Taiwan, November 29 – December 2 2011. Springer Berlin Heidelberg, Germany.

[61] Zhengfeng Ji, Youming Qiao, Fang Song, and Aaram Yun. General linear group action on tensors: A candidate for post-quantum cryptography. In *Theory of cryptography conference*, pages 251–281. Springer, 2019.

[62] Kaijie Jiang, Anyu Wang, Hengyi Luo, Guoxiao Liu, Tang Gang, Yanbin Pan, and Xiaoyun Wang. Re-randomize and extract: A novel commitment construction framework based on group actions. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part II*, volume 15602 of *Lecture Notes in Computer Science*, pages 124–153. Springer, 2025.

[63] Antoine Joux. MPC in the head for isomorphisms and group actions. *IACR Cryptol. ePrint Arch.*, page 664, 2023.

[64] Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. CSI-Otter: Isogeny-based (partially) blind signatures from the class group action with a twist.

In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 729–761, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[65] D. R. Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.

[66] Veronika Kuchta, Jason T. LeGrow, and Edoardo Persichetti. Post-quantum blind signatures from matrix code equivalence. *IACR Cryptol. ePrint Arch.*, page 274, 2025.

[67] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.

[68] Yi-Fu Lai. Capybara and tsubaki: Verifiable random functions from group actions and isogenies. *IACR Commun. Cryptol.*, 1(3):1, 2024.

[69] Yi-Fu Lai. A note on isogeny group action-based pseudorandom functions. Cryptology ePrint Archive, Paper 2024/2042, 2024.

[70] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. Compact, efficient and UC-secure isogeny-based oblivious transfer. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 213–241, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland.

[71] Jeffrey Leon. Computing automorphism groups of error-correcting codes. *IEEE Transactions on Information Theory*, 28(3):496–511, 2003.

[72] Antonin Leroux. *Quaternion algebras and isogeny-based cryptography*. PhD thesis, École Polytechnique, 2022.

[73] Antonin Leroux and Maxime Roméas. Updatable encryption from group actions. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II*, pages 20–53, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland.

[74] Yinan Li and Youming Qiao. Linear algebraic analogues of the graph isomorphism problem and the erdős-Rényi model. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 463–474, Berkeley, CA, USA, October 15–17, 2017. IEEE Computer Society Press.

[75] Valeria Loscri, Luca Chiaraviglio, and Anna Maria Vegni. The road towards 6g: Opportunities, challenges, and applications. *A Comprehensive View of the Enabling Technologies*, 2024.

[76] Hengyi Luo, Kaijie Jiang, Yanbin Pan, and Anyu Wang. Commitment schemes based on module-LIP. Cryptology ePrint Archive, Paper 2025/431, 2025.

[77] Joseph Macula and Katherine E. Stange. Extending class group action attacks via sesquilinear pairings. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 371–395, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

[78] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 448–471, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[79] Ian McQuoid and Jiayu Xu. An efficient strong asymmetric PAKE compiler instantiable from group actions. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part VIII*, volume 14445 of *Lecture Notes in Computer Science*, pages 176–207, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

[80] Jonas Meers and Doreen Riepel. CCA secure updatable encryption from non-mappable group actions. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 137–169, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland.

[81] Michele Mosca and Marco Piani. Quantum threat timeline report 2024. evolutionQ, 2024. https://globalriskinstitute.org/publication/2024-quantum-threat-timeline-report/.

[82] National institute of standards and technology. Post-quantum cryptography standardization, 2016. https://csrc.nist.gov/Projects/cryptographic-standards-and-guidelines.

[83] Jonas Nick, Tim Ruffing, and Yannick Seurin. MuSig2: Simple two-round Schnorr multi-signatures. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part I*, volume 12825 of *Lecture Notes in Computer Science*, pages 189–221, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.

[84] Aurel Page and Damien Robert. Introducing clapoti(s): Evaluating the isogeny class group action in polynomial time. Cryptology ePrint Archive, Paper 2023/1766, 2023.

[85] Lorenz Panny, Christophe Petit, and Miha Stopar. KLaPoTi: An asymptotically efficient isogeny group action from 2-dimensional isogenies. Cryptology ePrint Archive, Paper 2024/1844, 2024.

[86] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.

[87] Oded Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space, 2004.

[88] Krijn Reijnders, Simona Samardjiska, and Monika Trimoska. Hardness estimates of the code equivalence problem in the rank metric. *Designs, Codes and Cryptography*, 92(3):833–862, 2024.

[89] Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 472–503, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[90] Damien Robert. Fast pairings via biextensions and cubical arithmetic. Cryptology ePrint Archive, Paper 2024/517, 2024.

[91] Alexander Rostovtsev and Anton Stolbunov. PUBLIC-KEY CRYPTOSYSTEM BASED ON ISOGENIES. Cryptology ePrint Archive, Paper 2006/145, 2006.

[92] René Schoof. *Nonsingular plane cubic curves over finite fields*. J. Comb. Theory, 1987.

[93] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[94] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Springer-Verlag, New York, NY, 2. aufl. edition, 2009.

[95] Andrew V. Sutherland. Identifying supersingular elliptic curves. *LMS Journal of Computation and Mathematics*, 15:317–325, 2012.

[96] Andrew V. Sutherland. Isogeny volcanoes. *CoRR*, abs/1208.5370, 2012.

[97] Gang Tang, Dung Hoang Duong, Antoine Joux, Thomas Plantard, Youming Qiao, and Willy Susilo. Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 582–612, Trondheim, Norway, May 30 – June 3, 2022. Springer, Cham, Switzerland.

[98] William C. Waterhouse. *Abelian varieties over finite fields*. Annales scientifiques de l'École Normale Supérieure, 1969.

[99] Mark Zhandry. Quantum money from abelian group actions. In Venkatesan Guruswami, editor, *ITCS 2024: 15th Innovations in Theoretical Computer Science Conference*, volume 287, pages 101:1–101:23, Berkeley, CA, USA, January 30 – February 2, 2024. Leibniz International Proceedings in Informatics (LIPIcs).