PEGASIS: Practical Effective Class Group Action using 4-Dimensional Isogenies

Pierrick Dartois[1,2(✉)], Jonathan Komada Eriksen[5], Tako Boris Fouotsa[3], Arthur Herlédan Le Merdy[4], Riccardo Invernizzi[5], Damien Robert[1,2], Ryan Rueger[6,7], Frederik Vercauteren[5], and Benjamin Wesolowski[4]

[1] University of Bordeaux, CNRS, Bordeaux INP, IMB, UMR 5251, 33400 Talence, France
pierrick.dartois@u-bordeaux.fr
[2] INRIA, IMB, UMR 5251, 33400 Talence, France
damien.robert@inria.fr
[3] EPFL, LASEC, Lausanne, Switzerland
tako.fouotsa@epfl.ch
[4] ENS de Lyon, CNRS, UMPA, UMR 5669, Lyon, France
{arthur.herledan_le_merdy,benjamin.wesolowski}@ens-lyon.fr
[5] KU Leuven, COSIC, Heverlee, Belgium
{frederik.vercauteren,riccardo.invernizzi}@esat.kuleuven.be
[6] IBM Research Europe, Zurich, Switzerland
ryan@rueg.re
[7] Technische Universität München, Munich, Germany

Abstract. In this paper, we present the first practical algorithm to compute an effective group action of the class group of any imaginary quadratic order $\mathcal{O}$ on a set of supersingular elliptic curves primitively oriented by $\mathcal{O}$. Effective means that we can act with any element of the class group directly, and are not restricted to acting by products of ideals of small norm, as for instance in CSIDH. Such restricted effective group actions often hamper cryptographic constructions, e.g. in signature or MPC protocols.

Our algorithm is a refinement of the Clapoti approach by Page and Robert, and uses 4-dimensional isogenies. As such, it runs in polynomial time, does not require the computation of the structure of the class group, nor expensive lattice reductions, and our techniques allow it to be instantiated with the orientation given by the Frobenius endomorphism. This makes the algorithm practical even at security levels as high as CSIDH-4096. Our implementation in SageMath takes 1.5 s to compute a group action at the CSIDH-512 security level, 21 s at CSIDH-2048 level and around 2 min at the CSIDH-4096 level. This marks the first instantiation of an effective cryptographic group action at such high security levels. For comparison, the recent KLaPoTi approach requires around 200 s at the CSIDH-512 level in SageMath and 2 s in Rust.

Y. Tauman Kalai and S. F. Kamara (Eds.): CRYPTO 2025, LNCS 16000, pp. 67–99, 2025.
https://doi.org/10.1007/978-3-032-01855-7_3

1 Introduction

The simplicity and flexibility of the Diffie-Hellman key-agreement protocol [25] has made it ideally suited to create more advanced protocols, such as public-key encryption [28], digital signatures [53], password-authenticated key exchange [35], threshold encryption [24], threshold signatures [33,34], updatable encryption [9] and many more. These protocols are all based on the computational efficiency and commutativity of group exponentiation, combined with the computational hardness of the discrete logarithm problem.

Due to Shor's algorithm [54], the discrete logarithm problem is not quantum hard. To construct a post-quantum analogue of the Diffie-Hellman protocol, we need to remove the underlying group structure. One way to achieve this is to use a commutative group action instead. The vectorisation and parallelisation problems, which are analogues of the discrete-logarithm and computational Diffie-Hellman problems respectively, are assumed to be (superpolynomially, but sub-exponentially [38]) hard, even for quantum computers. Furthermore, both problems are equivalently hard for quantum computers [31,32,41]. Using group actions, it is possible to construct post-quantum analogues of the advanced protocols given above: public-key encryption [42], digital signatures [7], password-authenticated key-exchange [1], threshold encryption and signatures [23] and updatable encryption [39].

A group action is said to be *effective* (EGA) if, amongst other conditions, the action can be computed in polynomial time [2]. That is, if G acts on X, we have a polynomial-time algorithm that computes gx for any g in G and x in X. If we only have a polynomial-time algorithm that can compute $g_i x$ for $\{g_1, \ldots, g_n\}$, a set of (polynomially many) generators of G, the action is said to be a *restricted effective* group action (REGA).

The most prominent instantiation of a REGA is given by the action of the class group of an imaginary quadratic order on a certain set of oriented elliptic curves, with the CSIDH protocol [12] (or its variant CSURF [11]) being the most efficient. In CSIDH, the orientation is given by the Frobenius endomorphism, and the set of oriented curves simply consists of the supersingular curves defined over $\mathbb{F}_p$, up to $\mathbb{F}_p$-isomorphism. Since we can only efficiently evaluate the action of ideals of smooth norm, CSIDH is indeed a REGA and not an EGA. This often results in difficulties in constructing post-quantum variants of protocols based on Diffie-Hellman: many of the previously mentioned examples really require an EGA, while other examples, such as the signature scheme SeaSign [21], resort to some form of rejection sampling to not leak information about the secret (as a workaround for being a REGA).

A three step approach to turn a REGA into an EGA was given by the signature scheme CSI-FiSh [7]: the first step computes the structure of the class group (which can be done in quantum polynomial time, but classically takes subexponential time), the second step computes a (very) short basis of the relation lattice (which takes exponential time, even quantumly) and the third step expresses any element as a product of the generators with small exponents, using the short basis. Step 1 has been done in practice for CSIDH-512, but fur-

ther research [6,8,48] concludes that using a 512 bit prime for CSIDH does not achieve NIST level 1 security, with the latter two sources claiming that primes of more than 2000 bits are required. Since the computation of the class group is infeasible for such large primes, the primitive SCALLOP [20] and improvements [3,13] avoided such computation by using a (large prime conductor) suborder of an order with small discriminant, so that the class group structure comes for free. This approach however does not fundamentally solve steps 2 and 3 (see https://yx7.cc/blah/2023-04-14.html for an informal, but detailed discussion of this issue), and the highest level achieved in [3] is equivalent to CSIDH-1536, for which a single group action takes almost 12 min using a `C++` implementation. This approach is therefore completely hopeless to instantiate practical effective group actions at higher security levels (even using a quantum computer).

Our Contributions. We describe a practical algorithm to compute the group action of any element in the class group of an imaginary quadratic order $\mathcal{O}$ on a set of supersingular elliptic curves over $\overline{\mathbb{F}}_p$ primitively oriented by $\mathcal{O}$. Our algorithm works for all orders of discriminant $\Delta_\mathcal{O}$ all the way up to $|\Delta_\mathcal{O}| \approx p$ and in particular, applies in the CSIDH/CSURF setting, where the orientation is given by Frobenius, and we have $\Delta_\mathcal{O} = -p$ or $\Delta_\mathcal{O} = -4p$. As such, we obtain the first EGA at security levels equivalent to CSIDH-2048 and CSIDH-4096. We implement the algorithm in SageMath, and show that it achieves a highly practical runtime, which scales well to higher security levels. Not only does our algorithm allow us to easily instantiate security levels far beyond what previous EGA instantiations were able to; the timings of our proof-of-concept SageMath implementation also outperform all other EGA instantiations (even compared to their highly optimized `C++`/Rust implementations) at corresponding security levels. Our implementation is publicly available at https://github.com/pegasis4d.

Technical Overview. Our work is based on the Clapoti framework [46], and proceeds by embedding the isogeny we wish to evaluate in an isogeny in dimension 4. In order to evaluate the action given by the ideal $\mathfrak{a} \subseteq \mathcal{O}$ on an elliptic curve E using only one 4-dimensional 2^e-isogeny, the Clapoti framework requires finding two other integral ideals $\mathfrak{b}, \mathfrak{c}$ equivalent to $\mathfrak{a}$ such that

$$uN(\mathfrak{b}) + vN(\mathfrak{c}) = 2^e, \tag{1}$$

with the restriction that u, v are integers that can be written as the sum of two squares (in particular, they must be easy to factorise), and where $2^e < p$ (or more accurately, the full torsion group $E[2^e]$ should be defined over $\mathbb{F}_{p^{2k}}$ for a small k).

Several issues arise when trying to solve the above equation for $|\Delta_\mathcal{O}| \approx p$: even without the restriction on u and v being sums of squares, sometimes the equation simply has no solution. Indeed, since the smallest ideals equivalent to $\mathfrak{a}$ are already expected to have norm around $\sqrt{p}$, and a solution is guaranteed to exist only when $N(\mathfrak{b})N(\mathfrak{c}) < 2^e$, this contradicts the requirement $2^e < p$.

As such, imposing the extra condition for both u and v to be sums of squares further lowers the probability of finding a solution.

The main insight is that we can derive a related equation which is much easier to solve by exploiting the fact we can always efficiently compute small degree isogenies. We use this insight in two ways: first, we write $\mathfrak{b} = \mathfrak{b}_k\mathfrak{b}_e$ and $\mathfrak{c} = \mathfrak{c}_k\mathfrak{c}_e$ where $\mathfrak{b}_e$ and $\mathfrak{c}_e$ are ideals of smooth norm and compute the isogenies $\varphi_{\mathfrak{b}_e} : E \to E_1$ and $\varphi_{\mathfrak{c}_e} : E \to E_2$, simply using Elkies' algorithm [29] or Vélu's algorithm [55]. We can then apply the Clapoti framework to $E_1 \times E_2$ instead of $E \times E$, resulting in the equation

$$uN(\mathfrak{b}_k) + vN(\mathfrak{c}_k) = 2^e,$$

where now the product $N(\mathfrak{b}_k)N(\mathfrak{c}_k)$ is typically much smaller than 2^e since we have removed the norms of the ideals $\mathfrak{b}_e$ and $\mathfrak{c}_e$.

Second, we use the same insight to relax the conditions on u and v: instead of requiring that each can be written as a sum of squares, we again allow to take out smooth factors g_u and g_v such that $u = g_u(x_u^2+y_u^2)$ and $v = g_v(x_v^2+y_v^2)$. Note that integers containing prime factors (to an odd power) which are 3 mod 4 can never be written as a sum of squares, so allowing factors like $3, 7, 11, 19, \ldots$ to be taken out of u and v drastically increases the probability that both u/g_u and v/g_v are sums of squares. Applying the above, allows us to solve the (relaxed) norm equation efficiently, even at the highest security levels.

With u and v of this form, we are able to efficiently construct a 2-dimensional u-isogeny $\Phi_u : E_1^2 \to E_u^2$ and a 2-dimensional v-isogeny $\Phi_v : E_2^2 \to E_v^2$. We then compute a 4-dimensional 2^e-isogeny $F : E_u^2 \times E_v^2 \to E_{\mathfrak{a}}^2 \times E'^2$ that "embeds" Φ_u, Φ_v, $\varphi_{\mathfrak{b}_k} : E_1 \to E_{\mathfrak{a}}$ and $\varphi_{\mathfrak{c}_k} : E_2 \to E_{\mathfrak{a}}$, where $E_{\mathfrak{a}} := \mathfrak{a} \cdot E$. Using level-2 theta coordinates as in [17], the computation of F is relatively fast. We finally extract the resulting curve $E_{\mathfrak{a}}$ from the codomain of F. For very high security levels, we remark that it is helpful to relax the conditions on u, v and allow them to be arbitrary positive integers, at the cost of using more dimension-4 isogenies (see Sects. 4.2 and 6.4).

Comparison with Related Work. We briefly compare our algorithm with 2 related works: the original Clapoti framework [46] and the recent KLaPoTi algorithm [47] that relies on KLPT.

Clapoti: As described above our work is based on the Clapoti framework [46], which presents the first polynomial time algorithm to act with a random ideal $\mathfrak{a} \subseteq \mathcal{O}$ by finding two integral ideals $\mathfrak{b}, \mathfrak{c}$ equivalent to $\mathfrak{a}$ such that

$$uN(\mathfrak{b}) + vN(\mathfrak{c}) = M\,. \tag{2}$$

For a provably polynomial time algorithm, M has to be chosen powersmooth, but in order to achieve a practical algorithm, one really is limited to $M = 2^e$ where the 2^e-torsion is easily accessible, in particular, $2^e < p$. However, the expected number of solutions is $2^e/N(\mathfrak{b})N(\mathfrak{c})$ where $N(\mathfrak{b}), N(\mathfrak{c})$ are roughly $\sqrt{p}$ and thus if $p + 1 = f2^e$ even for small f, this equation simply will not have

a solution. To illustrate this: for the 4000-bit parameter set, directly applying Clapoti will fail in 97% of the cases when $M = 2^e < p$, and this is without the restriction on u and v being sums of squares. If one adds this restriction on u and v, the probability to find a solution is close to 0 since the probability that a pair (u, v) each of size $\sqrt{p}$ is a sum of squares is proportional to $1/\log(p)$. Furthermore, note that it is impossible to test all pairs (u, v) since this would involve factorisation, even further lowering the probability by another 2 to 3 orders of magnitude depending on the size of p. In conclusion: running Clapoti with $M = 2^e < p$ will almost always fail, and thus is not a practically efficient algorithm.

KLaPoti: A recent paper by Panny, Petit and Stopar [47] has also instantiated a primitive KLaPoTi based on Clapoti. KLaPoTi differs from PEGASIS in several aspects. The main ones are the way Eq. (1) is solved, the dimension of the isogeny used in the evaluation of the group action and the size of the base prime p. KLaPoTi relies on the KLPT algorithm [37] to solve Eq. (1), which now requires u, v to be perfect squares. The fact that u and v are perfect squares allows to perform the group action computation using dimension-2 isogenies only. Nevertheless, the relatively large size of the solution obtained by this KLPT approach requires $|\Delta|^3 < 2^e$ where Δ is the discriminant of the order $\mathcal{O}$. This implies that KLaPoTi uses a base prime p whose bitsize is 3 times larger than the one used in CSIDH and PEGASIS, and uses an order $\mathcal{O}$ of discriminant Δ with $|\Delta|^3 < 2^e < p$. This highly affects the performance and the key sizes in KLaPoTi. Even though PEGASIS performs the group action using 4-dimensional isogenies, the fact that $|\Delta| \approx 2^e \approx p$ in PEGASIS enables a group action computation which is more efficient. For example, to achieve similar security to CSIDH-512, KLaPoTi uses a prime p of 1536 bits and a single group action computation takes approximately 3 min [47, Section 7.1] in SageMath, while PEGASIS uses a prime of about 512 bits and a single group action computation takes approximately 1.5 s.

Organisation of Paper. The rest of the paper is organized as follows. In Sect. 2 we recall the necessary background material. In Sect. 3 we describe our whole algorithm for general orientations. In Sect. 4, we detail an important subroutine, namely how to compute an isogeny of prescribed degree in dimension 2, before we in Sect. 5 specialise our algorithm for the CSIDH orientation. Finally, in Sect. 6 we detail some implementation choices before we present our timings.

Full Version of the Paper. Due to space limitations, this is an abbreviated version of the full version of the paper [18], which contains the missing proofs and an extended appendix with more details on theta coordinates and 4D isogenies; an extra variant of the algorithm; working exclusively over $\mathbb{F}_p$ and an in depth explanation on how parameters were chosen.

2 Preliminaries

For a general introduction to isogenies, especially in the context of cryptography, we refer the reader to [30].

2.1 Isogeny Class-Group Action on Oriented Curves

Classic results [56, Th. 4.5] tell us that the class-group $\mathrm{Cl}(\mathcal{O})$ of an imaginary quadratic order $\mathcal{O}$ acts freely and transitively on the set $\mathrm{Ord}_q(\mathcal{O})$ of *ordinary* elliptic curves $E/\mathbb{F}_q$ with endomorphism ring $\mathrm{End}(E) \cong \mathcal{O}$. In [14], generalising the work of [11,12], Colò and Kohel introduced the framework for a similar action, in which $\mathrm{Cl}(\mathcal{O})$ acts on a set of *supersingular* elliptic curves E which contain $\mathcal{O}$ as a subring of $\mathrm{End}(E)$. We briefly recount this general construction, together with results from [45].

Let $\mathcal{O} = \mathbb{Z}[\alpha]$ be an imaginary quadratic order and $E/\mathbb{F}_{p^n}$ an elliptic curve. We call an injective morphism of rings $\iota\colon \mathcal{O} \xhookrightarrow{\mathrm{End}} (E)$ an $\mathcal{O}$-*orientation* and say it is *primitive* if $\iota(\alpha) = \omega$ is a primitive element of $\mathrm{End}(E)$ viewed as a lattice, *i.e.* ω does not factor through a (non-trivial) scalar multiplication. We call the pair (E, ι) a (primitively) $\mathcal{O}$-oriented elliptic curve.

Let (E, ι) be primitively $\mathcal{O}$-oriented and $\mathfrak{a}$ an integral invertible ideal of $\mathcal{O}$ with norm coprime to p. Let $\varphi_{\mathfrak{a}}\colon E \to E_{\mathfrak{a}}$ be the isogeny with kernel

$$E[\mathfrak{a}] = \bigcap_{\sigma \in \mathfrak{a}} \ker(\iota(\sigma))$$

and degree $\mathrm{N}(\mathfrak{a})$. Now consider the map $\mathcal{O} \to \mathrm{End}(E_{\mathfrak{a}}) : \gamma \mapsto \varphi_{\mathfrak{a}}\iota(\gamma)\widehat{\varphi_{\mathfrak{a}}}$. It is an injective morphism of additive groups, but sends $1 \mapsto [\deg(\varphi_{\mathfrak{a}})]$ and so is not a morphism of rings. We fix this by first noting that $\varphi_{\mathfrak{a}}\iota(\gamma)\widehat{\varphi_{\mathfrak{a}}}$ always factors through $[\deg(\varphi_{\mathfrak{a}})]$ and then defining $(\varphi_{\mathfrak{a}})_*(\iota)$ as the map given by $(\varphi_{\mathfrak{a}})_*(\iota)(\gamma) = \varphi_{\mathfrak{a}}\iota(\gamma)\widehat{\varphi_{\mathfrak{a}}}/[\deg(\varphi_{\mathfrak{a}})]$ to obtain a morphism of rings and thus a well-defined $\mathcal{O}$-orientation. Indeed, by diagram-chasing one verifies that $\ker(\varphi_{\mathfrak{a}})$ is invariant under $\omega = \iota(\alpha)$, demonstrating that $\varphi_{\mathfrak{a}}\iota(\gamma)\widehat{\varphi_{\mathfrak{a}}}(E[\deg(\varphi_{\mathfrak{a}})]) = \{0\}$ for all γ in $\mathbb{Z}[\alpha]$, and that $\varphi_{\mathfrak{a}}\iota(\gamma)\widehat{\varphi_{\mathfrak{a}}}$ factors through $[\deg(\varphi_{\mathfrak{a}})]$. In fact, $(\varphi_{\mathfrak{a}})_*(\iota)$ is a *primitive* orientation [45, Prop. 3.5].

We say that the isogenies $\varphi_{\mathfrak{a}}\colon (E, \iota) \to (E_{\mathfrak{a}}, (\varphi_{\mathfrak{a}})_*(\iota))$ are $\mathcal{O}$-*oriented*. Likewise, an isomorphism $\varphi\colon E \xrightarrow{\sim} E'$ is called an $\mathcal{O}$-*isomorphism* $(E, \iota) \to (E', \iota')$ if $\iota' = \varphi_*(\iota) := (\gamma \mapsto \varphi\iota(\gamma)\hat{\varphi})$. Using this terminology, we define $\mathrm{SS}^{\mathrm{pr}}_{p^n}(\mathcal{O})$ to be the $\mathcal{O}$-isomorphism classes of primitively $\mathcal{O}$-oriented supersingular elliptic curves defined over $\mathbb{F}_{p^n}$. This set is non-empty if and only if p is not split in $K = \mathcal{O} \otimes_{\mathbb{Z}} \mathbb{Q}$ and does not divide the conductor of $\mathcal{O}$ in $\mathcal{O}_K$ [45, Prop. 3.2].

We note that every class $[\mathfrak{a}]$ in the class-group $\mathrm{Cl}(\mathcal{O})$ is represented by an invertible integral ideal $\mathfrak{b}$ with norm $\mathrm{N}(\mathfrak{b})$ coprime to p [16, Cor. 7.17]. Moreover, it is clear that principal ideals $\gamma\mathcal{O}$ correspond to endomorphisms $\varphi_{\gamma\mathcal{O}} = \iota(\gamma)$. Now the map

$$\mathrm{Cl}(\mathcal{O}) \times \mathrm{SS}^{\mathrm{pr}}_p(\mathcal{O}) \to \mathrm{SS}^{\mathrm{pr}}_p(\mathcal{O}) \qquad [\mathfrak{a}], (E, \iota) \mapsto \mathfrak{a} \cdot (E, \iota) := (E_{\mathfrak{a}}, (\varphi_{\mathfrak{a}})_*(\iota))$$

is well-defined; and when $\mathrm{SS}_p^{\mathrm{pr}}(\mathcal{O})$ is non-empty, the action is free with at most two orbits [45, Prop. 3.3, Th. 3.4]. Restricting to one of these orbits, we obtain a free and transitive group action.

The vectorisation problem of this action - given $(E, \iota), (E', \iota')$ compute $[\mathfrak{a}]$ such that $\mathfrak{a} \cdot (E, \iota) = (E', \iota')$ - is thought to be quantum resistant. This property led to several cryptographic constructions based on ideal class group actions [12–14,20] which define (restricted) effective group actions, as mentioned in Sect. 1. The first one to be introduced was CSIDH (Commutative Supersingular Isogeny Diffie-Hellman)[1] [12] where $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$ and the orientation induces an isomorphism $\mathcal{O} \xrightarrow{\sim} \mathrm{End}_{\mathbb{F}_p}(\mathcal{O})$, mapping $\sqrt{-p}$ to the Frobenius endomorphism of $(x, y) \mapsto (x^p, y^p)$ for all $E \in \mathrm{SS}_p^{\mathrm{pr}}(\mathcal{O})$. Even though these schemes are thought to be quantum resistant, the best known quantum attack against the underlying vectorisation problem is a subexponential algorithm due to Kuperberg [38]. For that reason, there is an uncertainty on the parameters required to ensure a sufficient security level for CSIDH (and other class group action schemes). As mentioned in Sect. 1, conservative sources [6,8,48] estimate that a 2000 bit prime p is required to ensure a NIST-1 security level for CSIDH.

2.2 Polarised Isogenies Between Abelian Varieties

Every abelian variety A has a *dual* $\widehat{A}$ of the same dimension and every isogeny $\varphi\colon A \to B$ has a *dual* $\widehat{\varphi}\colon \widehat{B} \to \widehat{A}$ of the same degree. If there exists a special kind of isogeny $\lambda_A\colon A \to \widehat{A}$ called a *polarisation*, we say (A, λ_A) is a *polarised* abelian variety (PAV). When λ_A is an isomorphism, we say it is a *principal* polarisation and call A a *principally* polarised abelian variety (PPAV). Elliptic curves are exactly the 1-dimensional PPAVs; they are canonically principally polarised.

Definition 2.1 (Polarised Isogenies). *Let $\varphi\colon (A, \lambda_A) \to (B, \lambda_B)$ be an isogeny between PAVs. If there exists an integer d such that $\widehat{\varphi}\lambda_B\varphi = [d]\lambda_A$, we say that φ is* (λ_A, λ_B)-polarised *with* polarised degree $\deg_{\mathrm{p}}(\varphi) = d$. *When A, B are PPAVs we define the* polarised dual *of φ to be $\widetilde{\varphi} = \lambda_A^{-1}\widehat{\varphi}\lambda_B\colon (B, \lambda_B) \to (A, \lambda_A)$. Then φ is (λ_A, λ_B)-polarised with degree d if $\widetilde{\varphi}\varphi = [d]$ in $\mathrm{End}(A)$.*

(λ_A, λ_B)-polarised isogenies of polarised degree d are characterised by an isotropy condition of their kernel.

Lemma 2.1 *Let $\varphi\colon (A, \lambda_A) \to (B, \lambda_B)$ be a (λ_A, λ_B)-polarised isogeny with polarised degree d such that $p \nmid d$ and let $g := \dim(A)$. Then $\ker(\varphi)$ is a subgroup of cardinality d^g of $A[d]$, isotropic with respect to the Weil pairing $e^{[d]\lambda_A}$ associated to the polarisation $[d]\lambda_A : A \to \widehat{A}$ (as defined in [40, p. 135]).*

Conversely, if A is a PPAV of dimension g and $\varphi\colon A \to B$ has kernel $\ker(\varphi) \subseteq A[d]$ of cardinality d^g and isotropic for $e^{[d]\lambda_A}$, then there exists a unique principal polarisation λ_B on B such that $\varphi : A \to B$ is a (λ_A, λ_B)-polarised isogeny with polarised degree d.

[1] Note that CSIDH is based on the CRS protocol [15,52], which uses the full endomorphism ring of an ordinary curve instead of an orientation of a supersingular curve.

Proof The proof is given in the full version of the paper [18]. □

For brevity, we will drop mention of the polarisations when they are clear from context, and say *d-isogeny* to mean an isogeny with polarised degree d. In particular, it is implicit that the corresponding polarisation to a PAV denoted A is λ_A. Between elliptic curves, the principal polarisation being canonical, the dual $\widehat{\varphi}$ and polarised dual $\widetilde{\varphi}$ of an isogeny can be identified and will both be denoted by $\widehat{\varphi}$; and the notions of degree and polarised degree are also the same.

We recall a standard result about factoring isogenies between PPAVs, which we will need to recall the Clapoti construction in the more general context of [46, Remark 2.10].

Lemma 2.2 (Factoring Isogenies between PPAVs). *Let A, B be PPAVs defined over a field k and $\varphi\colon A \to B$ be a d-isogeny between them. For every pair of positive coprime integers d_1, d_2 each being coprime to* $\mathrm{char}(k)$ *satisfying $d_1 d_2 = d$, there exists a PPAV C, a d_1-isogeny $\varphi_1\colon A \to C$ and a d_2-isogeny $\varphi_2\colon C \to B$, each uniquely defined up to isomorphism, such that $\varphi = \varphi_2 \circ \varphi_1$.*

Proof The proof is given in the full version of the paper [18]. □

2.3 Kani's Lemma and Embedding Isogenies Into Higher Degree

If $(\varphi_{ji})_{ij}$ is a matrix of polarised isogenies $\varphi_{ij}\colon A_i \to B_j$ between PPAVs, then its polarised dual is given by $(\widetilde{\varphi_{ij}})_{ij}$. Whether or not a 2×2 matrix of polarised isogenies is a polarised isogeny is the content of Kani's lemma.

Lemma 2.3 (Kani, [49, Lemma 2.1]). *Let A_1, A_2, B_1, B_2 be PPAVs defined over k and let $\varphi_{ij}\colon A_i \to B_j$ be polarised d_{ij}-isogenies. The map*

$$\Phi = \begin{pmatrix} \varphi_{11} & \varphi_{21} \\ \varphi_{12} & \varphi_{22} \end{pmatrix} : A_1 \times A_2 \to B_1 \times B_2$$

is a polarised isogeny if and only if $d_{11} = d_{22}, d_{12} = d_{21}$ and the corresponding square S_Φ

$$\begin{array}{ccc} A_1 & \xrightarrow{\varphi_{11}} & B_1 \\ {\scriptstyle -\varphi_{12}}\downarrow & & \downarrow{\scriptstyle \widetilde{\varphi_{21}}} \\ B_2 & \xrightarrow[\widetilde{\varphi_{22}}]{} & A_2 \end{array}$$

commutes. In such cases, $\deg_p(\Phi) = d_{11} + d_{12}$ *and we call Φ the* Kani-map *corresponding to the* Kani-square *S_Φ. Further, if $d_{11} = d_{22}$ is coprime to $d_{12} = d_{21}$, we call the Kani-square a* (d_{11}, d_{12})-isogeny diamond. *Finally, when d_{11} is coprime to d_{12} and $d = d_{11} + d_{12}$ is coprime to the characteristic of k, then*

$$\ker(\Phi) = \{([d_{11}]x, \widetilde{\varphi_{21}}\varphi_{11}(x)) \mid x \in A_1[d]\}\,.$$

If Φ can be efficiently evaluated, then so can φ_{ij}, and Φ is said to *embed* φ_{ij}. In other words, Φ is an *efficient representation* of the φ_{ij} in the sense of the following definition.

Definition 2.2. *Let $\mathscr{A}$ be an algorithm. Given a d-isogeny $\varphi : A \to B$ between PPAVs of dimension g defined over $\mathbb{F}_q$, an* efficient representation *of φ with respect to $\mathscr{A}$ is some data $D_\varphi \in \{0,1\}^*$ of polynomial size in g, $\log(d)$ and $\log(q)$ such that for all $P \in A(\mathbb{F}_{q^k})$, $\mathscr{A}$ with input (D_φ, P) returns $\varphi(P)$ in polynomial time in g, $\log(d)$, k and $\log(q)$.*

3 Our Algorithmic Approach

We now describe our algorithm for computing the class group action by a general orientation.

3.1 Applying Kani's Lemma in Dimension 4

The factorisation lemma (Lemma 2.2) gives us a strategy to embed arbitrary isogenies into higher dimension [46, Remark 2.10]. If $\varphi\colon A \to B$, $\psi\colon A \to C$ are isogenies between PPAVs with coprime polarised degrees, we can form a Kani-diamond by factoring the composition $\psi\widetilde{\varphi}$ as $\widetilde{\varphi'}\psi'$ whereby $\psi'\colon B \to D$ and $\varphi'\colon C \to D$ have polarised degrees $\deg_{\mathrm{p}}(\varphi') = \deg_{\mathrm{p}}(\varphi), \deg_{\mathrm{p}}(\psi) = \deg_{\mathrm{p}}(\psi')$. The kernel of the resulting Kani-map can be computed so long as $\psi\widetilde{\varphi}$ is known on $B[\deg_{\mathrm{p}}(\varphi) + \deg_{\mathrm{p}}(\psi)]$.

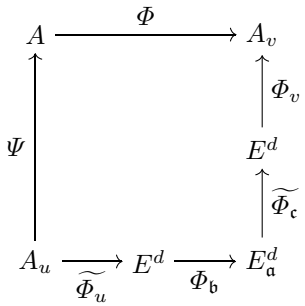
When computing the class-group action of $[\mathfrak{a}]$ on a primitively oriented elliptic curve (E, ι), the isogenies φ, ψ are taken to be $\widehat{\varphi}_{\mathfrak{b}}, \widehat{\varphi}_{\mathfrak{c}}\colon E_{\mathfrak{a}} \to E$ where $\mathfrak{b}, \mathfrak{c}$ are representatives of $[\mathfrak{a}]$. The composition $\widehat{\varphi}_{\mathfrak{c}}\varphi_{\mathfrak{b}} = \varphi_{\overline{\mathfrak{c}}\mathfrak{b}}$ is an endomorphism on E dictated by ι and so the kernel of the resulting 2-dimensional Kani-map can be computed so long as the $\mathrm{N}(\mathfrak{b})+\mathrm{N}(\mathfrak{c})$ torsion is accessible. For practically efficient computation (using [19] for instance), we must find ideals $\mathfrak{b}, \mathfrak{c}$ representing $[\mathfrak{a}]$ such that $\mathrm{N}(\mathfrak{b}) + \mathrm{N}(\mathfrak{c}) = 2^e$, where $2^{e+2} \mid p + 1$ in order to work over $\mathbb{F}_{p^2}$ (the reason we need 2^{e+2} to divide $p + 1$ is due to technicalities when using level-2 theta coordinates to compute higher-dimensional isogenies). However, finding ideals $\mathfrak{b}, \mathfrak{c}$ satisfying this norm equation is not easy in practice and can only be done with standard techniques when p is much bigger than $|\mathrm{disc}(\mathcal{O})|$ (which is not suitable for CSIDH/CSURF, where $|\mathrm{disc}(\mathcal{O})| = p$ or $4p$).

Goal: *solve the norm equation:*

$$u\mathrm{N}(\mathfrak{b}) + v\mathrm{N}(\mathfrak{c}) = 2^e \tag{3}$$

with $2^{e+2} \mid p+1$, $[\mathfrak{a}] = [\mathfrak{b}] = [\mathfrak{c}]$, $u, v > 0$, $\gcd(u\mathrm{N}(\mathfrak{b}), v\mathrm{N}(\mathfrak{c})) = 1$, and such that we can build efficiently computable dimension-d isogenies $\Phi_u\colon E^d \to A_u, \Phi_v\colon E^d \to A_v$ of polarised degrees u, v.

Indeed, the factoring lemma (Lemma 2.2) ensures that we have a $(u\mathrm{N}(\mathfrak{b}), v\mathrm{N}(\mathfrak{c}))$-isogeny diamond



where $\Phi_{\mathfrak{b}} := \mathrm{diag}(\varphi_{\mathfrak{b}}), \Phi_{\mathfrak{c}} := \mathrm{diag}(\varphi_{\mathfrak{c}})\colon E^d \to E^d_{\mathfrak{a}}$, Φ is an $u\mathrm{N}(\mathfrak{b})$-isogeny and Ψ is a $v\mathrm{N}(\mathfrak{c})$-isogeny. Hence, by Kani's Lemma 2.3, we can embed the compositions $\Phi_{\mathfrak{b}} \circ \widetilde{\Phi_u}$ and $\Phi_{\mathfrak{c}} \circ \widetilde{\Phi_v}$ into a $2d$-dimensional 2^e-isogeny $F : A_u \times A_v \to E^d_{\mathfrak{a}} \times A$.

Except when u and v are perfect squares as in [47] (which does not make the norm equation much easier to solve), we cannot expect Φ_u and Φ_v to be one-dimensional. Using higher dimensions $d \geq 2$ appears necessary because it is difficult to construct (efficient representations of) outgoing 1-dimensional isogenies Φ_u, Φ_v of large given degree u, v on E without knowing the endomorphism ring $\mathrm{End}(E)$: the only tools we have at our disposal to produce arbitrary-degree isogenies all require higher dimensions. In fact, a variant of this problem (construct a large *prime* degree isogeny on E) is conjectured to be cryptographically hard [4, Problem 2].

In any case, *Zarhin's trick* [27, Th. 11.29, Eq. (4)] would allow us to efficiently construct endomorphisms Φ_u, Φ_v of E^4 with the prescribed polarised degrees u, v. However, we consider computing the corresponding 8-dimensional isogeny F to be too expensive. Instead, we will see in Sect. 4 how to find solutions when $d = 2$. Consequently, we have to compute a 4-dimensional isogeny F, which can be done efficiently with level-2 theta coordinates [17]. As mentioned earlier, the algorithms to compute the 2^e-isogeny F require 2^{e+2}-torsion points, so we need $2^{e+2} | p + 1$ instead of $2^e | p + 1$ to be able to work over $\mathbb{F}_{p^2}$.

Unfortunately, there do not always exist positive integer solutions u, v to $u\mathrm{N}(\mathfrak{b}) + v\mathrm{N}(\mathfrak{c}) = 2^e$. Indeed, the *Frobenius coin problem* tells us that we are only guaranteed solutions when $2^e \geq \mathrm{N}(\mathfrak{b})\mathrm{N}(\mathfrak{c}) - \mathrm{N}(\mathfrak{b}) - \mathrm{N}(\mathfrak{c})$; on the other hand, when $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$ (as in CSIDH), the smallest representative of any class $[\mathfrak{a}]$ in $\mathrm{Cl}(\mathcal{O})$ is expected to be of size roughly $\sqrt{p}$. So, recalling that $2^{e+2} \mid p + 1$, we see that we rarely have guaranteed solutions u, v. The solution we propose to overcome this difficulty is to give some additional freedom on the choice of ideals $\mathfrak{b}, \mathfrak{c}$ by factoring out their smooth part.

3.2 Factoring Out Easy Steps First

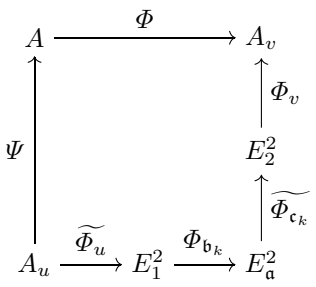
We will now describe how to compute the action of $[\mathfrak{a}]$ on (E, ι) in four dimensions. Throughout this section, $\mathfrak{b}, \mathfrak{c}$ are primitive[2] representatives of $[\mathfrak{a}]$.

[2] i.e. neither $\mathfrak{b}$ nor $\mathfrak{c}$ is contained in $N\mathcal{O}$ for any $N > 1$.

The Idea. Suppose we can factor $\mathfrak{b} = \mathfrak{b}_e\mathfrak{b}_k, \mathfrak{c} = \mathfrak{c}_e\mathfrak{c}_k$, so that the isogenies $\varphi_{\mathfrak{b}_e}\colon E \to E_1, \varphi_{\mathfrak{c}_e}\colon E \to E_2$ are efficiently computable (*e.g.* smooth degree) and so that we can find efficiently computable isogenies $\Phi_u\colon E_1^2 \to A_u, \Phi_v\colon E_2^2 \to A_v$ (*e.g.* integer-matrix endomorphisms) with polarised degrees u, v satisfying

$$u\mathrm{N}(\mathfrak{b}_k) + v\mathrm{N}(\mathfrak{c}_k) = 2^e. \tag{4}$$

Then we can consider the following $(u\mathrm{N}(\mathfrak{b}), v\mathrm{N}(\mathfrak{c}))$-isogeny diamond



where Φ_I is the 2-dimensional isogeny $\mathrm{diag}(\varphi_I, \varphi_I)$ given by the action of $I \subseteq \mathcal{O}$. Then, Kani's lemma yields a 4-dimensional isogeny

$$F := \begin{pmatrix} \Phi_{\mathfrak{b}_k} \circ \widetilde{\Phi_u} & \Phi_{\mathfrak{c}_k} \circ \widetilde{\Phi_v} \\ -\Psi & \widetilde{\Phi} \end{pmatrix} : A_u \times A_v \to E_{\mathfrak{a}}^2 \times A, \tag{5}$$

with kernel

$$\begin{aligned} \ker(F) &= \left\{ ([u\mathrm{N}(\mathfrak{b}_k)]x, \Phi_v\widetilde{\Phi_{\mathfrak{c}_k}}\Phi_{\mathfrak{b}_k}\widetilde{\Phi_u}(x)) \mid x \in A_u[2^e] \right\} \\ &= \left\{ ([u\mathrm{N}(\mathfrak{b}_k)]x, \Phi_v\Phi_{\overline{\mathfrak{c}_k}\mathfrak{b}_k}\widetilde{\Phi_u}(x)) \mid x \in A_u[2^e] \right\} \\ &= \left\{ ([\mathrm{N}(\mathfrak{b}_k)]\Phi_u(y), \Phi_v\Phi_{\overline{\mathfrak{c}_k}\mathfrak{b}_k}(y)) \mid y \in E_1^2[2^e] \right\}. \end{aligned} \tag{6}$$

Since $\mathfrak{b}, \mathfrak{c}$ both represent $[\mathfrak{a}]$, $\overline{\mathfrak{c}}\mathfrak{b} = \sigma\mathcal{O}$ is principal and we can evaluate $\varphi_{\overline{\mathfrak{c}}\mathfrak{b}} = \iota(\sigma)$. Having computed $\varphi_{\mathfrak{c}_e}$ and $\varphi_{\mathfrak{b}_e}$, we can evaluate

$$\mathrm{N}(\mathfrak{b}_e)\mathrm{N}(\mathfrak{c}_e)\varphi_{\overline{\mathfrak{c}_k}\mathfrak{b}_k} = \varphi_{\mathfrak{c}_e}\iota(\sigma)\widehat{\varphi_{\mathfrak{b}_e}}. \tag{7}$$

and compute $\ker(F)$ (more exactly points of 2^{e+2}-torsion generating $\ker(F)$ as required in [17]). We refer to Sect. 5.2 for more details on the kernel computation in the case of CSIDH.

The Algorithm. By construction, $u\mathrm{N}(\mathfrak{b}_k) + v\mathrm{N}(\mathfrak{c}_k) = 2^e$ has at least as many positive integer solutions u, v as $u\mathrm{N}(\mathfrak{b}) + v\mathrm{N}(\mathfrak{c}) = 2^e$. Indeed, every ideal appearing as $\mathfrak{b}_k$ can also be chosen as $\mathfrak{b}$ (with $\mathfrak{b}_e = 1$). More relevant, though, is that the norms of the ideals $\mathfrak{b}_k$ are smaller, and so we hope for more u, v solutions.

The ideals $\mathfrak{b}_e, \mathfrak{c}_e$ should be chosen so that the corresponding isogenies $\varphi_{\mathfrak{b}_e}, \varphi_{\mathfrak{c}_e}$ are easy to compute, *e.g.* of smooth degrees. More precisely, letting $\mathfrak{B}$ be a set

of small primes, we permit $\mathfrak{b}_e, \mathfrak{c}_e$ to be a product of ideals lying above the primes in $\mathfrak{B}$. Note that since $\mathfrak{b}_e, \mathfrak{c}_e$ are primitive, these primes will all be split in $\mathcal{O}$.

With this requirement we now proceed as follows (we refer to Sect. 5.1 for details).

1. Obtain a set S of short representatives of $[\mathfrak{a}]$ (*e.g.* by Lagrange-reduction).
2. Factor each $\mathfrak{b}$ in S as $\mathfrak{b} = \mathfrak{b}_e\mathfrak{b}_k$ maximising the norm of $\mathfrak{b}_e$.
3. Select a pair of ideals $\mathfrak{b} = \mathfrak{b}_e\mathfrak{b}_k$ and $\mathfrak{c} = \mathfrak{c}_e\mathfrak{c}_k$ from S with $\mathrm{N}(\mathfrak{b}_k), \mathrm{N}(\mathfrak{c}_k)$ coprime, prioritizing those with the smallest $\mathrm{N}(\mathfrak{b}_k), \mathrm{N}(\mathfrak{c}_k)$.
4. Enumerate the positive integer solutions u, v of $u\mathrm{N}(\mathfrak{b}_k) + v\mathrm{N}(\mathfrak{c}_k) = 2^e$.
5. Return a solution u, v if it allows for the construction of Φ_u, Φ_v; else choose a different $\mathfrak{b}, \mathfrak{c}$ pair.

We briefly remark that this technique of splitting the isogeny computation into an easy 1-dimensional, followed by potentially expensive ($d \geq 2$)-dimensional computation also allows us to compute more isogenies Φ_u of prescribed polarised degree u. Indeed, the naïve approach of constructing Φ_u as a 2×2 integer-matrix

$$\begin{pmatrix} x_u & -y_u \\ y_u & x_u \end{pmatrix} \tag{8}$$

only permits $u = x_u^2 + y_u^2$ to be a sum of two squares. Composing Φ_u as the diagonal of some one-dimensional isogeny of sufficiently smooth degree g_u (*e.g.* $\mathfrak{B}$-smooth), and then a matrix as above allows for polarised degrees $u = g_u(x_u^2 + y_u^2)$ (see Sect. 4.1).

3.3 Finding the Codomain Orientation

Once we have computed the 4-dimensional isogeny $F : A_u \times A_v \to E_\mathfrak{a}^2 \times A$ and extracted $E_\mathfrak{a}$ from its codomain (following the approach presented in the full version of the paper [18, Appendix B] in the case of CSIDH), we still have to compute the orientation $\iota_\mathfrak{a} := (\varphi_\mathfrak{a})_*(\iota)$ on $E_\mathfrak{a}$. In the case of CSIDH, this orientation is naturally given by the Frobenius endomorphism and does not need to be computed. However, this orientation is not that easy to determine in general.

Given a generator $\alpha \in \mathcal{O}$ of $\mathcal{O}$ (i.e. $\mathcal{O} = \mathbb{Z}[\alpha]$), we only have to compute an efficient representation of $\iota_\mathfrak{a}(\alpha)$, as defined in Definition 2.2. Assuming $|\mathrm{disc}(\mathcal{O})| \leq 2^{2f}$ with $2^f \mid p+1$, we can always find α of norm bounded by 2^{2f} and it suffices to evaluate $\iota_\mathfrak{a}(\alpha)$ on a basis of $E_\mathfrak{a}[2^f]$ to efficiently represent this endomorphism, *e.g.* using the higher dimensional interpolation algorithms from [50, Theorem 3.7 & Sect. 6.4]. Furthermore, if $e + 2 \leq f$, then knowing $\iota(\alpha)$ on the 2^f-torsion is sufficient to compute the 2^{e+2}-torsion lying above $\ker(F)$ given by Eq. (6) so to compute the action of any ideal on E. The same holds for $E_\mathfrak{a}$: the knowledge of $\iota_\mathfrak{a}(\alpha)$ on $E_\mathfrak{a}[2^f]$ is sufficient to propagate the ideal class group action further.

Now we explain how we compute $\iota_{\mathfrak{a}}(\alpha)$ on a basis of $E_{\mathfrak{a}}[2^f]$. Using the fact that $\overline{\mathfrak{b}}\mathfrak{b} = \sigma\mathcal{O}$, we obtain that

$$\iota_{\mathfrak{a}}(\alpha) = (\varphi_{\mathfrak{a}})_*(\iota) = \frac{1}{\mathrm{N}(\mathfrak{a})}\varphi_{\mathfrak{a}} \circ \iota(\alpha) \circ \widehat{\varphi}_{\mathfrak{a}} = \frac{1}{\mathrm{N}(\mathfrak{b})}\varphi_{\mathfrak{b}} \circ \iota(\alpha) \circ \widehat{\varphi}_{\mathfrak{b}}.$$

Let (P_1, Q_1) and $(P_{\mathfrak{a}}, Q_{\mathfrak{a}})$ be two bases of $E_1[2^f]$ and $E_{\mathfrak{a}}[2^f]$ respectively. Then by Eq. (5), we have $F(\Phi_u(P_1, 0), 0) = ([u]\varphi_{\mathfrak{b}_k}(P_1), 0, *)$ so we can evaluate $[u]\varphi_{\mathfrak{b}_k}(P_1)$, hence $\varphi_{\mathfrak{b}_k}(P_1)$ by multiplying by an inverse of $u \mod 2^f$. We compute $\varphi_{\mathfrak{b}_k}(Q_1)$ similarly. Now, with (easy) discrete logarithm computations in $E_{\mathfrak{a}}[2^f]$, we may find $a, b, c, d \in \mathbb{Z}$ such that $\varphi_{\mathfrak{b}_k}(P_1) = [a]P_{\mathfrak{a}} + [b]Q_{\mathfrak{a}}$ and $\varphi_{\mathfrak{b}_k}(Q_1) = [c]P_{\mathfrak{a}} + [d]Q_{\mathfrak{a}}$. Then, a simple matrix inversion yields

$$\widehat{\varphi}_{\mathfrak{b}_k}(P_{\mathfrak{a}}) = [\delta\mathrm{N}(\mathfrak{b}_k)]([d]P_1 - [b]Q_1) \text{ and } \widehat{\varphi}_{\mathfrak{b}_k}(Q_{\mathfrak{a}}) = [\delta\mathrm{N}(\mathfrak{b}_k)](-[c]P_1 + [a]Q_1),$$

with δ an inverse of $ad - bc \mod 2^f$ (which does exist because $\mathrm{N}(\mathfrak{b}_k) = \deg(\varphi_{\mathfrak{b}_k})$ is odd, so $\varphi_{\mathfrak{b}_k}$ maps a 2^f-torsion basis to a 2^f-torsion basis). So we have computed $(\widehat{\varphi}_{\mathfrak{b}_k}(P_{\mathfrak{a}}), \widehat{\varphi}_{\mathfrak{b}_k}(Q_{\mathfrak{a}}))$. Similarly, knowing $\varphi_{\mathfrak{b}_e}$, we can compute the action of its dual $\widehat{\varphi}_{\mathfrak{b}_e}$ on the 2^f-torsion provided that $\mathrm{N}(\mathfrak{b}_e)$ is odd. We can impose this condition by removing 2 from the set of allowed primes $\mathfrak{B}$. Hence, we can evaluate $\widehat{\varphi}_{\mathfrak{b}_e} \circ \widehat{\varphi}_{\mathfrak{b}_k} = \widehat{\varphi}_{\mathfrak{b}}$ on the basis $(P_{\mathfrak{a}}, Q_{\mathfrak{a}})$. We can then evaluate $[\mathrm{N}(\mathfrak{b})]\iota_{\mathfrak{a}}(\alpha) = \varphi_{\mathfrak{b}} \circ \iota(\alpha) \circ \widehat{\varphi}_{\mathfrak{b}} = \varphi_{\mathfrak{b}_k} \circ \varphi_{\mathfrak{b}_e} \circ \iota(\alpha) \circ \widehat{\varphi}_{\mathfrak{b}_e} \circ \widehat{\varphi}_{\mathfrak{b}_k}$ on $(P_{\mathfrak{a}}, Q_{\mathfrak{a}})$, using our knowledge of the action of $\iota(\alpha)$ on the 2^f-torsion, of $\varphi_{\mathfrak{b}_e}$ and of $\varphi_{\mathfrak{b}_k}$ via F. Since $\mathrm{N}(\mathfrak{b}) = \mathrm{N}(\mathfrak{b}_e)\mathrm{N}(\mathfrak{b}_k)$ is odd, we can invert it modulo 2^f and finally obtain $(\iota_{\mathfrak{a}}(\alpha)(P_{\mathfrak{a}}), \iota_{\mathfrak{a}}(\alpha)(Q_{\mathfrak{a}}))$, as desired.

4 Computing 2-Dimensional Isogenies of Prescribed Polarised Degree

In this section, we explain how to construct dimension 2 isogenies Φ_u, Φ_v of prescribed polarised degree u, v, in order to solve Eq. (3) (or more precisely the version in Eq. (4)).

We first treat an easy special case in Sect. 4.1. Then we explain in Sect. 4.2 how to treat the case of u (or v) an arbitrary positive integer. The drawback of the latter approach is that it involves computing a 4-dimensional 2^e-isogeny to build Φ_u in dimension 2.

4.1 The Simplest Case: When the Polarised Degree Is $\mathfrak{B}$-Good

Let us first assume that u and v are a sum of two squares. Then, given a curve E and an integer $u = x_u^2 + y_u^2$, the endomorphism

$$M_u = \begin{pmatrix} x_u & -y_u \\ y_u & x_u \end{pmatrix} \tag{9}$$

on E^2 is such that $M_u\widetilde{M_u} = [u]$ is the multiplication by u on each component. Hence M_u induces an isogeny Φ_u of polarised degree u as desired.

The probability that a random number u can be written as sum of two squares is roughly $1/\sqrt{\log(u)} \approx 1/\sqrt{\log(p)/2}$. Unfortunately, writing u as above requires the factorisation of u. If we instead restrict u to be a prime, the probability goes down to the order of $1/\log(u)$. Since we need both u and v to be of this form, we expect to find a valid pair after $\approx \log(p)^2$ attempts. Looking at the size of the primes we are working with, this approach becomes soon too expensive.

Instead, we can adopt the same strategy as in Sect. 3.2 to relax our condition and aim for $\mathfrak{B}$-good values u, v, for a given set of small primes.

Definition 4.1. *Let $\mathfrak{B}$ be a set of small primes. We say that $u \in \mathbb{N}^*$ is $\mathfrak{B}$-good when u is of the form $u = g_u(x_u^2 + y_u^2)$ where $x_u, y_u \in \mathbb{N}$ and $g_u \in \mathbb{N}^*$ is a product of primes in $\mathfrak{B}$.*

We may then write $u = g_u(x_u^2 + y_u^2)$ and $v = g_v(x_v^2 + y_v^2)$, as above, where g_u and g_v contain all the prime factors of u, v that are in $\mathfrak{B}$. Recall that an integer can be written as a sum of two squares if and only if, in its prime power decomposition, there is no prime power ℓ^k such that k is odd and $\ell \equiv 3 \mod 4$ [36, Ch. 2.15, Ex. 10]. We can then restrict g_u and g_v to primes that are 3 mod 4. Notice that small prime numbers are more likely to appear in the factorization of a number, and hence to cause failure of the sum of squares condition. Heuristically, by taking out factors $3, 7$ and 11 we already increase the probability of finding a sum of squares by a factor 3. If we choose $\mathcal{O}$ such that these small primes are split in $\mathcal{O}$, we can easily compute isogenies of degrees g_u and g_v via the action of $\mathrm{Cl}(\mathcal{O})$. In particular, in the case of CSIDH, these isogenies are defined over $\mathbb{F}_p$ so they are even more efficient to compute (see Sect. 5.2).

Given a pair u, v with no factors in $\mathfrak{B}$, we are now left with the task of determining if both numbers can be expressed as sums of two squares. As mentioned above, checking this condition requires factoring, which we cannot afford. We instead perform trial division up to a fixed bound (10^4 for the 500 and 1000 parameters, and 10^5 for all the others), and then test what is left for primality. We observe that this procedure reduces the chances of finding a good pair by 2 to 3 orders of magnitude, as expected.

If the primality test succeeds, and all nonsquare factors are 1 mod 4, we can apply Cornacchia's algorithm to obtain x_u and y_u (respectively x_v, y_v). This strategy is further discussed in Sect. 5.2 in the case of CSIDH.

Remark 4.1. It is possible to relax the $\mathfrak{B}$-good condition on (u, v) by allowing u (resp. v) to take the form $g_u(ax_u^2 + by_u^2)$ (resp. $g_v(ax_v^2 + by_v^2)$), where $g_u, g_v, a, b \in \mathfrak{B}$, while ensuring that Kani's lemma remains applicable as in Sect. 3.1. Then, at the cost of computing 4 additional isogenies of degree a and b with Elkies' algorithm, we improve the likelihood that u and v are suitable for computing isogenies of degree u and v in dimension 2. Experiments have shown that allowing a and b to take values in $\{2, 3, 5, 7\}$ increases the probability of success by approximately a factor of 3. Further details on this construction can be found in the full version of the paper [18, Appendix C].

4.2 The General Case

We now treat the case of a general u, v. Although it is easy to construct Φ_u, Φ_v in dimension 4, we saw in Sect. 3.1 that this would involve a dimension 8 isogeny for F. In this section we explain how to construct them in dimension 2, on E^2. For that, we extend the approach of QFESTA [43] from dimension 1 to dimension 2. A drawback is that our approach will be heuristic, although our implementation shows that the heuristic works well in practice.

Let (E, ι) be an $\mathcal{O}$-oriented supersingular elliptic curve. Let $\Delta = \mathrm{disc}(\mathcal{O})$ be the discriminant of $\mathcal{O}$. First, we show that we can heuristically build (efficient) endomorphisms on E^2 of polarised degree N, as long as $N \gg |\Delta|$. By construction, we can evaluate any endomorphism $\gamma \in \mathcal{O}$, which gives us a 1-dimensional isogeny $\iota(\gamma)$ of degree $N(\gamma)$. Since $\sqrt{\Delta} \in \mathcal{O}$, the quadratic form $\gamma \mapsto N(\gamma)$ restricts on the suborder $\mathbb{Z} + \sqrt{\Delta}\mathbb{Z}$ of $\mathcal{O}$ to the quadratic form $q(x, y) = x^2 + |\Delta| y^2$.

Now let $\gamma_1, \gamma_2 \in \mathcal{O}$ of norms n_1, n_2 respectively, and consider the endomorphism

$$\gamma = \begin{pmatrix} \iota(\gamma_1) & \iota(\overline{\gamma_2}) \\ -\iota(\gamma_2) & \iota(\overline{\gamma_1}) \end{pmatrix} \in \mathrm{End}(E^2).$$

Then, by [50, Lemma 3.2], its polarised dual is

$$\widetilde{\gamma} = \begin{pmatrix} \iota(\overline{\gamma_1}) & -\iota(\overline{\gamma_2}) \\ \iota(\gamma_2) & \iota(\gamma_1) \end{pmatrix},$$

so that $\widetilde{\gamma} \circ \gamma = [n_1 + n_2]$ in $\mathrm{End}(E^2)$ and γ is an $(n_1 + n_2)$-isogeny. In particular, we can build endomorphisms on E^2 of polarised degree N as long as N is represented by the quadratic form $q(x, y, z, t) = (x^2 + y^2) + |\Delta|(z^2 + t^2)$.

Heuristic 1. *Let $N \gg |\Delta|$. Then we can heuristically build an efficient endomorphism $\gamma \in \mathrm{End}(E^2)$ of polarised degree N.*

Justification Since $N \gg |\Delta|$, we can sample many small z, t such that $(z^2 + t^2)|\Delta| < N$. Then if $m = N - |\Delta|(z^2 + t^2)$ is a sum of two squares, we can find γ by the discussion above. This situation can be efficiently detected whenever m is a prime congruent to 1 mod 4, which heuristically happens with probability $\approx 1/\log(N)$. Since $N \gg |\Delta|$, we can try with many different couples (z, t) until one succeeds. □

In practice, we can use a direct adaptation of the algorithm `RepresentInteger` from [22] to find a solution of $(x^2 + y^2) + |\Delta|(z^2 + t^2) = N$.

Now by Lemma 2.2, if γ is an efficiently represented endomorphism of E^2 of polarised degree $N = N_1 N_2$, with N_1, N_2 coprime, then it decomposes uniquely as $\gamma = \mu_2 \circ \gamma_1 = \mu_1 \circ \gamma_2$ where the γ_i, μ_i are isogenies of polarised degree N_i. Furthermore, by Kani's lemma (Lemma 2.3) the 4-dimensional isogeny

$$\Gamma = \begin{pmatrix} \gamma_1 & \widetilde{\mu_2} \\ -\gamma_2 & \widetilde{\mu_1} \end{pmatrix} \tag{10}$$

is of polarised degree $N_1 + N_2$. The kernel of Γ is given by $\mathrm{Ker}\Gamma = \{(N_1 P, \gamma P) \mid P \in E^9[N_1 + N_2]\}$, so we can efficiently evaluate Γ to obtain the 2-dimensional

isogeny γ_1 of polarised degree N_1 as long as the $(N_1 + N_2)$-torsion on E is smooth and accessible (*i.e.* defined over a small extension of $\mathbb{F}_p$). Hence, an efficient representation of γ yields an efficient representation of γ_1 in the sense of Definition 2.2.

For simplicity, and because this matches our implementation choice, we will now restrict to the case where we look for $N_1 + N_2 = 2^e$ a power of two.

Theorem 4.1. *Assume that $E[2^e] \subseteq E(\mathbb{F}_{p^2})$ (with $e \geq 3$), that $2^{3e/2-1} \gg |\Delta|$, and that $u < 2^{e-1}$ is odd. Then, under Heuristic 1, we can efficiently represent an isogeny Φ_u of polarised degree u on E^2.*

Proof. First consider $N = N_1 N_2$ with $N_1 = u, N_2 = 2^e - u$. Since u is assumed to be odd, N_1 is coprime to N_2. If $N \gg |\Delta|$, we can use Heuristic 1 to build an endomorphism in dimension 2 of polarised degree N, which we split via a 4-dimensional isogeny of polarised degree 2^e to get Φ_u.

If this is not the case, then since $2^{3e/2} \gg |\Delta|$ by assumption, this means that u is smaller than 2^f, with $f = \lceil e/2 \rceil$. We use a two step solution like in [26,44]. First we let $u_2 = u(2^f - u)$, then $2^f - 1 \leq u_2 < 2^{2f-2} \leq 2^{e-1}$. Then we let $N = u_2(2^e - u_2)$. By our choice of f, $N \geq 2^{3e/2-1}$, so $N \gg |\Delta|$ and we can use Heuristic 1 to build an endomorphism in dimension 2 of polarised degree N, which we split a first time (via a 4-dimensional isogeny of polarised degree 2^e) to get a 2-dimensional isogeny Φ_{u_2} of polarised degree u_2, which we split again (via a 4-dimensional isogeny of polarised degree 2^f) to get Φ_u. □

Remark 4.2.

- To compute Γ as defined in Eq. (10), we only need the m-torsion to be smooth and accessible on E, where $N_1 + N_2 \mid m^2$, see [51, § 6.3 and Appendix B]. This allows to relax the conditions on Theorem 4.1 to $u < 2^{2e}$ and $2^{3e-1} \gg \Delta$.
- In practice, for our application from Sect. 3, we have $u, v \approx \sqrt{|\Delta|}$. Furthermore, in the case of CSIDH, we have $\Delta = -p$, and we select p such that $p = c2^f - 1$ for a small cofactor c. In particular, we can take $e = f$ in Theorem 4.1, so that $2^e \approx p$. In that case we already have $u(2^e - u) \gg p$, and we do not need the two step solution, nor the more relaxed torsion condition from the item above.

Combining Sect. 3 with Theorem 4.1, the resulting algorithm to compute the action of a quadratic ideal $\mathfrak{a} \subset \mathcal{O}$ on an $\mathcal{O}$-oriented elliptic curve is very similar to the algorithm used in [5] to convert a quaternionic ideal to an isogeny when the full endomorphism ring of E is known. We make up for the lack of known endomorphisms on E to build isogenies of suitable fixed degree by going up to dimension 2 and working on E^2. This doubling of dimension propagates to the rest of the algorithm, and we end up computing Φ_u, Φ_v and $E_{\mathfrak{a}}$ via an isogeny of dimension 4, compared to dimension 2 in [5].

This change of dimension has important consequences for the fine-tuning of the algorithm. Indeed, as we have seen in Sect. 4.1, Φ_u (resp. Φ_v) are much easier

to compute whenever u (resp. v) is a sum of two squares, or at least $\mathfrak{B}$-good in the sense of Definition 4.1. In particular, in that case we do not need to use a 4-dimensional isogeny to compute Φ_u.

A similar situation happened in [5]: if u was a sum of two squares we could compute Φ_u directly without going through a 2-dimensional isogeny. However, experiments showed that spending more time on the norm equation to find u a sum of two squares did not compensate having to do one less 2-dimensional isogeny. The situation is different in our case, because a 4-dimensional 2^e-isogeny is much slower than a 2-dimensional 2^e-isogeny (by a factor roughly $\times 8$). Depending on the size of our parameters, experiments show that the optimal choices depend on the size of Δ.

- If Δ is of moderate size, we look for $\mathfrak{B}$-good u, v, and only compute one 4-dimensional isogeny for $E_{\mathfrak{a}}$.
- If Δ is of large size, we look for only one of u, v to be $\mathfrak{B}$-good, while we let the other one be arbitrary. This time computing $\varphi_{\mathfrak{a}}$ requires two 4-dimensional isogenies. For CSIDH, our experiments in Sect. 6.4 suggest this strategy becomes faster than the previous one for p of 4000 bits. We remark that for very large Δ, we expect that letting u, v be arbitrary and computing three 4-dimensional isogenies would be the optimal choice.

In the case of CSIDH (detailed in Sects. 5 and 6), only the first choice is implemented and the computation stays reasonably efficient even for our biggest parameters (p of 4000 bits) as long as the allowed set of small split primes $\mathfrak{B}$ is well chosen. We refer to Sect. 6.4 for further discussion on the above trade-off.

5 Our Algorithm Specialized to CSIDH

In this section, we apply our algorithm to the CSIDH setting, or more accurately, the CSURF setting [11]; we fix a prime of the form $p = c2^f - 1$ where c is a small odd integer and $f \gg 1$ so that $p \equiv 7 \pmod 8$, and work with curves oriented by $\mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$, where the orientation is given by sending $\sqrt{-p}$ to the Frobenius endomorphism. Not only does using the orientation given by Frobenius make evaluating the orientation very fast, but by using the maximal order instead of the typical CSIDH suborder $\mathbb{Z}[\sqrt{-p}]$, we are able to execute our whole algorithm over $\mathbb{F}_p$, by choosing a particularly suitable basis of $E[2^e]$, and working with x-only arithmetic. This is further detailed in Sect. 5.2 and in the full version of the paper [18, Appendix D]

At a high level, the three-step process is summarized as in Algorithm 1. We now describe the involved steps in more detail.

5.1 Step 1: the Norm Equation

In this section, we explain how to solve the norm equation (4) sketched in Sect. 3.2. This subsection is summarized in Algorithm 2. As in Sect. 4.1, we

Algorithm 1. EvaluateAction$(E, \mathfrak{a})$

Input: E a supersingular elliptic curve over $\mathbb{F}_p$ defined on the surface.
Input: $\mathfrak{a} \subseteq \mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$ an ideal.
Output: $\mathfrak{a} \star E$

1: Compute $\mathfrak{l} \subseteq \mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$ an ideal of small norm. ▷ Precomputation
2: Set $E' := E$ and $\mathfrak{a}' = \mathfrak{a}$.
3: $\texttt{fail}, \mathfrak{b}_e, \mathfrak{b}_k, \mathfrak{c}_e, \mathfrak{c}_k, u, v, e :=$ FindUV$(\mathfrak{a}')$ ▷ Step 1
4: **if** $\texttt{fail}$ **then** ▷ Rerandomize the ideal
5: Compute $\mathfrak{a}' = \mathfrak{a}'\mathfrak{l}$
6: Compute $E' = \bar{\mathfrak{l}} \star E'$ using Elkies algorithm.
7: **run** Line 3 on $E', \mathfrak{a}'$.
8: **end if**
9: $\texttt{KernelData} :=$ ComputeKernelData$(E', \mathfrak{b}_e, \mathfrak{b}_k, \mathfrak{c}_e, \mathfrak{c}_k, u, v, e)$ ▷ Step 2
10: Compute the 4-dimensional isogeny F, defined by $\texttt{KernelData}$. ▷ Step 3
11: Extract $E_{\mathfrak{a}}$ from the codomain of F. ▷ See [18, Appendix B.4].
12: **return** $E_{\mathfrak{a}}$.

fix a set of small Elkies primes $\mathfrak{B}$, *i.e.* of primes that split in $\mathbb{Q}(\sqrt{-p})$. Notice that we can always impose $2 \in \mathfrak{B}$. We begin by creating a list of primitive ideals equivalent to $\mathfrak{a}$. More precisely, we obtain a list of equivalent ideals $\mathfrak{a}^{(i)}$ such that $\mathfrak{a}^{(i)} = \mathfrak{a}_e^{(i)}\mathfrak{a}_k^{(i)}$ and $N(\mathfrak{a}_k^{(i)})$ is a product of primes in $\mathfrak{B}$. We then want to find a pair (i, j) such that for $N_1 = N(\mathfrak{a}_k^{(i)})$ and $N_2 = N(\mathfrak{a}_k^{(j)})$ (note that the smooth part has been removed from the norm), we can find positive integers u, v solving the equation

$$uN_1 + vN_2 = 2^e. \tag{11}$$

Since generically N_1N_2 determines the minimal value for e such that the above equation has a solution, we test pairs in order of increasing N_1N_2.

Given such a pair of norms N_1, N_2, we first check that they are coprime. Indeed, if $\gcd(N_1, N_2)$ is not a power of 2, Eq. (11) has no solutions. Since $2 \in \mathfrak{B}$ and we assume N_1, N_2 has no prime factors in $\mathfrak{B}$, this requirement simply becomes $\gcd(N_1, N_2) = 1$.

If $\gcd(N_1, N_2) = 1$, we can find u_0 and v_0 such that $u_0N_1 + v_0N_2 = 1$ via the extended Euclidean algorithm, and consequently $2^eu_0N_1 + 2^ev_0N_2 = 2^e$. Note that one of u_0 and v_0 will be negative and that we parametrize all possible solutions to the equation by writing

$$(2^eu_0 + kN_2)N_1 + (2^ev_0 - kN_1)N_2 = 2^e,$$

for some $k \in \mathbb{Z}$. This also allows to determine the minimal e for which Eq. (11) has a solution by enumerating the set of k for which both $u = 2^eu_0 + kN_2$ and $v = 2^ev_0 - kN_1$ are positive.

Remark 5.1. The exponent e in Eq. (11) determines the number of four-dimensional 2-isogeny steps to be computed, so taking a lower exponent e will be more efficient.

As remarked before and also observed in [5], solving Eq. (11) directly for $\mathfrak{a}^{(i)}$ and $\mathfrak{a}^{(j)}$, often fails since N_1, N_2 are in $O(\sqrt{p})$ and $2^e < p$, which makes the original Clapoti approach unpractical in our case. Removing factors in $\mathfrak{B}$, makes N_1 and N_2 much smaller and hence greatly increases the probability of finding solutions to Eq. (11): the number of expected solutions grows roughly as the product of all factors removed. This leads to the following tradeoff: the bigger the set of primes $\mathfrak{B}$, the higher the probability of finding a solution, but the more time is spent in the second step computing small degree isogenies. Furthermore, even if the probability of finding a solution for a given choice of $\mathfrak{B}$ is low, and in particular, we fail to find a solution, we can simply rerandomize the starting ideal (by some small ideal) and start over. More details about this approach are given below.

Assuming that we can find enough solution pairs (u, v), for each one we have to determine whether u and v are $\mathfrak{B}$-good, *i.e.* if we can write $u = g_u(x_u^2 + y_u^2)$ and $v = g_v(x_v^2 + y_v^2)$ with the prime factors of g_u and g_v in $\mathfrak{B}$. Recall that a number can be expressed as a sum of two squares if and only if all its prime factors that are 3 mod 4 appear with an even exponent. The obvious choice is then to set g_u (resp. g_v) to be the product of all prime factors that are 3 mod 4 and contained in $\mathfrak{B}$ and dividing u (resp. v). To determine whether what is left contains more prime factors 3 mod 4 we would need to factor u, which is too expensive in general. As such we simply proceed by trial division up to a small bound, hoping that what is left at the end is a prime that is 1 mod 4 (note that we simply reject when it is 3 mod 4). If everything succeeds, we have found the factorization of u, and hence its decomposition as a sum of two squares. The same procedure applies to v. How to choose an optimal set $\mathfrak{B}$ is analyzed in greater detail in Sect. 6.1.

Once we have found a full solution, we proceed to the next step: we will denote the ideals $\mathfrak{a}^{(i)}, \mathfrak{a}^{(j)}$ as $\mathfrak{b}$ and $\mathfrak{c}$, so in particular, $N_1 = N(\mathfrak{b}_k)$ and $N_2 = N(\mathfrak{c}_k)$.

Rerandomization For some ideals, Eq. (11) may not have a solution with u, v of the required shape. As already observed, increasing the bound $\mathfrak{B}$ is always a possible solution, though this quickly becomes expensive.

Another solution to this is to simply rerandomize the starting ideal $\mathfrak{a}$. Namely, if none of the sampled, equivalent ideals gives us a solution for Eq. (11), we can multiply $\mathfrak{a}$ by a non-principal ideal $\mathfrak{l}$ for which the action is easy to evaluate (*e.g.* an ideal above ℓ, where ℓ is the smallest split prime). We obtain a new ideal $\mathfrak{a}' = \mathfrak{a}\mathfrak{l}$ and try to solve Eq. (11) for this ideal instead. If we manage to do so, we can simply run our algorithm on $\mathfrak{a}'$, using this solution, starting from the starting curve $E_{\overline{\mathfrak{l}}} = \overline{\mathfrak{l}} \star E$ instead, which is easy to compute by the choice of $\mathfrak{l}$. Otherwise, we can rerandomize again by computing $\mathfrak{a}'' = \mathfrak{l}\mathfrak{a}'$, and so on. As soon as Eq. (11) has a reasonable chance of being solved, this method takes care of failures quite efficiently, making Step 1 faster at the cost of a few Elkies steps. The average number of rerandomizations for different levels are reported in Table 2.

Algorithm 2. FindUV($\mathfrak{a}$)

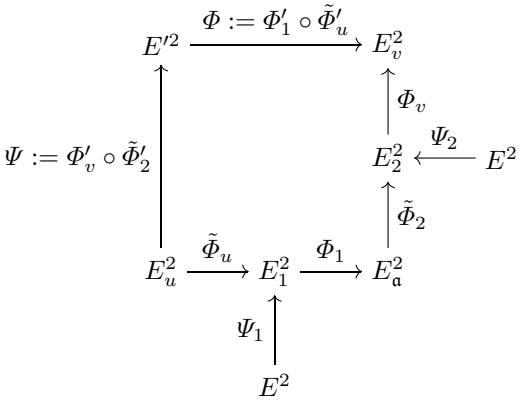
Input: $\mathfrak{a} \subseteq \mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$ an ideal.
Output: `Fail`, flag indicating failure to find solution.
Output: $\mathfrak{b}_e, \mathfrak{b}_k, \mathfrak{c}_e, \mathfrak{c}_k \subseteq \mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$ ideals, such that $\mathfrak{b}_e\mathfrak{b}_k \sim \mathfrak{c}_e\mathfrak{c}_k \sim \mathfrak{a}$.
Output: u, v, e positive integers, such that $u \cdot N(\mathfrak{b}_k) + v \cdot N(\mathfrak{c}_k) = 2^e$, for $e < f - 3$.

Compute a list of short ideals $\mathfrak{a}^{(i)}$ equivalent to $\mathfrak{a}$
Write $\mathfrak{a}^{(i)} = \mathfrak{a}_e^{(i)}\mathfrak{a}_k^{(i)}$, such that $N(\mathfrak{a}_e^{(i)})$ is a product of primes in $\mathfrak{B}$
Sort the list by $N(\mathfrak{a}_k^{(i)})$
for all pairs (i, j) with $1 \leq i < j \leq 10$ **do**
    Set $N_1 = N(\mathfrak{a}_k^{(i)})$, $N_2 = N(\mathfrak{a}_k^{(j)})$
    Find all pairs (u, v) such that $uN_1 + vN_2 = 2^e$, with $e \leq f - 3$
    for each pair (u, v) **do**
        Remove factors of $\mathcal{B}$ from u, v
        Perform trial division on u, v ▷ Bound depends on security level
        if factors 3 mod 4 are found with odd exponent **then**
            Continue to next pair
        end if
        if remaining part of u, v are both prime **then**
            return `False`, $\mathfrak{a}_e^{(i)}, \mathfrak{a}_k^{(i)}, \mathfrak{a}_e^{(j)}, \mathfrak{a}_k^{(j)}, u, v, e$
        end if
    end for
end for
return `True`, _, _, _, _, _, _, _.

5.2 Step 2: Computing Kernel Points

After determining a suitable pair of ideals $\mathfrak{b}, \mathfrak{c}$, our goal is to construct the 4-dimensional 2^e-isogeny described in Sect. 3.2. We now detail how to derive the kernel of this isogeny. This subsection is summarized in Algorithm 3.

Let E denote the starting curve on which we want to act with $\mathfrak{a}$, then we obtain the following diagram derived from the isogeny diamond in Sect. 3.2:



In the diagram above, we have:

- $\Psi_1 = \mathrm{diag}(\varphi_{\mathfrak{b}_e}, \varphi_{\mathfrak{b}_e})$, where $\varphi_{\mathfrak{b}_e}$ is the isogeny corresponding to the action of $\mathfrak{b}_e$; analogously, $\Psi_2 = \mathrm{diag}(\varphi_{\mathfrak{c}_e}, \varphi_{\mathfrak{c}_e})$;
- $\Phi_1 = \mathrm{diag}(\varphi_{\mathfrak{b}_k}, \varphi_{\mathfrak{b}_k})$ and $\Phi_2 = \mathrm{diag}(\varphi_{\mathfrak{c}_k}, \varphi_{\mathfrak{c}_k})$;
- Φ_u and Φ_v are isogenies of polarised degree u and v respectively; as explained in Sect. 4.1, we can write $\Phi_u = M_u \mathrm{diag}(\varphi_u, \varphi_u)$, with $\deg(\varphi_u) = g_u$, $\Phi_v = M_v \mathrm{diag}(\varphi_v, \varphi_v)$ with $\deg(\varphi_v) = g_v$, M_u and M_v being given by Eq. (9).

Further, the isogenies Φ and Ψ are the isogenies of polarised degree uN_1 and vN_2 completing the square, whose existence is guaranteed by Lemma 2.2. In this setting we can also describe them explicitly, as the following lemma shows.

Lemma 5.1. *Φ is of the form $\Phi'_1 \circ \widetilde{\Phi}'_u$, where:*

- *$\Phi'_1 := \mathrm{diag}(\varphi'_{\mathfrak{b}_k}, \varphi'_{\mathfrak{b}_k}) : {E'_1}^2 \to E_v^2$ with $E'_1 := [\overline{\mathfrak{b}}_k] \cdot E_v$ and $\varphi'_{\mathfrak{b}_k}$, the isogeny associated to the action of $\mathfrak{b}_k$ on E'_1;*
- *$\Phi'_u := M_u \mathrm{diag}(\varphi'_u, \varphi'_u) : {E'_1}^2 \to {E'}^2$, M_u being given by Eq. (9) and φ'_u being the isogeny of degree g_u given by the action of the same ideal I_u as φ_u (as a product of Elkies primes).*

Ψ is of the form $\Phi'_v \circ \widetilde{\Phi}'_2$, where:

- *$\Phi'_2 := \mathrm{diag}(\varphi'_{\mathfrak{c}_k}, \varphi'_{\mathfrak{c}_k}) : {E'_2}^2 \to E_u^2$ with $E'_2 := [\overline{\mathfrak{c}}_k] \cdot E_u$ and $\varphi'_{\mathfrak{c}_k}$, the isogeny associated to the action of $\mathfrak{c}_k$ on E'_2;*
- *$\Phi'_v := M_v \mathrm{diag}(\varphi'_v, \varphi'_v) : {E'_2}^2 \to {E'}^2$, M_v being given by Eq. (9) and φ'_v being the isogeny of degree g_v given by the action of the same ideal I_v as φ_v (as a product of Elkies primes).*

In particular, the common codomain of $\widetilde{\Phi}$ and Ψ is a product of elliptic curves ${E'}^2$.

Proof. The proof is given in the full version of the paper [18, Section 5.2]. □

Let (P, Q) be a basis of $E_1[2^{e+2}]$ (recall from Sect. 3.1 that the theta isogeny algorithm requires the 2^{e+2}-torsion to compute a 2^e-isogeny, see [18, Appendix A] for more details). To compute the kernel of the isogeny in dimension 4, we need to compute the points

$$P_u = \varphi_u(P), \quad Q_u = \varphi_u(Q)$$

on E_u and

$$P_v = \varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(P), \quad Q_v = \varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(Q),$$

on E_v.

Isogenies of degree g_u and g_v. The isogenies φ_u and φ_v with $\deg(\varphi_u) = g_u$ and $\deg(\varphi_v) = g_v$ have no specific requirement other than their degree. Although

it is possible to deal with all small primes via Vélu's algorithm [55], it is more efficient to restrict $\mathfrak{B}$ to Elkies primes (*i.e.* splitting primes in $\mathbb{Q}(\sqrt{-p})$) and we can use the correponding algorithm [29] to compute the small degree isogenies.

Note that φ_u of degree g_u and φ_v of degree g_v can be used to construct 2-dimensional isogenies of polarised degree $u = g_u(x_u^2 + y_u^2)$ and $v = g_v(x_v^2 + y_v^2)$ respectively by composition with matrices M_u, M_v described in Sect. 4.1. However, we do not need to perform that step now, as it can be included in the 4-dimensional computation.

Working over $\mathbb{F}_p$. Note that, by definition, the isogenies $\varphi_{\mathfrak{b}}$ and $\varphi_{\mathfrak{c}}$ are $\mathbb{F}_p$-rational, and as noted above, our choice of primes in $\mathfrak{B}$ imply that φ_u and φ_v can also be chosen to be defined over $\mathbb{F}_p$. The main difficulty comes from evaluating these isogenies on a basis of $E[2^{e+2}]$, which is not defined over $\mathbb{F}_p$.

We will show that all operations can in fact be carried out over the Kummer line, where we have a rational basis. Let f the largest integer so that $2^f \mid (p+1)$. By our choice of prime, the two eigenvalues of Frobenius on $E[2^{f-1}]$ are ± 1. Thus, we can pick a basis $\langle P, Q\rangle = E[2^{f-1}]$ corresponding to the two eigenvectors of Frobenius; we let P, Q be points of order 2^{f-1} satisfying

$$\pi(P) = P, \quad \pi(Q) = -Q. \tag{12}$$

Clearly, these points P, Q are $\mathbb{F}_p$-rational on the Kummer line $E/\pm 1$, and together (adjusting by translation by a point of 2-torsion if needed) they form a basis of $E[2^{f-1}]$. This is where we use that we are on the surface. If we were on the floor, picking similar points would not give a basis, since in that case, $[2^{f-2}]P = [2^{f-2}]Q$, the unique rational point of 2-torsion.

In the full version of the paper [18, Appendix D.1] we show how to efficiently generate such a basis, based on a technique which is similar to how one usually computes bases of $E[2^f]$ over $\mathbb{F}_{p^2}$.

Evaluating the Ideals. The rest of this section will be devoted to the computation of P_v, Q_v. In particular, we need to evaluate $\widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}$ on a basis of $E_1[2^{e+2}]$. From Eq. (7) we know that

$$\widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k} = \frac{1}{N(\mathfrak{b}_e)N(\mathfrak{c}_e)} \varphi_{\mathfrak{c}_e} \circ \iota(\sigma) \circ \widehat{\varphi}_{\mathfrak{b}_e} \tag{13}$$

with $\overline{\mathfrak{c}\mathfrak{b}} = \sigma\mathcal{O}$. By construction, $N(\mathfrak{b}_e)$ and $N(\mathfrak{c}_e)$ are both smooth, with all their factors in $\mathfrak{B}$. Let us assume for simplicity that they are also odd; the even case needs a bit more care and is detailed afterwards.

We detail how to evaluate the endomorphism $\iota(\sigma)$. First, assume that σ is of the form $a + b\sqrt{-p}$, with $a, b \in \mathbb{Z}$. Explicitly, this is mapped to the endomorphism $[a] + [b]\pi$, where π denotes the Frobenius endomorphism. Evaluating such an endomorphism on our choice of basis is particularly easy: with the same notation as in the previous section, the evaluation is simply given by

$$\begin{aligned} [a]P + [b]\pi(P) &= [a]P + [b]P = [a+b]P, \\ [a]Q + [b]\pi(Q) &= [a]Q + [b](-Q) = [a-b]Q. \end{aligned}$$

Note in particular that we can evaluate this on the Kummer line. Although σ is an element of the order $\mathbb{Z}[\frac{1+\sqrt{-p}}{2}]$, and hence is more generally of the form $a + b\sqrt{-p}$, where a, b are in $\mathbb{Z}[\frac{1}{2}]$, *i.e.* are rational numbers with denominator at most 2, we show in [18, Appendix D.2] how to evaluate this on our basis (on the Kummer line) without having to use division points, which may not be $\mathbb{F}_p$-rational.

Further, note that since we are after the action on the 2^{e+2}-torsion, we can compute $(N(\mathfrak{b}_e)N(\mathfrak{c}_e))^{-1} \pmod{2^{e+2}}$, to account for the division in Eq. (13). Hence, all that remains to evaluate $\widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}$ is to describe how we efficiently recover the isogenies $\varphi_{\mathfrak{b}_e}$ and $\varphi_{\mathfrak{c}_e}$.

These isogenies are (by definition) the product of prime degree Elkies isogenies. More specifically, say that for a given prime ℓ, we have $v_\ell(N(\mathfrak{b}_e)) = k$, *i.e.* ℓ divides $N(\mathfrak{b}_e)$ exactly k times. Then $\mathfrak{b}_e$ will act with exactly k degree-ℓ Elkies isogenies. We can also assume that all these isogenies will be in the same direction, since otherwise $\mathfrak{b}_e$ and consequently $\mathfrak{b}$ would factor through multiplication by ℓ, and we assumed that $\mathfrak{b}$ was primitive (if it was not, the ideal $\mathfrak{b}/\ell$ would also be equivalent to $\mathfrak{a}$, but with smaller norm).

With Elkies' algorithm, we can return both kernel polynomials of degree ℓ determining the two Elkies isogenies of degree ℓ. To check which polynomial corresponds to the correct isogeny, we check the eigenvalue of Frobenius, as described in the original CSIDH paper [12, Section 3]. In more detail, assume that $(\ell, \sqrt{-p} - \lambda) = \mathfrak{l} \subseteq \mathfrak{b}_e$, *i.e.* we wish to evaluate the action of $\mathfrak{l}$. Since the roots of the correct kernel polynomial $h(x)$ are precisely the x-coordinates of the points of order ℓ, satisfying $\pi(P) = [\lambda]P$, we simply test this equality symbolically: we have the equality

$$(x^p, y^p) \equiv [\lambda](x, y) \pmod{h(x), y^2 - f(x)},$$

where $f(x)$ is the polynomial coming from the curve equation. If these values are not equal, we know that this kernel polynomial corresponds to the other eigenvalue. Note that when we are doing multiple ℓ-isogenies, we need only check the eigenvalue of Frobenius for the first ℓ-isogeny; after that, we can recognise the right kernel polynomial to be the one not corresponding to the previous j-invariant.

Remark 5.2. If $\mathfrak{b}_e$ and $\mathfrak{c}_e$ share some steps, we can perform those steps (corresponding to the action of the ideal $\mathfrak{b}_e + \mathfrak{c}_e$) in advance, and start evaluating the ideal action from the resulting curve. The advantage is twofold: first, we avoid evaluating the same ideal twice. Second, we do not need to evaluate the common part on points. We can instead start computing P_u and P_v directly from the codomain curve.

Even Norm Part We now explain how to deal with even normed ideals. For this, we write the ideal $\mathfrak{b}_e$ as

$$\mathfrak{b}_e = \mathfrak{b}_{e'}\mathfrak{b}_2$$

where $\mathfrak{b}_2$ is an ideal of norm $2^{f_\mathfrak{b}}$, and correspondingly for $\mathfrak{c}_e$. As explained in Remark 5.2, the shared steps of $\mathfrak{b}_e, \mathfrak{c}_e$ can be computed at the beginning, hence we can assume that $\mathfrak{b}_2$ and $\mathfrak{c}_2$ are coprime (*i.e.* the corresponding 2-isogenies go in "opposite" directions). Writing $f_{12} = f_\mathfrak{b} + f_\mathfrak{c}$, we get that

$$\widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k} = \frac{1}{N(\mathfrak{b}_{e'})N(\mathfrak{c}_{e'})2^{f_{12}}} \varphi_{\mathfrak{c}_e} \circ \iota(\sigma) \circ \widehat{\varphi}_{\mathfrak{b}_e}.$$

The only added difficulty that needs to be accounted for is that we cannot find an inverse to $2^{f_{12}}$ modulo 2^{e+2}.

We fix this by taking a basis of a larger torsion group. Using the same technique as for the odd normed case, we can thus evaluate

$$[2^{f_{12}+1}] \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k} = \frac{1}{N(\mathfrak{b}_{e'})N(\mathfrak{c}_{e'})} \varphi_{\mathfrak{c}_e} \circ \iota(\sigma) \circ \widehat{\varphi}_{\mathfrak{b}_e},$$

and hence we can find the evaluation of $\widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}$ on $E_1[2^{e+2}]$ by instead starting with a basis of $E[2^{e+2+f_{12}}]$. (We remark that by construction, we have $e + 2 + f_{12} \leq f$.)

Algorithm 3. ComputeKernelData$(E, \mathfrak{b}_e, \mathfrak{c}_e, \mathfrak{b}_k, \mathfrak{c}_k, u, v, e)$

Input: E a supersingular elliptic curve over $\mathbb{F}_p$ defined on the surface.
Input: $\mathfrak{b}_e, \mathfrak{b}_k, \mathfrak{c}_e, \mathfrak{c}_k \subseteq \mathbb{Z}\left[\frac{1+\sqrt{-p}}{2}\right]$ ideals, such that $\mathfrak{b}_e\mathfrak{b}_k \sim \mathfrak{c}_e\mathfrak{c}_k$.
Input: u, v, e positive integers, such that $u \cdot N(\mathfrak{b}_k) + v \cdot N(\mathfrak{c}_k) = 2^e$, for $e < f - 3$.
Output: `KernelData` necessary for computing the 4-dimensional isogeny.

1: Compute a basis P, Q of $E[2^{f-1}]$. ▷ See [18, Appendix D] to sample special basis.
2: Compute the isogeny $\phi_{\mathfrak{b}_e} : E \to E_1$ using Elkies algorithm.
3: $P_1, Q_1 := \phi_{\mathfrak{b}_e}(P), \phi_{\mathfrak{b}_e}(Q)$.
4: Double P_1, Q_1 until they are of order 2^{e+2}. ▷ Nr. of doublings depends on $\phi_{\mathfrak{b}_e}$.
5: Write $u = g_u(x_u^2 + y_u^2)$ and $v = g_v(x_v^2 + y_v^2)$.
6: Compute a random isogeny $\phi_u : E_1 \to E_u$ of degree g_u using Elkies algorithm.
7: $P_u, Q_u := \phi_u(P_1), \phi_u(Q_1)$
8: Find a generator σ of the principal ideal $\overline{\mathfrak{c}_e\mathfrak{c}_k}\mathfrak{b}_e\mathfrak{b}_k$.
9: Compute the isogeny $\phi_{\mathfrak{c}_e} : E \to E_2$, using Elkies algorithm.
10: Compute $d \equiv N(\mathfrak{c}_{e'})^{-1} \pmod{2^{e+2}}$, where $\mathfrak{c}_{e'}$ is the odd normed part of $\mathfrak{c}_e$.
11: $P_2, Q_2 := [d] \circ \phi_{\mathfrak{c}_e} \circ \sigma(P), [d] \circ \phi_{\mathfrak{c}_e} \circ \sigma(Q)$. ▷ See [18, Algorithm 4] for evaluating σ.
12: Double P_2, Q_2 until they are of order 2^{e+2}. ▷ Nr. of doublings depends on $\phi_{\mathfrak{b}_e}$.
13: Compute a random isogeny $\phi_v : E_2 \to E_v$ of degree g_v using Elkies algorithm.
14: Compute $P_v, Q_v = \phi_v(P_2), \phi_v(Q_2)$.
15: `KernelData` $:= [N(\mathfrak{b}_k), N(\mathfrak{c}_k), g_u, x_u, y_u, g_y, x_v, y_v, P_u, Q_u, P_v, Q_v]$.
16: **return** `KernelData`.

5.3 Step 3: Computing the 4-Dimensional Isogeny

In this step, we compute the 4-dimensional 2^e-isogeny isogeny $F : A_u \times A_v \to E_\mathfrak{a}^2 \times A$ from Sect. 3.2 (defined in Eq. (5)). We can actually be much more explicit

than in Sect. 3.2. From Sect. 5.2, we obtain that $A_u = E_u^2$, $A_v = E_v^2$ and $A = E'^2$. In particular

$$F = \begin{pmatrix} \Phi_1 \circ \widetilde{\Phi}_u & \Phi_2 \circ \widetilde{\Phi}_v \\ -\Phi'_v \circ \tilde{\Phi}'_2 & \Phi'_u \circ \tilde{\Phi}'_1 \end{pmatrix} : E_u^2 \times E_v^2 \to E_{\mathfrak{a}}^2 \times E'^2.$$

Equation (6) also becomes:

$$\begin{aligned} &\ker(F) \\ &= \Big\{ ([uN_1]\varphi_u(P), [uN_1]\varphi_u(Q), \\ &\qquad [g_u]M_v \cdot M_u^T(\varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(P), \varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(Q))) \mid P, Q \in E_u[2^e] \Big\} \\ &= \Big\{ ([N_1]M_u(\varphi_u(P), \varphi_u(Q)), \\ &\qquad M_v(\varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(P), \varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(Q))) \mid P, Q \in E_1[2^e] \Big\}. \end{aligned} \tag{14}$$

To compute F, we proceed with theta coordinates of level 2 and the algorithms from [17] (see also the full version of the paper [18, Appendix B]). These algorithms require a basis of an isotropic subgroup (for the Weil pairing) $K' \subset (E_u^2 \times E_v^2)[2^{e+2}]$ such that $[4]K' = \ker(F)$.

Let (P, Q) be a basis of $E_1[2^{e+2}]$, $\zeta := e_{2^{e+2}}(P, Q)$, η_1, α be inverses of N_1, uN_1 modulo 2^{e+2} respectively, $(P_u, Q_u) := \varphi_u(P, Q)$ and $(P_v, Q_v) := \varphi_v \circ \widehat{\varphi}_{\mathfrak{c}_k} \circ \varphi_{\mathfrak{b}_k}(P, Q)$ the points computed in Step 2. Then, applying [18, Lemma A.1] to F and the ζ-symplectic basis $(x_1, x_2, y_1, y_2) := ((\varphi_{\mathfrak{b}_k}(P), 0), (0, \varphi_{\mathfrak{b}_k}(P)), ([\eta_1]\varphi_{\mathfrak{b}_k}(Q), 0), (0, [\eta_1]\varphi_{\mathfrak{b}_k}(Q)))$, we obtain a basis

$$\begin{aligned} T_1 &:= ([N_1 x_u]P_u, [N_1 y_u]P_u, [x_v]P_v, [y_v]P_v) \\ T_2 &:= ([-N_1 y_u]P_u, [N_1 x_u]P_u, [-y_v]P_v, [x_v]P_v) \\ T_3 &:= ([(1 - 2^e\alpha)x_u]Q_u, [(1 - 2^e\alpha)y_u]Q_u, [\eta_1 x_v]Q_v, [\eta_1 y_v]Q_v) \\ T_4 &:= ([-(1 - 2^e\alpha)y_u]Q_u, [(1 - 2^e\alpha)x_u]Q_u, [-\eta_1 y_v]Q_v, [\eta_1 x_v]Q_v), \end{aligned}$$

of an isotropic subgroup $K' \subset (E_u^2 \times E_v^2)[2^{e+2}]$ such that $[4]K' = \ker(F)$.

To compute F and extract $E_{\mathfrak{a}}$ from its codomain, we take as input:

- The points P_u, Q_u, P_v, Q_v computed in Step 2.
- The integers $N_1, N_2, g_u, x_u, y_u, g_v, x_v, y_v, e$ obtained from a solution of Eq. (11) obtained in Step 1.

This data is not only useful to compute $T_1, \ldots, T_4$ but also gluing and diagonal isogenies located in the beginning of the 2-isogeny chain F that must be handled with care. For this technical reason, one must provide the data above and not $T_1, \cdots, T_4$ or kernel points of F directly. We refer to the full version of the paper [18, Appendix B] for a detailed presentation of the computation of F with level-2 theta coordinates (following the approach of [17]) and to [18, Appendix A] for a cryptographer friendly introduction to higher-dimensional isogeny computations with theta coordinates.

Remark 5.3 (Working over $\mathbb{F}_p$). Using the basis from Eq. (12) for our elliptic curves, we can work with rational theta null points all along our dimension 4 isogeny chain. This is one of the reasons to work on the surface: on the floor, the elliptic curve level 2 theta null points are not rational. Our kernel generators have Frobenius-eigenvalue ± 1, so define $\mathbb{F}_p$-rational points on the Kummer variety and thus their theta-coordinates are $\mathbb{F}_p$-rational (recall that level-2 theta coordinates only give an embedding of the Kummer into projective space, not of the whole variety). This allows us to do the full dimension-4 isogeny computation over $\mathbb{F}_p$, with the exception of the gluing isogenies. Indeed, as currently implemented the gluing formulas mix points with eigenvalue 1 and points with eigenvalue -1, so we need to temporally work over $\mathbb{F}_{p^2}$. We expect that using more symmetric gluing formulas would allow us to stay over $\mathbb{F}_p$ even for gluing, but we leave that for future work.

6 Implementation Results

In this section we describe different implementation choices and their relative efficiency tradeoffs. We give concrete timings, and compare them with the current state of the art of group action primitives. Our implementation is publicly available at https://github.com/pegasis4d.

6.1 Parameter Choices

Since our algorithm relies on higher dimensional representations of isogenies, we start with choosing primes of the form $p = c2^f - 1$, ensuring that the full 2^f-torsion is defined over $\mathbb{F}_{p^2}$. Furthermore, since the probability that Eq. (11) admits a solution, depends heavily on how close 2^f is to p, we restrict to the case where c is small.

Among the possible choices for c we retain those that maximize the number of small primes that split in $\mathcal{O}$, and in particular, primes that are 3 mod 4, since these will greatly increase the probability of finding $\mathfrak{B}$-good pairs (u, v). Note that prime factors of c will automatically be split.

To choose the set $\mathfrak{B}$, we proceed as follows: we only include split primes in $\mathfrak{B}$, which has 3 advantages: first, we can execute the whole algorithm over $\mathbb{F}_p$, instead of over $\mathbb{F}_{p^2}$, resulting in a major speed-up. Second, it allows us to use Elkies' algorithm to compute small degree isogenies without explicitly constructing kernel points over extensions of $\mathbb{F}_p$. Third: such split primes are the only ones that appear as factors of norms of primitive ideals in $\mathcal{O}$.

The number of primes to include in $\mathfrak{B}$ is mostly a tradeoff between steps 1 and 2: including more primes in $\mathfrak{B}$ makes Eq. (11) easier to solve (step 1 becomes faster), but step 2 will have more (and larger) Elkies isogenies to compute. For each parameter set, we started by setting $\mathfrak{B} = \{2, 3\}$ and progressively added primes until computing ℓ-isogenies would become too expensive. We then selected the set with the best overall timing. The final choices are reported in Table 1. A more detailed discussion of the selection process can be found in the full version of the paper [18, Appendix E].

Table 1. Parameter sets used in our implementation for different bitsizes. The prime p is of the form $p = c2^f - 1$ and $\mathfrak{B}$ is the set of small split primes used.

Parameter set	f	c	$\mathfrak{B}$
500	503	33	2, 3, 7, 11, 13
1000	1004	15	2, 3, 5, 7, 11
1500	1551	9	2, 3, 5, 11
2000	2026	51	2, 3, 7, 11, 17
4000	4084	63	2, 3, 7, 11, 17, 19

6.2 Solving the Norm Equation

The first step (Sect. 5.1) begins with creating a reduced basis of the starting ideal $\mathfrak{a}$, followed by a search for small vectors. We are looking for short vectors of (reduced) norm roughly $\sqrt{p}$, but we allow them to be slightly larger at this point (up to $\log(p)\sqrt{p}$). We then attempt to solve Eq. (11) by taking pairs of the shortest vectors. In general, the more pairs, the higher the probability to find a solution. However, the first combinations will be the smallest ones, hence the ones with the highest probability of success. In practice, we observed that taking combinations only involving the 10 shortest vectors, and rerandomizing upon failure, was the most efficient choice for all parameter sets.

Finally, for each pair of (u, v) that we obtain, we have to attempt to write u/g_u and v/g_v as sums of squares. For this, we do trial division on both with all prime factors up to 10^4 for the 500 and 1000 bits parameters, and up to 10^5 for all larger parameters. If we find factors $3 \bmod 4$ (with odd exponent) not included in $\mathfrak{B}$, we immediately discard the pair. Otherwise, we test what is left for primality. If they are prime, we have the factorization of (u, v) and hence their decomposition as sum of two squares. Otherwise, we move on to the next pair.

6.3 Implementation Results

The results of our SageMath 10.5 implementation can be found in Table 2; timings for each step are in seconds, and are obtained by averaging 100 runs on an Intel Core i5-1235U clocked at 4.0 GHz.

We conclude this section with a more detailed comparison with the other available isogeny-based EGAs in the literature. The comparison is summarised in Table 3. The timings for SCALLOP [20] were measured on an Intel i5-6440HQ processor clocked at 3.5 GHz, while the timings for SCALLOP-HD [13] were measured on an Intel Alder Lake CPU core clocked at 2.1 GHz, and the timings for PEARL-SCALLOP [3] were measured on an Intel i5-1038NG7 processor clocked at 2.0 GHz. The timings for the two versions of KLaPoTi [47] were all re-measured on the same hardware setup as the timings presented in Table 2.

PEGASIS is the first instantiation of an EGA at the 2000-bit and 4000-bit security level. In fact, it is even the first time that the full CSIDH group action

Table 2. Timings in seconds for SageMath 10.5 PEGASIS implementation to evaluate one group action at different prime sizes, measured on an Intel Core i5-1235U CPU with maximal clock speed of 4.0 GHz. The last column indicates the number of ideal-class rerandomizations required to find a solution in Step 1. These results are averaged over 100 runs. Step 1 is the time used to solve the norm equation, Step 2 is the time used to derive the kernel of the dimension 4 isogeny, and Step 3 is the time used to compute the dimension 4 isogeny.

Prime size (bits)	Prime	Time (s)				Rerand.
		Step 1	Step 2	Step 3	Total	
508	$3 \cdot 11 \cdot 2^{503} - 1$	0.097	0.48	0.96	1.53	0.17
1008	$3 \cdot 5 \cdot 2^{1004} - 1$	0.21	1.16	2.84	4.21	0.07
1554	$3^2 \cdot 2^{1551} - 1$	1.19	2.85	6.49	10.5	1.53
2031	$3 \cdot 17 \cdot 2^{2026} - 1$	1.68	8.34	11.3	21.3	0.70
4089	$3^2 \cdot 7 \cdot 2^{4084} - 1$	15.6	52.8	53.5	122	0.41

Table 3. Comparison of PEGASIS and other effective group actions in the literature. Timings in seconds of different implementations to evaluate one group action at different (rounded) prime sizes. The SCALLOP variants are starred because they were measured on different hardware setups. The results of both KLaPoTi and PEGASIS are averaged over 100 runs and measured on the same hardware.

Paper	Language	Time (s)				
		Approx. prime size (bits)				
		500	1000	1500	2000	4000
SCALLOP [20]*	C++	35	750			
SCALLOP-HD [13]*	Sage	88	1140			
PEARL-SCALLOP [3]*	C++	30	58	710		
KLaPoTi [47]	Sage	207				
	Rust	1.95				
PEGASIS (This work)	Sage	1.53	4.21	10.5	21.3	121

can be computed at the 1000-bit level. However, as Table 3 shows, PEGASIS also significantly outperforms all earlier works at their security levels. The closest comparison comes with the Rust implementation of KLaPoTi. However, the fact that KLaPoTi was able to achieve a speedup by two orders of magnitude simply by switching from a high-level SageMath implementation to a low-level Rust implementation is also very promising for PEGASIS. Under the assumption that a comparable speedup would be possible for PEGASIS, we see that PEGASIS gives a highly-practical EGA, even at the highest security levels.

One of the main reasons of this efficiency gain is that unlike all the other EGA instantiations, we are able to use the orientation given by Frobenius. This has three key benefits:

– Evaluating the orientation is very efficient,
– We do not need to represent, nor push forward the orientation,
– We can work over $\mathbb{F}_p$.

The price to pay is the need to go up to dimension 4 to compute the action, but as Table 2 shows, it is still reasonably efficient. We stress that our implementation is only a proof of concept, to explore whether level 4000 was feasible in practice. In particular, we are missing many standard implementation tricks; as a simple example, we work with affine coordinates instead of projective coordinates. Despite this, our timings are very encouraging, and we hope a low level optimised implementation could even be made comparable in efficiency to a state-of-the-art implementation of CSIDH ran as a REGA, like in [10].

6.4 Further Improvements

As described in Sect. 4.2, it is possible to force only one among u/g_u and v/g_v to be a sum of squares, and perform the other isogeny in dimension 4. Note that such 4-dimensional isogeny will have only half of the length of the isogeny that we perform at the end, thus being twice as fast (at least, since the isogeny computation is quasi-linear in its length). On the other hand, steps 1 and 2 would both become much faster. This approach hence becomes promising as soon as the 4-dimensional isogeny computation has a cost comparable to step 2, which is the case for the 2000 and 4000 parameter (see Table 2). For such parameters, we computed the first two steps of the algorithm, with $\mathfrak{B} = \{2, 3, 7\}$ in both cases; the results are reported in Table 4. We note that the number of rerandomizations stays comparable whilst the total time for step 1 and step 2 notably decreases.

Table 4. Time taken to compute Step 1 and Step 2 when solving the norm equation with single sum of squares, measured in wall-clock seconds. The last column gives the number of rerandomizations needed. These results are averaged over 100 runs.

Prime (bits)	Prime	Time (s)			Rerand.
		Step 1	Step 2	Total	
2031	$3 \cdot 17 \cdot 2^{2026} - 1$	0.49	3.83	4.32	0.70
4089	$3^2 \cdot 7 \cdot 2^{4084} - 1$	3.25	22.8	26.0	1.25

Estimating the cost of one u-isogeny in dimension 4 as half of the cost of step 3 in Table 2, we expect to reach 21.3 s for the 2000 parameter and 106.2 s for the 4000. This approach is hence comparable to ours in the former case, and even faster in the latter. We leave its implementation as future work. On the other hand, doing both u and v using 4-dimensional isogenies is less efficient at all security levels we considered.

Acknowledgements. This work was supported in part by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (under grant agreement ISOCRYPT - No. 101020788, and grant agreement AGATHA CRYPTY - No. 101116169); by the Research Council KU Leuven grant C14/24/099 and by CyberSecurity Research Flanders with reference number VR20192203; by the Agence Nationale de la Recherche under grant ANR-22-PNCQ-0002 (HQI); by SNSF Consolidator Grant CryptonIs 213766. R. Invernizzi is funded by Research Foundation - Flanders (FWO) under a PhD Fellowship fundamental research (project number 1138925N).

References

1. Abdalla, M., Eisenhofer, T., Kiltz, D., Kunzweiler, S., Riepel, D.: Password-authenticated key exchange from group actions. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II, vol. 13508, LNCS, pp. 699–728. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_24
2. Alamati, N., De Feo, L., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II, vol. 12492, LNCS, pp. 411–439. Springer, Cham, (2020). https://doi.org/10.1007/978-3-030-64834-3_14
3. Allombert, B., et al.: PEARL-SCALLOP: Parameter Extension Applicable in Real Life for SCALLOP (2024)
4. Basso, A., et al.: PRISM: Simple And Compact Identification and Signatures From Large Prime Degree Isogenies. Cryptology ePrint Archive, Report 2025/135 (2025). https://eprint.iacr.org/2025/135
5. Basso, A., et al.: SQIsign2D-West: The Fast, the Small, and the Safer. Cryptology ePrint Archive, Report 2024/760 (2024). https://eprint.iacr.org/2024/760
6. Boyle, E., Kohl, L., Scholl, P.: Homomorphic secret sharing from lattices without FHE. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11477, pp. 3–33. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17656-3_1
7. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11921, pp. 227–247. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34578-5_9
8. X. Bonnetain and A. Schrottenloher. "Quantum Security Analysis of CSIDH". In: *EUROCRYPT 2020, Part II*. Ed. by A. Canteaut and Y. Ishai. Vol. 12106. LNCS. Springer, Cham, May 2020, pp. 493–522. https://doi.org/10.1007/978-3-030-45724-2_17
9. Boyd, C., Davies, G.T., Gjøsteen, K., Jiang, Y.: Fast and secure updatable encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 464–493. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_16
10. Campos, F., Hellenbrand, A., Meyer, M., Reijnders, K.: dCTIDH: Fast & Deterministic CTIDH. Cryptology ePrint Archive, Paper 2025/107 (2025). https://eprint.iacr.org/2025/107
11. Castryck, W., Decru, T.: CSIDH on the surface. In: Ding, J., Tillich, J.-P. (eds.) PQCrypto 2020. LNCS, vol. 12100, pp. 111–129. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-44223-1_7

12. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11274, pp. 395–427. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03332-3_15
13. Chen, M., Leroux, A., Panny, L.: SCALLOP-HD: group action from 2-dimensional isogenies. In: Tang, Q., Teague, V. (eds.) PKC 2024, Part II. LNCS, vol. 14603, pp. 190–216. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-57725-3_7
14. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. Cryptology ePrint Archive, Report 2020/985 (2020). https://eprint.iacr.org/2020/985
15. Couveignes, J.-M.: Hard Homogeneous Spaces. Cryptology ePrint Archive, Report 2006/291 (2006). https://eprint.iacr.org/2006/291
16. Cox, D.A.: Primes of the Form x2+ny2: Fermat, Class Field Theory, and Complex Multiplication, 2nd edn., p. 349. Wiley, Hoboken (2013)
17. Dartois, P.: Fast computation of 2-isogenies in dimension 4 and cryptographic applications. Cryptology ePrint Archive, Paper 2024/1180 (2024). https://eprint.iacr.org/2024/1180
18. Dartois, P., et al.: PEGASIS: Practical Effective Class Group Action using 4-Dimensional Isogenies. Cryptology ePrint Archive, Paper 2025/401 (2025). https://eprint.iacr.org/2025/401
19. Dartois, P., Maino, L., Pope, G., Robert, D.: An Algorithmic Approach to (2, 2)-isogenies in the Theta Model and Applications to Isogeny-based Cryptography. Cryptology ePrint Archive, Paper 2023/1747 (2023). https://eprint.iacr.org/2023/1747
20. De Feo, L., et al.: SCALLOP: scaling the CSI-FiSh. In: Boldyreva, A., Kolesnikov, V., (eds.) PKC 2023, Part I. LNCS, vol. 13940, pp. 345–375. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31368-4_13
21. De Feo, L., Galbraith, S.D.: SeaSign: compact isogeny signatures from class group actions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 759–789. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_26
22. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020. LNCS, vol. 12491, pp. 64–93. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64837-4_3
23. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020. LNCS, vol. 12491, pp. 64–93. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64837-4_3
24. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_28
25. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976). https://doi.org/10.1109/TIT.1976.1055638
26. Duparc, M., Fouotsa, T.B.: SQIPrime: A dimension 2 variant of SQISignHD with non-smooth challenge isogenies. Cryptology ePrint Archive, Report 2024/773 (2024). https://eprint.iacr.org/2024/773
27. Edixhoven, B., van der Geer, G., Moonen, B.: Abelian Varieties (2012). http://van-der-geer.nl/~gerard/AV.pdf
28. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_2

29. Elkies, N.D.: Elliptic and modular curves over finite fields and related computational issues. In: Buell, D.A., Teitelbaum, J.T. (eds.) Computational Perspectives on Number Theory: Proceedings of a Conference in Honor of A. O. L. Atkin, vol. 7, pp. 21–76. AMS/IP Studies in Advanced Mathematics. American Mathematical Society, International Press (1998)
30. Feo, L.D.: Mathematics of Isogeny Based Cryptography (2017). arXiv: 1711.04062 [cs.CR]. https://arxiv.org/abs/1711.04062
31. Galbraith, S., Panny, L., Smith, B., Vercauteren, F.: Quantum Equivalence of the DLP and CDHP for Group Actions. Cryptology ePrint Archive, Report 2018/1199 (2018). https://eprint.iacr.org/2018/1199
32. Galbraith, S.D., Lai, Y.-F., Montgomery, H.: A simpler and more efficient reduction of DLog to CDH for Abelian group actions. In: Tang, Q., Teague, V. (eds.) PKC 2024, Part II. LNCS, vol. 14603, pp. 36–60. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-57725-3_2
33. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold DSS signatures. In: Maurer, U. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 354–371. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68339-9_31
34. Harn, L.: Group-oriented (t, n) threshold digital signature scheme and digital multisignature. IEE Proc. Comput. Digital Techn. **141**(5), 307–313 (1994)
35. Jablon, D.P.: Strong password-only authenticated key exchange. ACM SIGCOMM Comput. Commun. Rev. **26**(5), 5–26 (1996)
36. Jacobson, N.: Basic Algebra I, 2nd edn. Courier Corporation, Chelmsford (1984)
37. Kohel, D., Lauter, K., Petit, C., Tignol, J.-P.: On the quaternionisogeny path problem. LMS J. Comput. Math. **17**(A), 418–432 (2014). https://doi.org/10.1112/S1461157014000151
38. Kuperberg, G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. SIAM J. Comput. **35**(1), 170–188 (2005)
39. Leroux, A., Roméas, M.: Updatable encryption from group actions. In: Saarinen, M.-J., Smith-Tone, D. (eds.) Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II. LNCS, vol. 14772, pp. 20–53. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-62746-0_2
40. Milne, J.S.: Abelian varieties. In: Cornell, G., Silverman, J.H. (eds.) Arithmetic Geometry, pp. 103–150. Springer. New York, NY (1986). https://doi.org/10.1007/978-1-4613-8655-1_5. ISBN: 978-1-4613-8655-1
41. Montgomery, H., Zhandry, M.: Full quantum equivalence of group action DLog and CDH, and more. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 3–32. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_1
42. Moriya, T., Onuki, H., Takagi, T.: SiGamal: a supersingular isogeny-based PKE and its application to a PRF. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020. LNCS, vol. 12492, pp. 551–580. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64834-3_19
43. Nakagawa, K., Onuki, H.: QFESTA: efficient algorithms and parameters for FESTA using quaternion algebras. In: Reyzin, L., Stebila, D. CRYPTO 2024, Part V. LNCS, vol. 14924, pp. 75–106. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68388-6_4
44. Nakagawa, K., Onuki, H.: SQIsign2D-East: A New Signature Scheme Using 2-dimensional Isogenies. Cryptology ePrint Archive, Report 2024/771 (2024). https://eprint.iacr.org/2024/771

45. Onuki, H.: On oriented supersingular elliptic curves. Finite Fields App. 69, 101777 (2021). https://doi.org/10.1016/j.ffa.2020.101777, https://www.sciencedirect.com/science/article/pii/S1071579720301465. ISSN: 1071-5797
46. Page, A., Robert, D.: Introducing Clapoti(s): Evaluating the isogeny class group action in polynomial time. Cryptology ePrint Archive, Report 2023/1766 (2023). https://eprint.iacr.org/2023/1766
47. Panny, L., Petit, C., Stopar, M.: KLaPoTi: An asymptotically efficient isogeny group action from 2-dimensional isogenies. IACR Cryptol. ePrint Arch. p. 1844 (2024). https://eprint.iacr.org/2024/1844
48. Peikert, C.: He gives C-sieves on the CSIDH. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12106, pp. 463–492. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_16
49. Robert, D.: Some applications of higher dimensional isogenies to elliptic curves (overview of results). Cryptology ePrint Archive, Report 2022/1704 (2022). https://eprint.iacr.org/2022/1704
50. Robert, D.: Breaking SIDH in polynomial time. In: Hazay, C., Stam, M. EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 472–503. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_17
51. Robert, D.: On the efficient representation of isogenies (a survey). Cryptology ePrint Archive, Report 2024/1071 (2024). https://eprint.iacr.org/2024/1071
52. Rostovtsev, A., Stolbunov, A.: Public-Key Cryptosystem Based on Isogenies. Cryptology ePrint Archive, Report 2006/145 (2006). https://eprint.iacr.org/2006/145
53. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_22
54. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: 35th FOCS, pp. 124–134. IEEE Computer Society Press, November 1994. https://doi.org/10.1109/SFCS.1994.365700
55. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-rendus de l'Académie des Sciences **273**, 238–241 (1971). https://gallica.bnf.fr
56. Waterhouse, W.C.: Abelian varieties over finite fields. eng. In: Annales scientifiques de l'École Normale Supérieure **2**(4), 521–560 (1969). http://eudml.org/doc/81852