Improved algorithms for finding fixed-degree isogenies between supersingular elliptic curves

Benjamin Benčina[1], Péter Kutas[2,3], Simon-Philipp Merz[4], Christophe Petit[2,5], Miha Stopar[2,6], and Charlotte Weitenkämper[2,3]

[1] Royal Holloway, University of London, UK
[2] University of Birmingham, UK
[3] Eötvös Loránd University, Hungary
[4] ETH Zürich, Switzerland
[5] Université libre de Bruxelles, Belgium
[6] Ethereum Foundation

Abstract. Finding isogenies between supersingular elliptic curves is a natural algorithmic problem which is known to be equivalent to computing the curves' endomorphism rings. When the isogeny is additionally required to have a specific degree d, the problem appears to be somewhat different in nature, yet it is also considered a hard problem in isogeny-based cryptography.
Let E_1, E_2 be supersingular elliptic curves over $\mathbb{F}_{p^2}$. We present improved classical and quantum algorithms that compute an isogeny of degree d between E_1 and E_2 if it exists. Let the sought-after degree be $d = p^{1/2+\epsilon}$ for some $\epsilon > 0$. Our essentially memory-free algorithms have better time complexity than meet-in-the-middle algorithms, which require exponential memory storage, in the range $1/2 \leq \epsilon \leq 3/4$ on a classical computer and quantum improvements in the range $0 < \epsilon < 5/2$.
Our strategy is to compute the endomorphism rings of both curves, compute the reduced norm form associated to $\mathrm{Hom}(E_1, E_2)$ and try to represent the integer d as a solution of this form. We present multiple approaches to solving this problem which combine guessing certain variables exhaustively (or Grover's search in the quantum case) with methods for solving quadratic Diophantine equations such as Cornacchia's algorithm and multivariate variants of Coppersmith's method. For the different approaches we provide implementations and experimental results. Finally, using well-known techniques, a solution to the norm form can be efficiently translated to recover the sought-after isogeny. One significant advantage of our approaches is that they only require very little (classical) memory and are fully parallelizable.

Keywords: Post-quantum cryptography · isogeny computation · cryptanalysis.

1 Introduction

At the core of isogeny-based cryptography is the problem of finding an isogeny between two given elliptic curves, i.e. a group homomorphism which maps dis-

tinguished points of one curve to the other. The *pure isogeny problem* is to find *any* such map between these curves.

However, in many cryptographic schemes additional information is known and the security of the schemes is based on variants of this problem. For instance, one may require a specific solution to the pure isogeny problem, such as an isogeny having a specific degree or a prescribed action on certain points. These are not only additional constraints for finding a solution. In fact, the guaranteed existence of a solution with specific properties supplies additional information about the problem. Thus, it is a priori not clear how the hardness of finding a solution with specific properties compares to the pure isogeny problem.

In perhaps the most famous isogeny-based primitive, the Supersingular Isogeny Diffie–Hellman (SIDH) key exchange [36], the degree of the secret isogenies and certain images under the secret isogenies were known. So-called *torsion point attacks*, first introduced by Petit [49] and later developed further [42, 51], used this additional information to recover the SIDH secrets for modified parameter choices. This raised first suspicions that the additional information could be exploited to weaken SIDH. The recent spectacular attacks on SIDH [11, 45, 53] confirmed these suspicions and furthermore developed an entirely new, powerful toolbox to recover secret isogenies provided one is given some images under the isogeny.

In 2021, Wesolowski proved that the pure isogeny problem between supersingular elliptic curves reduces to the computation of their endomorphism rings [64] which was previously only proved under certain heuristic assumptions [24, 50]. Yet, it is not clear how the hardness of finding an isogeny of a specific degree, i.e. the following problem, compares to the hardness of the pure isogeny problem in general.

Problem 1.1. Given supersingular elliptic curves E_1 and E_2 defined over the field $\mathbb{F}_{p^2}$ with p^2 elements, and given a positive integer d, find an isogeny $E_1 \to E_2$ of degree d if such an isogeny exists.

So far, the only known classical methods to compute solutions to Problem 1.1 are based on exhaustive search, meet-in-the-middle search or more general collision finding algorithms tailored to the concrete amount of memory available [19]. Regarding quantum algorithms, Tani's claw finding algorithm [60] was considered to solve Problem 1.1 with sufficiently smooth d for a while. However, Jaques and Schanck argued that the algorithm's cost of accessing memory renders it more expensive than its classical counterpart [37], and the algorithm was widely dismissed.

Finally, Fouotsa, Kutas, Merz and Ti [27] gave a reduction of Problem 1.1 to the problem of computing the curves' endomorphism rings, if additionally the image of a sufficiently large torsion subgroup is known under the secret isogeny. In an updated version of their article, it is shown that it is also sufficient if the image of a slightly larger torsion subgroup is known only up to scalar [26].

Contributions The strategy behind our new algorithms for solving Problem 1.1 can be roughly broken up into several distinct steps:

- Compute the endomorphism rings of E_1 and E_2.
- Construct a connecting ideal between these two quaternion orders.
- Compute the norm form associated to $\mathrm{Hom}(E_1, E_2)$.
- Represent d via this norm form.
- Compute an ideal equivalent to the connecting ideal of correct norm.
- Convert the ideal back to an isogeny representation (a composition of rational maps if d is smooth, or a more involved representation such as e.g. described in [54]).

We give a more detailed breakdown of the general strategy in Section 4. For many of the subtasks mentioned above, efficient algorithms exist already. Hence the core of our work focuses on the norm form of the connecting ideal and how we can find an element representing the desired isogeny of degree d. Our efforts can be seen as solving a quaternion version of the fixed-degree isogeny problem. We first compute an LLL-reduced basis of $\mathrm{Hom}(E_1, E_2)$ and write the norm form with respect to this basis. The problem can thus be expressed as solving

$$Q(x_1, x_2, x_3, x_4) = d \tag{1}$$

where Q is a quadratic form. Fortunately, using an LLL-reduced basis allows us to give bounds for each coefficient of a solution to the equation.

We provide various approaches for solving this equation which are based on guessing either one or two variables and then solving the remaining Diophantine equation with Cornacchia's algorithm or multivariate versions of Coppersmith's method. This way we obtain improved classical and quantum algorithms for a wide range of degrees.

Outline In Section 2 we present some preliminaries, covering the essential mathematical background on elliptic curves and quaternion algebras, followed by an exposition of several relevant techniques and algorithms for solving multivariate integer equations. We summarize state-of-the-art isogeny and endomorphism ring computations in Section 3.

An overview of our general strategy to find a d-isogeny is given in Section 4. We then focus our remaining sections on the quaternion version of the isogeny computation problem. Section 5 and Section 6 describe our methods for solving the latter using Cornacchia's algorithm and Coppersmith's method, respectively. We include implementation details and experimental results[7]. In Section 7 we present a hybrid approach where one guesses the isogeny partially and then uses our previous results. This allows us to apply our algorithms to a larger range of isogeny degrees d. We summarize our results and provide a thorough comparison with the state of the art in Section 8 before concluding the paper in Section 9. A further application of our results is described in Appendix A: We show how the algorithms developed previously can be used to solve the *order embedding problem* for certain parameters.

[7] Our implementation is available at `https://github.com/isogeny-finding/improved-isogeny-finding`.

2 Preliminaries

We will briefly introduce the necessary mathematical foundations for the algorithms discussed later. This section first covers basic theory of supersingular elliptic curves, their endomorphism rings and the quaternion algebra notions necessary to follow the computations in Sections 5 and 6. For a more detailed background on elliptic curves and isogenies, we refer the reader to [58]. Furthermore, several important algorithms due to Coppersmith and some variants thereof are presented in Section 2.3. These will be used to compute our fixed-degree isogenies later on.

Notation and terminology. Throughout the paper, we will use the following notation. We write $O(\text{poly}(x))$ for quantities asymptotically upper bounded by a polynomial in x. Sometimes, we may want to omit factors polynomial in $\log p$, where p is the characteristic of the finite field we are working with. In these case, we abbreviate $O(x \cdot \text{polylog}\, p)$ by $O^*(x)$. We call an integer *B-smooth*, if it has only prime factors smaller than B. When $B \ll n$, we sometimes say that the integer is "smooth", meaning that its smoothness bound B is in $O(\text{poly}(\log n))$.

2.1 Isogenies

Let E_1, E_2 be elliptic curves defined over a field $\mathbb{F}_q$. A non-constant rational map φ between E_1 and E_2 that is also a group homomorphism is called an *isogeny*. The isogeny φ induces an embedding of the function field $k(E_2)$ in $k(E_1)$ by composition, $\varphi^* : k(E_2) \to k(E_1)$, $f \mapsto f \circ \varphi$. The *degree* of φ, denoted by $\deg \varphi$, is the degree of the extension $k(E_1)/\varphi^*(k(E_2))$.

For every $\varphi : E_1 \to E_2$ of degree d there exists a unique isogeny $\hat{\varphi}$ with the property that $\varphi \circ \hat{\varphi} = [d]$, where $[d]$ denotes scalar multiplication by d (on E_2). This isogeny $\hat{\varphi}$ is called the *dual* of φ and it is also of degree d. Isomorphism classes of elliptic curves are encoded by their *j-invariant*. We denote the set of isogenies from E_1 to E_2 by $\text{Hom}(E_1, E_2)$.

An isogeny from E to itself is called an *endomorphism*. Together with the zero map, endomorphisms of E form a ring under addition and composition denoted by $\text{End}(E)$.

2.2 Quaternion algebras and Deuring's correspondence

Let p be a prime number and let (a, b) be $(-1, -1)$, $(-1, -p)$ or $(-q, -p)$, where $q \equiv 3 \bmod 4$ is a prime that is not a square modulo p, if p is $2, 3 \bmod 4$ or $1 \bmod 4$, respectively. The four-dimensional $\mathbb{Q}$-algebra spanned by $1, i, j, ij$ with multiplication rules $i^2 = a$, $j^2 = b$, and $ij = -ji$ is called the *quaternion algebra ramified at p and ∞*, and denoted $B_{p,\infty}$. In every quaternion algebra there is an involution that sends $\alpha = a_1 + a_2 i + a_3 j + a_4 ij$ to $\overline{\alpha} = a_1 - a_2 i - a_3 j - a_4 ij$. We define the *reduced trace* of a quaternion $\alpha \in B_{p,\infty}$ as $\text{tr}(\alpha) := \alpha + \overline{\alpha}$ and its *reduced norm* as $\text{Norm}(\alpha) := \alpha\overline{\alpha}$. We furthermore define an inner product on the quaternion

algebra as $\langle \alpha, \beta \rangle := \mathrm{tr}(\alpha\bar{\beta})$ for $\alpha, \beta \in B_{p,\infty}$ which induces the canonical norm $||\alpha|| = \sqrt{\langle \alpha, \alpha \rangle}$ of a quaternion α.

Let E be defined over a finite field of characteristic p. Then $\mathrm{End}(E)$ is either an order in an imaginary quadratic field in which case E is called *ordinary*, or a maximal order in the quaternion algebra $B_{p,\infty}$ ramified at p and at infinity in which case E is called *supersingular*. In this paper we are only interested in supersingular elliptic curves.

Deuring [23] showed that there is an equivalence of categories of isogenies between supersingular elliptic curves over $\overline{\mathbb{F}}_p$ and the left ideals of maximal orders of $B_{p,\infty}$, and a bijection between conjugacy classes of supersingular j-invariants and maximal orders (up to equivalence). This bijection is made explicit by sending a supersingular elliptic curve E to its endomorphism ring $\mathrm{End}(E)$. Given a supersingular elliptic curve E_1 over $\mathbb{F}_q$ with endomorphism ring $\mathcal{O}_1 := \mathrm{End}(E_1)$, the pair (E_2, φ), where E_2 is another supersingular elliptic curve over $\mathbb{F}_q$, and $\varphi : E_1 \to E_2$ is an isogeny, is furthermore sent to an integral left $\mathcal{O}_1$-ideal I with right order isomorphic to $\mathcal{O}_2 := \mathrm{End}(E_2)$. We call the ideal $I_\varphi =: I$ a *connecting ideal* of $\mathcal{O}_1$ and $\mathcal{O}_2$, and denote its norm by $\mathrm{Norm}(I) := n_I$. Since every element in I has norm a multiple of n_I, we can normalized by n_I and obtain the *reduced norm*. The set of isogenies from E_1 to E_2 then is a left $\mathcal{O}_1$-module and a right $\mathcal{O}_2$-module. In particular, these isogenies form a $\mathbb{Z}$-lattice of rank 4 [63, Lem. 42.1.11].

2.3 Coppersmith's methods

Inspired by lattice techniques from Håstad [33] and Girault–Toffin–Valleé [30], Coppersmith's methods can find "small" roots of polynomial equations over either $\mathbb{Z}$ or any integer ring $\mathbb{Z}_N$. These algorithms have found many applications in cryptography, e.g. cryptanalysis of RSA with small public exponent when some part of the message is known [16], cryptanalysis of RSA with private exponent smaller than $N^{0.29}$ [8], polynomial-time factorization of $N = p^r q$ for large r [9].

Several variants of Coppersmith's original algorithms for uni- and bivariate polynomials exist [14, 15, 16]. An alternative approach by Howgrave-Graham [34] is often argued to be simpler to analyse [17]. Both approaches can be generalized to handle polynomials with more variables, but the generalization is heuristic only as there is no guarantee that the polynomials found are algebraically independent [5, 17]. Below we focus on three variants by Coron for which an implementation was publicly available[8], and by Bauer–Joux which we implemented ourselves.

Bivariate approach of Coron Coron's algorithm [17] finds small roots of bivariate integer polynomials and follows Howgrave-Graham's approach [34]. The lattice reduction is applied to a full rank lattice that admits a natural triangular basis so that the determinant can be easily computed.

[8] `https://github.com/ubuntor/coppersmith-algorithm`

Given an irreducible polynomial $P(x,y) = \sum_{i,j} p_{i,j} x^i y^j$ with coefficients in $\mathbb{Z}$ and the promise that it has an integer root (x_0, y_0), where $x_0 < X, y_0 < Y$ for some bounds X, Y, the goal is to recover (x_0, y_0).

Let k be a parameter to be fixed later. This parameter will need to be large enough to ensure success of the algorithm. However, a larger k also implies working with a larger lattice, hence a slower attack in practice.

Let $a := P(0,0)$ and $W = \|P(xX, yY)\|_\infty$, where $\|P(x,y)\|_\infty = \max_{i,j}\{|p_{i,j}|\}$. We generate an integer n such that $W \leq n < 2 \cdot W$ and $\gcd(a,n) = 1$, and then define the polynomial $q(x,y) = a^{-1}P(x,y) \pmod n$.

We consider two types of polynomials. For all monomials $x^i y^j$ with $0 \leq i + j \leq k$, we form polynomials of the form $q_{ij} = X^{k-i}Y^{k-j}x^i y^j q$. For the remaining monomials up to degree $\delta + k$, where δ is the total degree of P, we form $q_{ij}(x,y) = nx^i y^j$. Note that all these polynomials have $q_{ij}(x_0, y_0) = 0 \pmod n$.

Let $\mathcal{M}$ be the set of all monomials of the polynomials q_{ij}, and denote by m the number of elements in $\mathcal{M}$. Notice that we have precisely m polynomials q_{ij}. Form a matrix M_1 by labeling each column with a monomial in $\mathcal{M}_1$, and write the coefficients of polynomials q_{ij} in the rows. Denote by L_1 the lattice generated by the rows of M_1. By applying LLL reduction [43] to L_1 and considering the vectors of the LLL-reduced basis $b_1, \ldots, b_m$ of L_1 in order, we retrieve a polynomial h defining the hyperplane of the lattice containing the small solutions of the original polynomial. Hence, h also admits (x_0, y_0) as a root modulo n, but has small coefficients due to LLL-reduction. If the solution (x_0, y_0) is sufficiently small, the polynomial h will be such that $h(x_0, y_0) = 0$ also holds over the integers, and can easily be solved. More precisely, if we define $\|h(x,y)\|^2 := \sum_{i,j} |h_{ij}|^2$ for h_{ij} the coefficient of the monomial $x^i y^j$ in a polynomial h, we have the following result due to Howgrave-Graham [34].

Lemma 2.1. *Let $h(x,y) \in \mathbb{Z}[x,y]$ be a sum of at most ω monomials. Suppose that $h(x_0, y_0) = 0 \pmod n$, where $|x_0| \leq X$, $|y_0| \leq Y$ and $\|h(xX, yY)\| < \frac{n}{\sqrt{\omega}}$, then $h(x_0, y_0) = 0$ holds over the integers.*

If the coefficients of $h(xX, yY)$ are sufficiently small, then $h(x,y)$ cannot be a multiple of $P(x,y)$. The following lemma indicates how small the coefficients need to be.

Lemma 2.2. *[17, Lem. 3] Let $a(x,y)$ and $b(x,y)$ be two non-zero polynomials over $\mathbb{Z}$, separately of maximum degree d in x and y, such that $b(x,y)$ is a multiple of $a(x,y)$ in $\mathbb{Z}[x,y]$. Assume that $a(0,0) \neq 0$ and $b(x,y)$ is divisible by a non-zero integer r such that $gcd(r, a(0,0)) = 1$. Then $b(x,y)$ is divisible by $r \cdot a(x,y)$, and*

$$\|b\| \geq 2^{-(d+1)^2} \cdot |r| \cdot \|a\|_\infty.$$

Using Lemma 2.2, the condition for the algebraic independence of $h(x,y)$ and $P(x,y)$ is

$$\|h(xX, yY)\| < 2^{-\omega} \cdot (XY)^k \cdot W.$$

Since $P(x, y)$ is assumed to be irreducible and $h(x, y)$ is not a multiple of $P(x, y)$, the polynomial $Q(x) = \text{Resultant}_y(h(x, y), P(x, y))$ is non-trivial and $Q(x_0) = 0$. Using any standard root-finding algorithm, x_0 can be recovered, and finally y_0 can be computed by solving $P(x_0, y) = 0$.

The performance of Coron's bivariate algorithm can be summarized in the following two theorems.

Theorem 2.3. *[17, Thm. 4] Let $P(x, y) \in \mathbb{Z}[x, y]$ be an irreducible polynomial, of maximum degree δ in each variable separately. Let X and Y be upper bounds on the desired integer solution (x_0, y_0), and let $W = \max_{i,j}\{|p_{ij}|X^iY^j\}$. If for some $\epsilon > 0$,*

$$XY < W^{\frac{2}{3\delta}-\epsilon}$$

then in time polynomial in $(\log W, 2^\delta)$, one can find all integer pairs (x_0, y_0) pairs such that $P(x_0, y_0) = 0$, $|x_0| \leq X$, and $|y_0| \leq Y$.

Theorem 2.4. *[17, Thm. 5] Under the hypothesis of Theorem 2.3, except that $P(x, y)$ has total degree δ, the bound is*

$$XY < W^{\frac{1}{\delta}-\epsilon}.$$

Multivariate approach of Coron Coron's method as described above can also be extended to handle multivariate polynomial equations [17, Sect. 6], but the extension is heuristic only.

In the three-variable case, polynomials defining the lattice will now be of the forms $X^{k-i}Y^{k-j}Z^{k-l}x^iy^jz^lq$ and $x^iy^jz^tn$ which evaluate at (x_0, y_0, z_0) to 0 over $\mathbb{Z}_n$. Note that given a polynomial $P(x, y)$, a bivariate algorithm only needs to compute one polynomial $h(x, y)$ that is algebraically independent from P to be able to compute (x_0, y_0) such that $P(x_0, y_0) = 0$. On the other hand when given a polynomial $P(x, y, z)$, we require two polynomials $h_1(x, y, z)$ and $h_2(x, y, z)$, where P, h_1, and h_2 are *algebraically independent*. The heuristic nature of the algorithm stems from the difficulty to guarantee algebraic independence (while linear independence when seen as vectors is guaranteed). The method similarly generalizes to more variables.

While Coron's paper does not include a formal claim about the performance of this variant (even up to an algebraic independence assumption), it is similar to the following method which does handle algebraic dependencies.

Bauer–Joux approach In contrast to Coron's algorithm which generalized the simplification found by Howgrave-Graham, the approach by Bauer and Joux [5] extends the original bivariate approach by Coppersmith [14] to three variables. It also uses truncated Gröbner bases to handle so-called *algebraic dependencies*. A similar approach without using Gröbner bases was already proposed by e.g. [38]; the main contribution of [5] is a criterion for guaranteed success. However, it is worth noting that their algorithm often works well heuristically even when the criterion is not met.

While Coron's approach works directly in the lattice generated by polynomials that share a common root (x_0, y_0, z_0) we wish to find, the Bauer–Joux approach aims to find a vector that is orthogonal to a vector s_0 derived from the root which we define later. This yields a polynomial sharing the root (x_0, y_0, z_0) with the initial polynomial.

Again, let $P(x, y, z)$ be a polynomial with integer coefficients and (x_0, y_0, z_0) a small root. Having $P(x, y, z)$ and knowing the bounds $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$, the goal is to recover the root (x_0, y_0, z_0). Let $(\mathcal{S}, \mathcal{M})$ be an admissible pair of sets of monomials for P as in [5], and denote by s and m the number of elements in the sets $\mathcal{S}$ and $\mathcal{M}$, respectively. Normally, we pick a set of monomials $\mathcal{S}$, then multiply them with the monomials of P to obtain the set $\mathcal{M}$.

The algorithm generates the following rational $m \times (m+s)$ matrix $\mathcal{M}_1$. The left $m \times m$ submatrix $D_{\mathcal{M}}$ is a rational diagonal matrix with $X^{-i}Y^{-j}Z^{-k}$ in the row corresponding to the monomial $x^i y^j z^k \in \mathcal{M}$. The columns of the right $m \times s$ submatrix R_1 are the integer coefficients of the polynomials $x^f y^g z^h P$ for $x^f y^g z^h \in \mathcal{S}$, where the coefficient goes into the row belonging to the corresponding monomial [5, Fig. 1].

Denote by L_1 the lattice generated by the rows of $\mathcal{M}_1$. Since $s < m$, there exists a sublattice $L_1' \subset L_1$ such that its vectors have the last s components equal to zero. This can be achieved by noting that R_1 is an integer matrix, so we compute a unimodular transformation U that transforms R_1 into a matrix that has an $s \times s$ identity matrix on the top and zeros everywhere else, then apply U to $D_{\mathcal{M}}$ as well, and take its bottom $(m-s)$ rows as a basis of L_1' (ignoring the zeros in the last s components). Denote $\mathcal{M}_1' = U\mathcal{M}_1$.

Denote by $r_0 = (x_0^i y_0^j z_0^k \mid x^i y^j z^k \in \mathcal{M})$ the solution vector. A short vector in L_1' is then given by $s_0 = r_0\mathcal{M}_1 = \left((\frac{x_0}{X})^i (\frac{y_0}{Y})^j (\frac{z_0}{Z})^k \;\middle|\; x^i y^j z^k \in \mathcal{M} \right) || (0 \ldots 0)$, where $||$ refers to concatenation of vectors. Fix $r := m - s$, compute an LLL-reduced basis $(b_1, \ldots, b_r)$ of L_1', and let $(b_1^\star, \ldots, b_r^\star)$ be its Gram–Schmidt orthogonalization. For $\|s_0\| < \|b_r^\star\|$, we know that $\langle b_r^\star, s_0 \rangle = 0$, i.e. $b_r^\star$ yields a polynomial $P_1'(x, y, z)$ that annihilates $\left(\frac{x_0}{X}, \frac{y_0}{Y}, \frac{z_0}{Z}\right)$. By a change of variables we obtain $P_1(x, y, z) = P_1'\left(\frac{x}{X}, \frac{y}{Y}, \frac{z}{Z}\right)$ which has (x_0, y_0, z_0) as a root. Note that we may obtain other polynomials that annihilate (x_0, y_0, z_0) by considering $b_{r-1}^\star$ and so on, making the next step unnecessary.

The second step is to compute the Gröbner basis $\mathcal{G}$ of the ideal $I = (P, P_1)$, truncated at the maximal degree of the monomials in the set $\mathcal{M}$. We then repeat the previous procedure almost exactly. Denote by t the number of elements in the set $\mathcal{G}$. We construct the rational $m \times (m+t)$ matrix $\mathcal{M}_2$ the same way we constructed $\mathcal{M}_1$ in the previous step, except that we use the polynomials from $\mathcal{G}$ in the columns of the right $m \times t$ matrix, instead of $\{q \cdot P \mid q \in \mathcal{S}\}$. The rest of the procedure is identical, and we obtain P_2 which annihilates (x_0, y_0, z_0). Note that we cannot guarantee that P_2 is algebraically independent from P and P_1, making this algorithm heuristic, although [5] gives a criterion for algebraic independence. The approach is summarized in the following theorem.

Theorem 2.5. *[5, Thm. 1] If $\mathcal{S}$ and $\mathcal{M}$ are admissible sets for P, we can find in polynomial time $P_1(x, y, z)$ which has (x_0, y_0, z_0) as a root over the integers and is algebraically independent from P, provided that*

$$X^{s_x} Y^{s_y} Z^{s_z} < W_1^s 2^{-(6+c)s(d_x^2+d_y^2+d_z^2)}$$

where we assume that $(m-s)^2 \leq cs(d_x^2 + d_y^2 + d_z^2)$ for some constant c. In this formula, W_1 denotes $\|P(xX, yY, zZ)\|_\infty$, and d_x, d_y, d_z denote the maximum degree of P in x, y, z, respectively. By s_x we denote the sum of degrees in x of all the monomials in the set $\mathcal{M} \setminus \mathcal{S}$, i.e. $s_x := \sum_{x^i y^j z^k \in \mathcal{M} \setminus \mathcal{S}} i$, with analogous definitions for s_y and s_z.

3 State of the art on isogeny computation

Naturally, in isogeny-based cryptography there arise the following three number theoretic problems:

- Computing the endomorphism ring of a supersingular elliptic curve.
- Computing any isogeny between two supersingular elliptic curves.
- Computing a degree-d isogeny between two supersingular elliptic curves if it exists.

As shown in [64], the first two problems are equivalent. Furthermore, finding non-scalar endomorphisms was recently proven to be equivalent to the endomorphism ring problem [48]. However, finding a fixed-degree isogeny is only known to be equivalent to endomorphism ring computations if the isogeny degree is smaller than $\sqrt{p}$ where p denotes the characteristic of the field. From a cryptographic standpoint, the degree of the secret isogeny is often revealed (e.g. in SIDH [36] and its variants [3, 20, 46]). The non-obvious result that being able to compute endomorphism rings also breaks these schemes was proven in [29] and [27]. In [2], SIDH-type signatures are proposed whose security relies on the hardness of finding fixed-degree isogenies and it is not known whether this instance can be reduced to endomorphism ring computation. Finally, the difficulty of proving equivalence between finding any isogeny and an isogeny of a fixed degree impacts the performance of SQISign [21]. This is because not being able to compute an isogeny of optimal length between two curves of known endomorphism ring slows down the protocol significantly.

In this section, we briefly survey the current state of the art for finding isogenies between two supersingular elliptic curves and closely related algorithms. First, we discuss the problem of computing endomorphism rings of supersingular elliptic curves. Then we review algorithms that recover *any* isogeny between two given supersingular curves, before we discuss algorithms that recover an isogeny of a given degree under the premise that such an isogeny exists.

Computing endomorphism rings of supersingular elliptic curves The problem of computing endomorphism rings of elliptic curves was first studied by

Kohel in his thesis [41]. The endomorphism ring computation problem underlies most isogeny-based protocols in the literature today, including SQISign [21] and CSIDH [12].

For supersingular elliptic curves over $\overline{\mathbb{F}}_p$ this is considered to be a hard problem. Kohel gave an algorithm with $O^*(p)$ time and memory costs which was later improved to $O^*(p^{1/2})$ by Galbraith. The algorithm given in [25] runs in $O\left((\log p)^2 p^{1/2}\right)$ with low memory requirements.

Computing an isogeny of arbitrary degree For any prime p, the full supersingular isogeny graph with its roughly $p/12$ isomorphism classes of supersingular elliptic curves over $\overline{\mathbb{F}}_p$ is connected. Thus, one could use a simple collision search to find a path between two given elliptic curves in $O^*(p^{1/2})$ time and memory.

Delfs and Galbraith showed how to find isogenies in the same time but requiring significantly less memory [22]. Their algorithm splits the isogeny computation into two parts. First, a random walk from both given curves is computed until a connection to the subgraph of supersingular elliptic curves defined over the base field $\mathbb{F}_p$ is found. There are roughly $\sqrt{p}$ subfield curves in the full isogeny graph and therefore this step requires $O^*(p^{1/2})$ bit operations. In the second step, one searches a subfield isogeny connecting both curves defined over $\mathbb{F}_p$. Using a meet-in-the-middle strategy, the isogeny can be recovered in $O^*(p^{1/4})$, or alternatively using a different collision finding algorithm requiring less memory. The concrete complexity of the Delfs–Galbraith algorithm was analyzed and further improved in [18]. However, the improvements did not change the asymptotic complexity of $O^*(p^{1/2})$.

Assuming GRH, the problem of finding an isogeny between two supersingular curves is polynomial-time and memory equivalent to computing their endomorphism rings [64]. Using the previously mentioned algorithm by Eisenträger et al. [25], the endomorphism rings of supersingular elliptic curves can be computed in $O^*(p^{1/2})$. A connecting isogeny (of rather large degree) can then be computed in classical polynomial time using the KLPT algorithm [40] or a rigorous variant due to Wesolowski [64].

Computing an isogeny of fixed degree This problem is a priori incomparable to the previous one, as extra data is given as input (the existence of a specific degree isogeny) but extra requirements are made on the output. The additional input data is particularly useful for small degrees, and the extra constraint on the output can be handled with variants of the KLPT algorithm for large degrees. In this paper we mostly focus on the "middle" cases, namely isogenies of degrees between $p^{1/2}$ and p^3.

Computing an unknown isogeny of known degree d between two d-isogenous supersingular elliptic curves can always be done using an exhaustive search over all $O(d)$ degree-d isogenies (or equivalently their kernels). In fact, if d is a prime this is the best known method prior to the results of this work.

When d is a smooth integer, a meet-in-the-middle approach with $O^*(\sqrt{d})$ time and memory complexity can be used. However, for large d the memory

requirements become unrealistic. Limiting the available memory leads to the conclusion that a van Oorschot–Wiener collision search whose concrete complexity depends on the amount of memory available is more efficient to compute the isogeny [19].

When the endomorphism rings of the two supersingular curves are known (or have been precomputed), d does not need to be smooth but merely the product of two factors of roughly the same size to make a meet-in-the-middle approach work. This is due to the fact that the isogenies corresponding to the factors, which are potentially of large degree, can be replaced by a powersmooth isogeny using, for instance, the KLPT algorithm [40]. While this approach adds to the overhead of the meet-in-the-middle, the powersmooth isogenies can still be computed in order to find a collision.

Computing endomorphism rings and then using an algorithm such as KLPT to compute a connecting isogeny will in general not return an isogeny of the sought-after degree. However, if $d \approx p^{1/2}$ or shorter, the isogeny is usually the shortest one between the two curves. Galbraith, Petit, Shani and Ti showed how this relative shortness could be exploited to recover the isogeny from the endomorphism rings [29, Sect. 4.2]. They used the fact that the smallest element in the connecting ideal, which can be computed efficiently using the endomorphism rings [39], corresponds to a small degree-d isogeny. This strategy works in polynomial time. The result trivially generalizes to isogenies degrees slightly larger than $p^{1/2}$ by exhaustively searching over all linear combinations of the smallest elements in the connecting ideal, growing exponentially with d.

When d is larger than p, the isogeny is usually not unique. One can compute the endomorphism ring of both curves in time $O^*(p)$, and a connecting ideal in polynomial time. If $d > p^3$ and d has at least two factors, one can then use a variant of the KLPT algorithm [40] to compute an ideal of the correct norm in the same class, and then translate this to a representation of an isogeny.

Computing an isogeny of fixed degree and given action Many isogeny-based protocols reveal the images of torsion points or subgroups through the secret isogeny, including [3, 12, 28, 36] Isogenies of degrees between $p^{1/2}$ and p^3 can sometimes be computed if (masked) torsion point images under the sought-after isogeny are known.

More precisely, assuming knowledge of the endomorphism rings of both curves, Fouotsa, Kutas, Merz and Ti [27] show how to efficiently recover an isogeny of degree $d < \frac{sT}{16}$, where s denotes the degree of the isogeny of smallest degree connecting the two given curves and T the size of the subgroup with known torsion point images. Depending on d, this requires images on a smaller subgroup compared to recent SIDH attacks which allow one to compute a connecting isogeny from the images without requiring the endomorphism rings [11, 53]. Further, an updated version of the reduction by Fouotsa, Kutas, Merz and Ti shows that the reduction still applies if the images of a slightly larger subgroup are given only up to an unknown scalar [26, Thm. 4.2] - a setting where the SIDH attacks are not known to apply.

Thus, when images under the sought-after isogeny are available for a sufficiently large subgroup, an isogeny of degree d could be computed efficiently after obtaining the endomorphism rings in $O^*(p^{1/2})$.

Order embedding problem The order embedding problem can be seen as another variant of the isogeny problem where the goal is to compute endomorphisms with specific traces and norms, i.e. embedding a quadratic order in the endomorphism ring of the curve.

One can look at this problem both in terms of elliptic curves and quaternion algebras. On the elliptic curve side, this looks like a hard problem as [48] implies that finding non-scalar endomorphisms is already as hard as computing endomorphism rings. In [1], it is shown that deciding whether a curve is oriented is subexponentially equivalent to actually finding the orientation.

On the quaternion side, the problem is naturally easier. The importance of the quaternion order embedding problem is highlighted in [65] where the order embedding problem is a missing step for proving equivalence between two hard problems, the endomorphism ring problem and the Uber isogeny problem. Wesolowski further provides a polynomial-time algorithm for orders with discriminant smaller than $\sqrt{d}$. In [1] the authors improve this to discriminants smaller than p under some heuristics.

Quantum algorithms Using quantum computation, some of the algorithms mentioned previously can be accelerated.

When Grover's search [31] is deployed, the endomorphism ring computation from [25] can be run in $O^*(p^{1/4})$ time and constant memory. Similarly, Biasse, Jao and Sankar showed how to accelerate the Delfs–Galbraith algorithm to run in $O^*(p^{1/4})$ [7]. Note that this algorithm can not only be used to find a connecting isogeny of arbitrary degree, but also for endomorphisms by finding loops in the isogeny graph.

To compute degree-d isogenies between two supersingular elliptic curves, Grover's quantum algorithm brings the complexity of the exhaustive search over all degree-d isogenies to $O^*(\sqrt{d})$ with constant memory. If d is prime, this is the best known algorithm prior to this work.

For a sufficiently smooth degree d, Tani's claw finding algorithm with complexity $d^{1/3}$ [60] has been suggested but widely dismissed since it assumes unrealistic costs of accessing memory. This has, for example, been pointed out by Jaques and Schanck [37]. They argued that it is more efficient to use the classical hardware dedicated to access memory for Tani's algorithm for a classical attack instead. In particular, Tani's algorithm does not seem to lead to a quantum speed-up.

4 General strategy: from finding isogenies to solving norm equations

Let E_1, E_2 be two given supersingular elliptic curves over $\mathbb{F}_{p^2}$ which are connected by an unknown isogeny $\varphi : E_1 \to E_2$ of degree d. Our aim is to provide improved algorithms for finding such a degree-d isogeny between E_1 and E_2.

Before we introduce our general strategy to recover this unknown isogeny, i.e. to solve Problem 1.1, in more detail, we will prove a lemma which will serve as a crucial tool in our approach. Let I be a connecting ideal between maximal orders $\mathcal{O}_1$ and $\mathcal{O}_2$ where $\mathrm{End}(E_1) \cong \mathcal{O}_1$ and $\mathrm{End}(E_2) \cong \mathcal{O}_2$. Let the norm of I be n_I. As shown in [40, Lem. 5], a degree-d isogeny between E_1 and E_2 corresponds to an element $\alpha \in I$ whose norm is $n_I d$. Hence we require a solution to the following problem to find the desired isogeny using e.g. [50].

Problem 4.1. Let $\mathcal{O}_1, \mathcal{O}_2$ be maximal orders in the quaternion ramified at p and ∞, $B_{p,\infty}$ and let I be a connecting ideal of $\mathcal{O}_1$ and $\mathcal{O}_2$. Find an element of norm $n_I d$ in I, if it exists.

Finding an ideal element of the required norm essentially implies that we need to solve a norm equation, namely $Q(x_1, x_2, x_3, x_4) = n_I d$, where Q is a norm form associated to the ideal. The form Q is only determined up to integral equivalence, meaning that a different choice of basis of I will provide a different Q. Note that $Q(x_1, x_2, x_3, x_4)$ can be written as $(x_1\, x_2\, x_3\, x_4)G(x_1\, x_2\, x_3\, x_4)^T$, where G is the associated Gram matrix. The (i, j)-th entry of this Gram matrix, denoted by g_{ij}, is $\langle \sigma_i, \sigma_j \rangle = \mathrm{tr}(\sigma_i \overline{\sigma_j})$ for a basis $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ of I and tr the trace. This Gram matrix has the property that every entry is an integer divisible by n_I, hence it makes sense to instead work with the reduced norm form (and Gram matrix), where every entry is divided by n_I. Then one is looking to represent the integer d instead of $n_I d$, and we use the normalized notions in the following. Henceforth, we will refer by Q to the norm form normalized by n_I, i.e. we aim to solve

$$Q(x_1, x_2, x_3, x_4) = d. \tag{1}$$

If we choose an LLL-reduced basis of the ideal then we can bound any potential solution $(x_1, \ldots, x_4)$ componentwise for a representation of d in the corresponding norm form as shown in the following lemma.

Lemma 4.2. *Let $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ be an LLL-reduced basis of the ideal I. Let Q be the associated reduced norm form. Then if $Q(x_1, x_2, x_3, x_4) = d$ has a solution, we have bounds*

$$|x_i| \leq 8\sqrt{\frac{d}{\mathrm{Norm}(\sigma_i)}}.$$

Proof. First let G be a symmetric positive definite real matrix. Then it has Cholesky decomposition of the form $G = B^T B$. Now one can equip the lattice $\mathbb{Z}^4$ with the inner product whose Gram matrix is G. Let us denote this lattice by L_G. Let L be the lattice generated by the matrix B together with the standard

inner product. It is easy to see that L and L_G are isometric lattices via the map that sends a vector $(x_1\,x_2\,x_3\,x_4) \in \mathbb{Z}^4$ to $B(x_1\,x_2\,x_3\,x_4)$.

In our specific case of interest, let G be the Gram matrix corresponding to the LLL-reduced basis $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ with Cholesky decomposition $G = B^T B$. By the above observation, B is LLL-reduced with respect to the usual Euclidean norm. This means we want to find x_1, x_2, x_3, x_4 such that $|B(x_1\,x_2\,x_3\,x_4)^T| = \sqrt{d}$ where $\|\cdot\|_2$ denotes the Euclidean norm. Since the basis of G is LLL-reduced, we can conclude that B is LLL-reduced with respect to the usual Euclidean norm. The lemma follows from the observation that the Euclidean norm of the i-th column of B is $\sqrt{\langle \sigma_i, \sigma_i \rangle}$.Finally, note that the constant in this lemma depends on the parameters used in the LLL reduction. We assume the choice originally made by Lenstra, Lenstra and Lovász [43]. □

For simplicity of our exposition, we will assume the following.

Assumption 4.3. $\mathrm{Norm}(\sigma_i) \approx \sqrt{p}$ *for* $i = 1, \ldots, 4$.

Note that the lattice $\mathrm{Hom}(E_1, E_2)$ has determinant p^2 [62, Cor. III.5.3.] and thus the assumption means that all four successive minima have roughly the same size which happens with overwhelming probability. Now we can state the following corollary using the same constant as in Lemma 4.2 and taking $d = p^{1/2+\epsilon}$ as before.

Corollary 4.4. *Suppose that Assumption 4.3 holds. Then*

$$|x_i| \leq c \cdot \sqrt{\frac{d}{p^{1/2}}} = c \cdot p^{\epsilon/2}$$

for the constant $c = 8$.

The general strategy for finding a degree-d isogeny between two supersingular elliptic curves which we will adhere to in this article is as follows:

1. Compute the endomorphism rings $\mathcal{O}_1$ and $\mathcal{O}_2$ of E_1 and E_2.
2. Find a connecting ideal I between $\mathcal{O}_1$ and $\mathcal{O}_2$.
3. Compute an LLL-reduced Gram matrix G of the ideal I.
4. Divide every entry of G by $\mathrm{Norm}(I)$ and compute the associated quadratic form Q.
5. Solve the Diophantine equation $Q(x_1, x_2, x_3, x_4) = d$.
6. Using [40, Lem. 5] and the solution to Q, compute an ideal J equivalent to I such that $\mathrm{Norm}(J) = d$.
7. Translate J to an isogeny between E_1 and E_2.

With the exception of Step 5, the complexity is known for the different tasks of our general strategy: Endomorphism rings can be computed classically in time $O^*(p^{1/2})$ or on a quantum computer in $O^*(p^{1/4})$ using the algorithm by Eisenträger et al. [25] and its quantum version (or alternatively, the Delfs–Galbraith algorithm [22]). A connecting ideal I can also be found efficiently, for instance using the algorithm of Kirschmer and Voight [39].

Thus, for the remainder of this article, we concentrate on specifically studying Step 5 in detail: solving $Q(x_1, x_2, x_3, x_4) = d$, where Q is a quadratic form such that every entry is an integer with approximate absolute value $\sqrt{p}$ and $|x_i|$ can be bounded by a small constant multiple of $\sqrt{\frac{d}{p^{1/2}}}$. We explore different avenues for solving Diophantine equations with the given restrictions. More explicitly, in Section 5 we describe an algorithm which solves Problem 4.1 for $d \approx p^{1/2+\epsilon}$ and some $\epsilon > 0$ on a quantum computer in roughly $O^*(p^{1/4} + \epsilon)$ with high probability or returns no solution, see Theorem 5.5. In Section 6, we present another algorithm based on a bivariate Coppersmith method which solves Problem 4.1 for $d \approx p^{1/2+\epsilon}$, $0 < \epsilon < 1/2$, in time $O^*(p^{1/2})$ on a classical and $O^*(p^{1/4})$ on a quantum computer.

5 Solving the norm equation with Cornacchia's algorithm

Generally, solving Diophantine equations with four variables like

$$Q(x_1, x_2, x_3, x_4) = d \tag{1}$$

is not straightforward. In this section, we will focus on one particular way of finding a solution to Equation (1), i.e. solving Step 5 of our general strategy. In our approach we first guess two of the variables and then solve the remaining bivariate equation using Cornacchia's algorithm. In Section 6, we will describe an alternative approach using Coppersmith's methods and some of its variants to solve the multivariate equations resulting from guessing at most two variables.

We begin by making random guesses for two of the variables in Equation (1), say $x_3 =: k$ and $x_4 =: l$, within the bounds given by Assumption 4.3 and Corollary 4.4. Guessing the two variables correctly will contribute $O(p^\epsilon)$ to the complexity classically, or $O(p^{\epsilon/2})$ using Grover's quantum search algorithm [31].

For each guess, it remains to solve the resulting quadratic bivariate equation or determine that no such solution exists. Assuming we guess k and l for x_3 and x_4, where both values are bounded by $c \cdot p^{\epsilon/2}$ with the constant c stemming from explicit choices made during the LLL reduction, the remaining equation to be solved is

$$\begin{aligned} f(x_1, x_2) &= Q(x_1, x_2, k, l) - d \\ &= g_{11}x_1^2 + g_{22}x_2^2 + 2g_{12}x_1x_2 && \text{(quadratic)} \\ &+ (2g_{13}k + 2g_{14}l)x_1 + (2g_{23}k + 2g_{24}l)x_2 && \text{(linear)} \\ &+ \big(2g_{34}kl + g_{33}k^2 + g_{44}l^2 - d\big), && \text{(constant)} \end{aligned}$$

where the g_{ij} denote the entries of the Gram matrix G and stem from its corresponding inner product defined on our lattice as described in Section 4.

Remark 5.1. Technically, f is a family of functions $f_{k,l}$ where each function depends on the specific values guessed for x_3 and x_4. To improve notation and readability, we implicitly assume that f (and the associated values D, E, F, x, y

and N defined below) all depend on the k and l which are fixed in the context where f is used.

Performing a change of variables similar to [56] (originally attributed to Lagrange) allows us to rewrite $f(x_1, x_2) = 0$ as an equation of the form

$$x^2 - Dy^2 = N \tag{2}$$

due to the following. Let f_{ij} denote the coefficient of $x^i y^j$ in f. The bivariate quadratic f can, in a first step, be transformed into an equation of the form

$$Dy^2 = (Dx_2 + E)^2 + DF - E^2, \tag{3}$$

where the new variable y is defined as $y := 2f_{20}x_1 + f_{11}x_2 + f_{10}$ and the substitutions

$$\begin{aligned} D &:= f_{11}^2 - 4f_{20}f_{02}, \\ E &:= f_{11}f_{10} - 2f_{20}f_{01}, \text{ and} \\ F &:= f_{10}^2 - 4f_{20}f_{00} \end{aligned}$$

are performed. In a second step, a new variable $x := Dx_2 + E$ is introduced and N defined as $N := E^2 - DF$ to facilitate rearranging Eq. (3) again into the desired form of Eq. (2).

Examining the coefficient values in our new quadratic equation, Equation (2), obtained from the change of variables leads us to several observations: Firstly, we can see that the size of N can be bounded polynomially in the absolute value of the largest entry of G (more precisely $N \in O(\max(g_{ij})^4 p^\epsilon)$. Secondly, we show that $D = -4\left(g_{11}g_{22} - g_{12}^2\right)$ is always negative as a consequence of the symmetric and positive definite nature of the Gram matrix G. Hence, Equation (2) has only finitely many solutions. In particular, when looking for a fixed-degree isogeny, we expect there to usually be a unique solution. Either way, we only require a single solution to obtain one isogeny of prescribed degree.

Such a solution can be found using Cornacchia's algorithm (see e.g. [47, Alg. 1]) as long as N does not have too many prime factors, as it requires finding (all) square roots of $D \pmod N$. Finding these square roots becomes expensive if N has too many distinct factors. More precisely, we choose to abandon a pair of guesses x_3, x_4 when factoring N reveals that N has more than $B \log\log N$ distinct prime factors for some fixed $B \in \mathbb{Z}$. To estimate the probability of this event, we use the following result.

Lemma 5.2. *Let N be an integer as in Equation (2) and let $B \in \mathbb{Z}_{>1}$. Under the heuristic assumption that the number of prime divisors of N behave as predicted by standard asymptotics for sufficiently large integers, we expect N to have more than $B \log\log N$ prime factors with probability smaller than $\frac{1}{2(B-1)^2}$.*

Proof. Let $\omega : \mathbb{N} \to \mathbb{N}$ be the function which maps a positive integer to its number of distinct prime divisors. Asymptotically, the distribution of $\omega(n)$ is a

normal distribution around the mean $B_1 + \log\log n$, where $B_1 \approx 0.261$ is the Mertens constant, with standard deviation $\log(\log n)^{1/2}$, see e.g. [32, Sect. 22.11] or [52].

Under the heuristic that N is large enough for these asymptotics to apply and that its number of prime factors behaves as predicted for a random integer of roughly the same size, we can use Chebyshev's inequality to get the bound

$$\Pr\left(\omega(N) - B_1 > B\log\log N\right) \leq \frac{1}{2B^2\log\log N}.$$

Here, we used that the normal distribution is symmetric around $B_1 + \log\log N$ and that the standard deviation is $\log(\log N)^{1/2}$. Since we are interested in N for which $\log\log N > 1$, we can very crudely estimate our bound by

$$\Pr(\omega(N) > B\log\log N) \leq \frac{1}{2(B-1)^2}. \quad \square$$

Note that taking a larger B to bound the number of prime factors of N, $B\log\log N$, accepted in our algorithm will increase the concrete cost of running Cornacchia's algorithm.

Remark 5.3. Assume that the asymptotic heuristics hold for all the N sampled by fixing x_3, x_4 for a fixed basis and assume that the correct solution is randomly distributed among these trials. Taking for instance $B = 11$, we expect to find a solution in $> 99\%$ of cases after iterating through all guesses for a fixed basis by Lemma 5.2. However, it may be possible that the correct solution $(x_1, \ldots, x_4)$ with respect to some fixed basis gives an N with too many prime factors. We accept this as the *failure probability* of our algorithm.

These observations lead us to the following proposition.

Proposition 5.4. *Assume $\omega(N) \leq B\log\log N$, i.e. N is not too smooth and has at most $B\log\log N$ prime factors. One can find a solution to the equation $f(x_1, x_2) = 0$ in quantum polynomial time, if it exists, or determine that there is no such solution.*

Proof. The main observation is that since G is a positive definite matrix, its leading principal minors are positive. Hence we have that $g_{12}^2 - g_{11}g_{22} < 0$ which implies that $D = (2g_{12})^2 - 4g_{11}g_{22} < 0$. Therefore, it is possible to use the above change of variables to reduce solving $f(x_1, x_2) = 0$ to solving $x^2 - Dy^2 = N$, where the size of N is polynomial in the size of G (i.e. the size of the absolute value of the largest entry). One can use Shor's algorithm [57] to factor N and then apply Cornacchia's algorithm to solve $x^2 - Dy^2 = N$. Reversing the substitutions leads to a solution for $f(x_1, x_2) = 0$. Note that if a guess k, l is incorrect, then $f(x_1, x_2) = 0$ will have no solution. Fortunately, running Cornacchia's algorithm helps us detect efficiently if no solution exists, see e.g. [13, Sect. 1.5.2]. $\square$

Algorithm 1: Using Cornacchia to recover element of reduced norm d in connecting ideal I

Input: Let $\mathcal{O}_1, \mathcal{O}_2$ be maximal orders in $B_{p,\infty}$ and let I be their connecting ideal containing an element of reduced norm d, where $d \approx p^{1/2+\epsilon}$. Let $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ be an LLL-reduced basis of I with $\|\sigma_i\| \approx p^{1/4}$. Finally, let $G = (g_{ij})$ be the corresponding Gram matrix, and $B \in \mathbb{Z}_{>1}$.

Output: $x_1, x_2, x_3, x_4 \in \mathbb{Z}$ such that $|x_i| \leq c \cdot p^{\epsilon/2}$ for $i = 1, \ldots, 4$ and $Q(x_1, x_2, x_3, x_4) = \|\sum_{i=1}^{4} x_i \sigma_i\| = d$.

for $(k, l) \in \{0, \pm 1, \ldots, \pm c \cdot p^{\epsilon/2}\} \times \{0, \pm 1, \ldots, \pm c \cdot p^{\epsilon/2}\}$ **do**
  $D \leftarrow 4(g_{12}^2 - g_{11}g_{22})$, $E \leftarrow 4(g_{12}(g_{13}k + g_{14}l) - g_{11}(g_{23}k + g_{24}l))$, $F \leftarrow 4((g_{13}k + g_{14}l)^2 - g_{11}(2g_{34}kl + g_{33}k^2 + g_{44}l^2 - d))$, $N \leftarrow E^2 - DF$;
  Factor N using either a classical algorithm or Shor's quantum algorithm;
  if N has more than $B \log\log N$ factors **then**
    continue
  else
    Run Cornacchia's algorithm to find solutions of $x^2 - Dy^2 = N$;
    if Cornacchia returns a solution (x, y) **then**
      $x_2 \leftarrow (x - E)D^{-1}$, $x_1 \leftarrow (2g_{11})^{-1}(y - 2(g_{12}x_2 + g_{13}k + g_{14}l))$;
      $x_3 \leftarrow k$, $x_4 \leftarrow l$;
      return x_1, x_2, x_3, x_4

Theorem 5.5. *Let $\mathcal{O}_1, \mathcal{O}_2$ be maximal orders in $B_{p,\infty}$ and let $d \approx p^{1/2+\epsilon}$ for some $\epsilon > 0$ and let $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ be an LLL-reduced basis of the connecting ideal I such that $\|\sigma_i\| = p^{\alpha_i}$ and $\alpha_i \approx 1/4$. Then Algorithm 1 computes an element of reduced norm d in I, i.e. solves Problem 4.1 for the given parameters, in time $O^*(p^{\epsilon/2})$ on a quantum computer, $O^*(p^{\epsilon}) \cdot L_{\log p}(1/3)$ classically or returns no solution. The algorithm fails to find an existing solution with probability smaller than $1/2(B-1)^{-2}$ under the heuristics of Lemma 5.2, where the probability is taken over the possible choices of LLL-reduced lattices and $B \log\log N$ is the number of prime factors allowed in Step 4 of Algorithm 1.*

Proof. Most of the proof is already covered by previous discussion, nevertheless we briefly recap the main points. The complexity of guessing two variables is $O(p^{\epsilon})$ in the classical case and reduced to $O(p^{\epsilon/2})$ using Grover's quantum search. This follows from Corollary 4.4 as $|x_i| < cp^{\epsilon/2}$. Once we make our guess we are left with a bivariate quadratic equation which we transform into an equation of the form $x^2 - Dy^2 = N$ where $D < 0$. In order to solve this using Cornacchia's algorithm we need to factor N, which has complexity $L_{\log p}(1/3)$ classically and polynomial complexity using Shor's quantum algorithm. We also need to compute all square roots of D modulo N. In every iteration after factoring N we abort if N has more than $B \log\log N$ prime factors, hence computing all square roots can be accomplished in polynomial time using the Tonelli-Shanks algorithm. This proves the complexity claims of the Theorem. Failure occurs if

for the correct solution the corresponding N has too many prime factors. Thus Lemma 5.2 implies the failure probability of the theorem.

□

Remark 5.6. The analysis of Theorem 5.5 raises the question whether the correct guesses for x_3 and x_4 with respect to different bases leads to N with distinct prime factorisation respectively. Experimentally, we re-randomised multiple bases using unimodular matrices and indeed the resulting N corresponding to the correct guesses with respect to the respective bases were different, did in general neither share the same factors nor have the same number of distinct prime factors.

Similarly, one could also just guess values for a different pair of x_i (instead of x_3 and x_4) to obtain a different N. For basis vectors *all* of size roughly $p^{1/2}$ this would not affect the algorithm's complexity.

Remark 5.7. Our calculations assume a generic case where the shortest isogeny between two curves has degree approximately $\sqrt{p}$. However, if there exists shorter isogenies between the two curves, then our algorithms are actually faster if we guess the variables corresponding to the longer isogenies in the LLL-reduced basis. Indeed, this follows from the fact that the volume of the lattice is always the same and if the shortest vector is unexpectedly short, then due to the near-orthogonal property of an LLL basis the other vectors have to be longer.

6 Solving the norm equation with Coppersmith's methods

In this section, we describe a slightly different approach to solve Problem 4.1. The first step is the same as in Section 5: We compute the reduced norm form with respect to an LLL-reduced basis, and our goal is still to represent the integer d, i.e. to find a solution x_1, x_2, x_3, x_4 such that $Q(x_1, x_2, x_3, x_4) = d$ given the same bounds on the x_i as before.

As an alternative to solving the equation using Cornacchia's algorithm, we apply several variants of Coppersmith's techniques to compute short solutions of polynomial equations. In Sections 6.1 and 6.2, we provide theoretical analyses highlighting for which isogeny degrees our different methods should work, and provide some experimental results in Section 6.3.

6.1 Guessing two variables

Again, we assume the same starting point as in Section 5. Recall that $G = (g_{ij})$ is the Gram matrix of the reduced norm form of the ideal I with corresponding basis $\sigma_1, \ldots, \sigma_4$. As before, we assume the generic case [29], where $\|\sigma_i\| = p^{1/4}$.

For $d \approx p^{1/2+\epsilon}$, where $\epsilon > 0$, we know that the components of a correct solution are bounded above by $|x_i| < c \cdot p^{\epsilon/2}$ according to Corollary 4.4 as we started with a reduced basis. Again, we guess two variables as in the previous

section with classical cost $O^*(p^\epsilon)$, or $O^*(p^{\epsilon/2})$ on a quantum computer. We will solve the remaining bivariate quadratic equation, denoted by $f(x_1, x_2)$ as in Section 5, following Coron's approach to Coppersmith's methods [17].

The nature of f implies that we can consider Theorem 2.4 with $\delta = 2$. Hence, Coron's algorithm should be able to find a solution to $f(x_1, x_2)$ (or detect that no solution exists) whenever $XY < W^{1/2}$, where $|x_1| < X$,$|x_2| < Y$ and $W = \max\{|f_{ij}|X^iY^j\}$. Here, f_{ij} denotes the coefficient of $x_1^i x_2^j$, and $X \approx Y \approx p^{\epsilon/2}$ by the bounds provided. Since the Gram matrix is reduced, it follows from our assumptions that $g_{ij} \approx \sqrt{p}$ which in turn implies that $W \approx p^{1/2+\epsilon} \approx d$. Now the condition $XY < W^{1/2}$ translates to $p^\epsilon < p^{1/4+\epsilon/2}$. Hence we can conclude that Coron's algorithm will be successful for $\epsilon < 1/2$, that is for ideals with norms between $p^{1/2}$ and p.

For each guessed pair of variables, Coron's algorithm runs in time polynomial in $\log W$, i.e. polynomial in $\log p$. We summarize our results in the following theorem.

Theorem 6.1. *Let $\mathcal{O}_1, \mathcal{O}_2$ be maximal orders in $B_{p,\infty}$. Let $d \approx p^{1/2+\epsilon}$ for some $0 < \epsilon < 1/2$. Further, let $\sigma_1, \ldots, \sigma_4$ be an LLL-reduced basis of the ideal I connecting $\mathcal{O}_1$ and $\mathcal{O}_2$ such that $\deg(\sigma_i) = p^{\alpha_i}$ with $\alpha_i \approx 1/4$. Then there exists an algorithm that computes an element of reduced norm d in I in time $O^*(p^\epsilon)$ classically or $O^*(p^{\epsilon/2})$ on a quantum computer, or determines that no such element exists.*

For Coron's method to achieve the results of Theorem 6.1, we assume that the shortest element in $\mathrm{Hom}(E_1, E_2)$ has degree approximately $\sqrt{p}$. The total cost of the entire algorithm is the same as that of the approach using Cornacchia's algorithm as its complexity is dominated by the guessing of variables and the endomorphism ring computations. The advantage of this approach is that it does not have a failure probability as in Theorem 5.5 and thus does not rely on non-standard heuristics such as the assumptions made in Lemma 5.2.

6.2 Guessing one variable

Next, we consider the case of guessing only a single variable and we explain for which ϵ, where $d = p^{1/2+\epsilon}$, we expect the approach to work.

Using our previous notation, we have $W \approx Q(x_1, x_2, x_3, x_4) = d = p^{1/2+\epsilon}$, where Q is again considered with respect to a reduced basis, i.e. the components of the solution are bounded by $x_i \approx p^{1/4}$.

Due to the symmetry in the set of monomials appearing in the norm equation, we focus on sets $\mathcal{S}$ that are invariant under permutations of variables (see the table below for examples). In particular, we have $s_x = s_y = s_z$ for these $\mathcal{S}$.

Neglecting the constant depending on the parameters used in LLL, which asymptotically (i.e. for simultaneously increasing values of p and d) only contribute to a small constant, Theorem 2.5 states that

$$X^{3s_x} < W^s.$$

Using the estimate $|x_i| \approx p^{\epsilon/2}$ for X and $W = p^{1/2+\epsilon}$, we have

$$3s_x\epsilon/2 < (1/2+\epsilon)s$$

giving us the estimate

$$\epsilon < \frac{s}{3s_x - 2s}.$$

Table 1 below provides values for s, $s_x = s_y = s_z$, s and $\frac{s}{3s_x-2s}$ for some a priori plausible symmetric sets $\mathcal{S}$. Regarding the last row, while increasing D ostensibly improves the above estimate, the matrix M_1 defining the lattice L_1 grows significantly as D increases, making LLL reduction much slower. Therefore despite the algorithm still technically being polynomial time for any fixed D, in practice we keep $D = 1$ (the first row of the table), because it is faster. The bound ϵ_D for ϵ is increasing as we increase D, and converges towards 0.25 as we send D to infinity; these values were computed using MATHEMATICA [35]. The values grow rapidly at first, e.g. for $D = 3$ we have $\epsilon_D = 0.16$, but the growth decelerates quickly, e.g. the first ϵ_D that surpasses 0.24 is ϵ_{53}.

Symmetric set $\mathcal{S}$	s	$s_x = s_y = s_z$	$\frac{s}{3s_x-2s}$
$\{1, x, y, z\}$	4	14	0.1176
$\{1, xy, xz, yz\}$	4	26	0.0571
$\{1, x, y, z, xy, xz, yz\}$	7	28	0.1000
$\{1, x, y, z, xy, xz, yz,$ $x^2, y^2, z^2\}$	10	30	0.1429
{monomials with total degree $\leq$ D}	$\sum_{i=0}^{D} \binom{i+2}{2}$	$\sum_{i=0}^{D+1}(D+2-i)\binom{i+1}{1} + \sum_{i=0}^{D}(D+1-i)\binom{i+1}{1}$	ϵ_D

Table 1. Values for plausible symmetric sets. The bound ϵ_D converges to 0.25 as D grows to infinity.

The cost of this approach is the cost of guessing one variable multiplied with the cost of running Coppersmith's algorithm once. There is a trade-off in the number of monomials to be included, since adding more monomials leads to higher complexity but a wider range of applicability.

For each guessed variable, the trivariate algorithm of Bauer and Joux runs in time polynomial in $\log p$, and the argument for the bivariate version of Coron's algorithm naturally extends to the trivariate version. We summarize our results in the following theorem:

Theorem 6.2. *Let $\mathcal{O}_1, \mathcal{O}_2$ be maximal orders in $B_{p,\infty}$. Let $d \approx p^{1/2+\epsilon}$ for some $0 < \epsilon < 1/4$. Further, let $\sigma_1, \ldots, \sigma_4$ be an LLL-reduced basis of the ideal I connecting $\mathcal{O}_1$ and $\mathcal{O}_2$ such that $\deg(\sigma_i) = p^{\alpha_i}$ with $\alpha_i \approx 1/4$. Then there exists*

an algorithm that computes an element of reduced norm d in I in time $O^*(p^{\epsilon/2})$ classically or $O^*(p^{\epsilon/4})$ on a quantum computer, or determines that no such element exists.

6.3 Experimental results

We implemented the different approaches to solve the norm equation and all of our code is available at `https://github.com/isogeny-finding/improved-isogeny-finding`.

Code and instance generation In the experiments, we used MAGMA [10] to generate maximal orders and connecting ideals containing an element with increasing reduced norm using random walks, then transformed them into the corresponding quadratic forms. We tested our implementations on randomly generated large primes ranging from 100 to 3000 bit-length, with a hundred quadratic form instances per prime and per ideal norm. We then gradually increased the ideal norm and recorded the maximal norm for which the method used successfully computed the roots in all tested instances.

This approach to generating instances also yields a solution, which we use to avoid guessing when working with large parameters, as it would be too computationally expensive. Instead, we pick one variable we consider known (i.e. correctly guessed) and then use our implementations of Coppersmith's methods to solve the form for the remaining three variables, which we can then compare with the known solution to test for correctness. In particular, once the Bauer–Joux or Coron's approaches find enough additional polynomials, we try to obtain the root by computing resultants, which simultaneously checks for algebraic dependence. As with many lattice reduction applications, the approaches seem to work better in practice than in theory.

Trivariate case Our SAGEMATH [61] implementations of the trivariate Bauer–Joux approach and the trivariate Coron approach find connecting ideals between two maximal orders $\mathcal{O}_1$ and $\mathcal{O}_2$ containing an element of reduced norm up to approximately $2^{0.67l}$ where l is the bit-length of the prime p.

As mentioned in Section 2.3, we aim to avoid the costly second step of the Bauer–Joux algorithm which entails a Gröbner basis computation and another LLL reduction; see e.g. [6]. Therefore, we consider other LLL-reduced and orthogonalized vectors $\{b_1^\star, \ldots, b_{r-1}^\star\}$ in reverse order instead of only working with the single vector $b_r^\star$. In particular, we check if any of them already yields another polynomial P_2 that annihilates the desired root and is algebraically independent from P and P_1, which was obtained from $b_r^\star$, by immediately trying to extract the common root of P, P_1, P_2. As with $b_r^\star$, this is guaranteed if $\|s_0\| < \|b_i^\star\|$, but can happen regardless. If this fails, we continue with the second step of the Bauer–Joux approach. As mentioned in Section 6.2, the set $\mathcal{S}$ we used was $\{1, x, y, z\}$ (i.e. first row or $D = 1$ in Table 1).

In the trivariate version implementation of Coron's approach, the parameter that adjusts the lattice dimension (analogous to the parameter k in the bivariate

approach) was set to zero for efficiency reasons, as the LLL reduction is much slower for any higher value of the parameter.

The results obtained for Coron's approach and the approach by Bauer and Joux are presented in Fig. 1 and Fig. 2, respectively. In both figures p denotes the prime used, and D denotes the maximal of ideal norms where the method was successful in all tested instances. We note that the maximum expected ratio $\frac{\log_2(D)}{\log_2(p)}$ for the Bauer–Joux approach with the set $\mathcal{S}$ we used is approximately 0.62 according to Section 6.2, so the experiments do in fact perform better than the theory predicts.

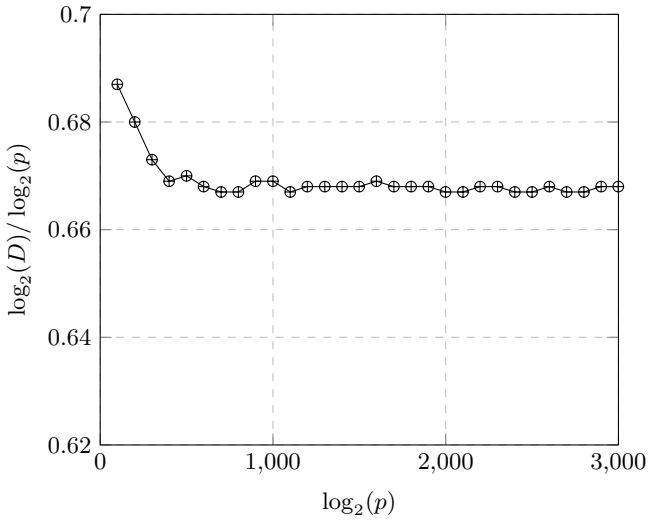

Fig. 1. The maximal ratio of bit-lengths of the ideal norm D and the prime p using the trivariate approach of Coron.

7 Hybrid algorithms

In the previous sections, we have established several new approaches for solving the norm equation relating to Problem 4.1. Further, we described how we can translate our results from the quaternion setting into explicit representations of isogenies which eventually present a solution to Problem 1.1. Before stating explicitly in which ranges of parameters these algorithms provide a speed-up over the currently best-known methods for fixed-degree isogeny finding in Section 8, we will now present another way in which to broaden the set of parameters our techniques apply. This approach slightly diverges from our general strategy outlined in Section 4 as we combine guessing parts of the initial isogeny with the previously described algorithms. We call this a *hybrid* approach. Note that

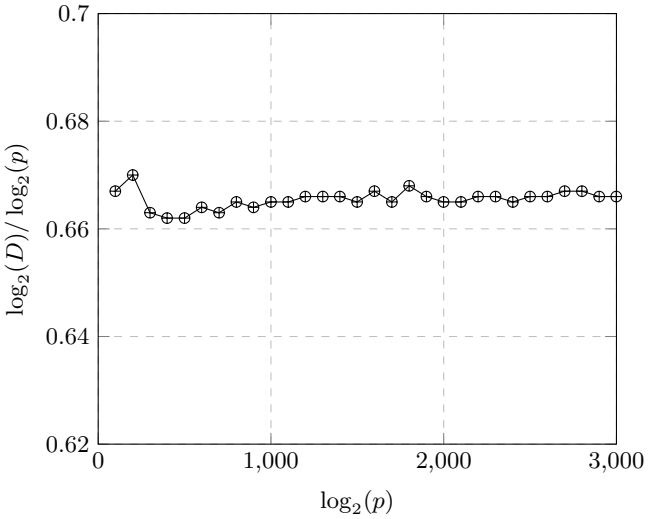

Fig. 2. The maximal ratio of bit-lengths of the ideal norm D and the prime p using the approach of Bauer and Joux.

the following approach only works when considering isogeny degrees that are sufficiently smooth.

Let ℓ be a small prime. Let us suppose that we are looking for an isogeny $\psi : E_1 \to E_2$ of degree $d = \ell^e \approx p^{1/2+\epsilon}$ where ϵ is too large for the attacks of Section 5 or Section 6 to improve upon the state of the art. We can use a combination of guessing the isogeny and solving the norm equation corresponding to the remaining isogeny to optimize the runtime of finding ψ. As before, we first compute the endomorphism rings of E_1 and E_2, giving us $\mathcal{O}_1$ and $\mathcal{O}_2$. We then guess a sufficiently large part of the isogeny, say $\psi' : E_1 \to E'$ for some elliptic curve E' which is d'-isogenous to E_1 with d' dividing d. The following discussion makes the optimal proportions of guessing more precise.

By translating the endomorphism ring knowledge to E' via ψ' to obtain $\mathcal{O}'$, we avoid adding additional costly endomorphism ring computations and have reduced our initial problem down to one with more favourable parameters. It remains to solve the problem of finding a d/d'-isogeny ψ'' between E' and E_2 with one of our Coppersmith variants from Section 6. If no isogeny is found, the guess for ψ' was incorrect and another candidate for the d'-isogeny should be tested. Once a solution pair ψ' and ψ'' is found, we have found an isogeny

$$\psi := \psi'' \circ \psi'$$

between E_1 and E_2 which is of degree $d = d' \cdot d/d' = \deg \psi' \cdot \deg \psi'' = \deg \psi$ as required.

Next, we discuss for which parameters we can utilize the trivariate Coppersmith approach from Section 6.2. When guessing a d'-isogeny emanating from

E_1 to some curve E', where we choose d' to be a factor of d, approximately of size $p^{\epsilon-1/4}$. The remaining isogeny should have degree $p^{3/4}$. We can apply the trivariate Coppersmith approach which either returns a solution or fails, in which case we make a new guess. The following proposition estimates the classical cost of this approach.

Proposition 7.1. *Let E_1, E_2 be supersingular elliptic curves over $\mathbb{F}_{p^2}$ which are d-isogenous, where $d = \ell^k$ and $d = p^{1/2+\epsilon}$ for some $\epsilon > 1/4$. There exists a classical algorithm that finds an isogeny of degree d between E_1 and E_2 with complexity $O^*(\max\{p^{1/2}, p^{\epsilon-1/8}\})$.*

Proof. First one computes the endomorphism rings of E_1 and E_2 which has complexity $O^*(p^{1/2})$. Guessing an isogeny and translating endomorphism rings can be accomplished in polynomial time. The number of isogenies to be guessed is $O^*(p^{\epsilon-1/4})$. Then one has to guess one more variable, the cost of that is $p^{1/8}$ according to Corollary 4.4. Then Theorem 6.2 implies that the total complexity is as claimed. □

Note that the hybrid approach can also be used on a quantum computer. Using quantum algorithms to compute the endomorphism rings and using Grover's search when guessing parts of the isogeny, one obtains a time complexity of $O^*(\max\{p^{1/4}, p^{\epsilon/2}\})$. However, this does not provide a speedup compared to our strategy using Cornacchia's algorithm described in Section 5.

8 Results

In this section, we compare the state of the art with our new results. The costs of isogeny-finding using our various techniques are summarized in Table 2.

8.1 Classical algorithms

As seen in the table, we distinguish between smooth-degree isogenies and those of non-smooth degree. In the case of non-smooth degrees, our algorithms return an isogeny representation as introduced in [44], i.e. a way to efficiently evaluate the isogeny at any point.

Smooth degrees If d is smooth, we compare our methods to meet-in-the-middle. The cost of meet-in-the-middle algorithms for an isogeny of degree $p^{1/2+\epsilon}$ is $p^{1/4+\epsilon/2}$. Since every approach involves endomorphism ring computations, we will only consider $\epsilon \geq 1/2$. It is easy to see from Table 2 that the best method for smooth degrees is our hybrid algorithm from Section 7. In order to observe an improvement, we require $\epsilon - 1/8 < 1/4 + \epsilon/2$, i.e. $\epsilon < 3/4$. Thus our methods should be asymptotically more powerful in the case where $p \leq d \leq p^{5/4}$.

Method	Cost (classical)	Cost (quantum)	Condition on size
State-of-the-art (general d)	$\frac{1}{2}+\epsilon$	$\frac{1}{4}+\frac{\epsilon}{2}$	-
State-of-the-art (large d)	$\frac{1}{2}$	$\frac{1}{4}$	$\epsilon > 5/2$
State-of-the-art (smooth d)	$\frac{1}{4}+\frac{\epsilon}{2}$	$\frac{1}{4}+\frac{\epsilon}{2}$	-
Cornacchia (Section 5)	$\max\{\frac{1}{2},\epsilon\}\log_p(L[\frac{1}{3}])$	$\max\{\frac{1}{4},\frac{\epsilon}{2}\}$	-
Coppersmith bivariate (Section 6.1)	$\frac{1}{2}$	$\frac{1}{4}$	$\epsilon < 0.5$
Coppersmith trivariate (Section 6.2)	$\frac{1}{2}$	$\frac{1}{4}$	$\epsilon < 0.25$
Hybrid approach (smooth d) (Section 7)	$\max\{\frac{1}{2},\epsilon-\frac{1}{8}\}$	$\max\{\frac{1}{4},\frac{\epsilon}{2}\}$	$\epsilon > 1/4$

Table 2. Costs of different approaches to find isogenies of given degree $d = p^{1/2+\epsilon}$ given as logarithm in base p, and (empirical) conditions for the algorithms to work. Our techniques improve classical costs for generic d and quantum costs for generic and smooth d whenever $0 < \epsilon < 5/2$.

Unlike general meet-in-the-middle algorithms, however, our algorithm is completely memory-free as well as parallelizable. Thus, even though our bivariate Coppersmith and hybrid algorithms match the time complexity of meet-in-the-middle in the case where $d \approx p$, they provide the significant benefit of requiring no memory. In particular the situation where $d \approx p$ is an important special case as heuristically there should be a degree-d isogeny between any two curves.

Non-smooth degrees All our algorithms have the same complexity when d is not smooth. In this case, we compare only to isogenies which have degree less than p^3. Therefore we have to examine the inequalities $\epsilon - 1/8 < 1/2 + \epsilon$ and $1/2 < 1/2 + \epsilon$. Clearly both inequalities are satisfied for any $\epsilon > 0$, thus we achieve an improvement for any isogeny degree smaller than p^3. For isogenies of degree larger than p^3, the approach outlined in Section 3 using generalized KLPT from [21] will be better.

8.2 Quantum algorithms

When taking quantum resources into account, the comparison to state-of-the-art algorithms is simpler as the smoothness of the isogeny degree d does not impact the performance of the current best methods, and neither of our newly proposed algorithms. The best approach amongst the ones provided in Sections 5 to 7 is the Cornacchia algorithm. Again, we assume that the isogeny has degree less

than p^3 as otherwise the generalized KLPT algorithm from SQISign [21] can be used to compute the sought-after isogeny. For all other degrees between $p^{1/2}$ and p^3 our Cornacchia approach is faster as $1/4 < 1/4+\epsilon/2$ and $\epsilon/2 < 1/4+\epsilon/2$. For isogenies of degree less than p, the method utilising bivariate Coppersmith yields the same complexity but does not require the same heuristic as our method from Section 5.

9 Conclusion

In this article, we provided new and improved algorithms for finding a degree-d isogeny between supersingular elliptic curves E_1 and E_2. Our approach computes the endomorphism rings of E_1 and E_2, the reduced norm form of the connecting ideal and then tries to represent d. We presented three different approaches for finding a representation of d, each with its own advantages and disadvantages. The first two approaches were based on guessing two variables, then solving the remaining bivariate equation. First, we solve the Diophantine equation with a variant of Cornacchia's algorithm; second, we use a bivariate generalization of Coron's version of Coppersmith's algorithm. The advantage of Cornacchia is that it provides the fastest quantum algorithm for isogeny finding, as well as the best classical algorithm for isogenies of non-smooth degree. However, it requires a small heuristic as, in exceptional cases, Cornacchia's algorithm can be very costly. The approach using Coron's algorithm matches the complexity of the Cornacchia approach but only works for a smaller range of parameters. On the other hand, the bivariate Coron approach does not rely on the Cornacchia heuristics. Our final strategy is to guess only one variable and use either a trivariate version of Coron's algorithm or a version of Coppersmith's algorithm by Bauer and Joux for solving the remaining equation in three variables. This method alone does not provide significant improvements but, together with guessing part of the secret isogeny, it provides the best classical complexity for smooth-degree isogenies. Note that partial guessing of the isogeny is not possible when we are looking for a map of non-smooth degree.

9.1 Open problems

All our methods require us to guess at least one of the four variables of the norm equation. We leave the interesting case of studying algorithms where we do not guess any variables for further research. We expect this approach to give rise to a polynomial-time reduction between finding degree-d isogenies for isogeny degrees larger than $\sqrt{p}$ (the case which is already covered in [29]). The difficulty of this approach is that in four variables certain algebraic dependencies arising in Coppersmith's methods seem to prevent further improvement. Another promising application is whether our methods can be utilized in a constructive setting, e.g. in any application of an effective Deuring correspondence.

Acknowledgements Péter Kutas is supported by the Hungarian Ministry of Innovation and Technology NRDI Office within the framework of the Quantum Information National Laboratory Program, the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and by the UNKP-23-5 New National Excellence Program. Péter Kutas and Christophe Petit are partly supported by EPSRC through grant number EP/V011324/1.

Bibliography

[1] Arpin, S., Clements, J., Dartois, P., Eriksen, J.K., Kutas, P., Wesolowski, B.: Finding orientations of supersingular elliptic curves and quaternion orders. Cryptology ePrint Archive, Paper 2023/1268 (2023), https://eprint.iacr.org/2023/1268

[2] Basso, A., Chen, M., Fouotsa, T.B., Kutas, P., Laval, A., Marco, L., Saah, G.T.: Exploring SIDH-based signature parameters. personal communication (2023)

[3] Basso, A., Maino, L., Pope, G.: Festa: Fast encryption from supersingular torsion attacks. Cryptology ePrint Archive, Paper 2023/660 (2023), https://eprint.iacr.org/2023/660

[4] Batut, C., Belabas, K., Bernardi, D., Cohen, H., Olivier, M.: User's Guide to PARI-GP. Université de Bordeaux I (2000)

[5] Bauer, A., Joux, A.: Toward a rigorous variation of Coppersmith's algorithm on three variables. In: Advances in Cryptology — EUROCRYPT 2007, Lecture Notes in Computer Science, vol. 4515, pp. 361–378. Springer Berlin Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_21

[6] Bauer, A., Vergnaud, D., Zapalowicz, J.C.: Inferring sequences produced by nonlinear pseudorandom number generators using Coppersmith's methods. In: Public Key Cryptography — PKC 2012, Lecture Notes in Computer Science, vol. 7293, pp. 609–626. Springer Berlin Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_36

[7] Biasse, J.F., Jao, D., Sankar, A.: A quantum algorithm for computing isogenies between supersingular elliptic curves. In: International Conference on Cryptology in India. pp. 428–442. Springer (2014)

[8] Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $n^{0.292}$. In: Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18. pp. 1–11. Springer (1999)

[9] Boneh, D., Durfee, G., Howgrave-Graham, N.: Factoring $n = p^r q$ for large r. In: Crypto. vol. 1666, pp. 326–337. Springer (1999)

[10] Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. J. Symbolic Comput. **24**(3–4), 235–265 (1997). https://doi.org/10.1006/jsco.1996.0125, computational algebra and number theory (London, 1993)

[11] Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 423–447. Springer (2023)

[12] Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: Advances in Cryptology - ASIACRYPT 2018. pp. 395–427 (2018), https://doi.org/10.1007/978-3-030-03332-3_15
[13] Cohen, H.: A course in computational algebraic number theory, vol. 138. Springer Science & Business Media (2013)
[14] Coppersmith, D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Advances in Cryptology—EUROCRYPT'96: International Conference on the Theory and Application of Cryptographic Techniques Saragossa, Spain, May 12–16, 1996 Proceedings 15. pp. 178–189. Springer (1996)
[15] Coppersmith, D.: Finding a small root of a univariate modular equation. In: Advances in Cryptology—EUROCRYPT'96: International Conference on the Theory and Application of Cryptographic Techniques Saragossa, Spain, May 12–16, 1996 Proceedings 15. pp. 155–165. Springer (1996)
[16] Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology **10**(4), 233–260 (1997)
[17] Coron, J.S.: Finding small roots of bivariate integer polynomial equations revisited. In: Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings 23. pp. 492–505. Springer (2004)
[18] Corte-Real Santos, M., Costello, C., Shi, J.: Accelerating the Delfs–Galbraith algorithm with fast subfield root detection. In: Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part III. pp. 285–314. Springer (2022)
[19] Costello, C., Longa, P., Naehrig, M., Renes, J., Virdia, F.: Improved classical cryptanalysis of SIKE in practice. In: Public Key Cryptography (2020)
[20] Costello, C.: B-SIDH: Supersingular Isogeny Diffie–Hellman using twisted torsion. In: Advances in Cryptology - ASIACRYPT 2020, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II. pp. 440–463 (2020)
[21] De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact post-quantum signatures from quaternions and isogenies. In: Advances in Cryptology — ASIACRYPT 2020, Lecture Notes in Computer Science, vol. 12491, pp. 64–93. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-64837-4_3
[22] Delfs, C., Galbraith, S.D.: Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. Designs, Codes and Cryptography **78**(2), 425–440 (2016)
[23] Deuring, M.: Die Typen der Multiplikatorenringe elliptischer Funktionenkörper: G. Herglotz zum 60. Geburtstag gewidmet. In: Abhandlungen aus dem mathematischen Seminar der Universität Hamburg. vol. 14, pp. 197–272. Springer (1941)
[24] Eisenträger, K., Hallgren, S., Lauter, K.E., Morrison, T., Petit, C.: Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In: Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International

Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III. pp. 329–368 (2018)
[25] Eisenträger, K., Hallgren, S., Leonardi, C., Morrison, T., Park, J.: Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. Open Book Series **4**(1), 215–232 (2020)
[26] Fouotsa, T.B., Kutas, P., Merz, S.P., Ti, Y.B.: On the isogeny problem with torsion point information. Cryptology ePrint Archive, Paper 2021/153 (2021), https://eprint.iacr.org/2021/153
[27] Fouotsa, T.B., Kutas, P., Merz, S.P., Ti, Y.B.: On the isogeny problem with torsion point information. In: IACR International Conference on Public-Key Cryptography. pp. 142–161. Springer (2022)
[28] Fouotsa, T.B., Moriya, T., Petit, C.: M-SIDH and MD-SIDH: Countering SIDH attacks by masking information. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 282–309. Springer (2023)
[29] Galbraith, S.D., Petit, C., Shani, B., Ti, Y.B.: On the security of supersingular isogeny cryptosystems. In: Advances in Cryptology - ASIACRYPT 2016. pp. 63–91 (2016). https://doi.org/10.1007/978-3-662-53887-6_3, https://doi.org/10.1007/978-3-662-53887-6_3
[30] Girault, M., Toffin, P., Vallée, B.: Computation of approximate l-th roots modulo n and application to cryptography. In: Advances in Cryptology—CRYPTO'88: Proceedings 8. pp. 100–117. Springer (1990)
[31] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996. pp. 212–219 (1996)
[32] Hardy, G.H., Wright, E.M., et al.: An introduction to the theory of numbers. Oxford university press (1979)
[33] Hastad, J.: On using RSA with low exponent in a public key network. In: Advances in Cryptology—CRYPTO'85 Proceedings 5. pp. 403–408. Springer (1986)
[34] Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Crytography and Coding: 6th IMA International Conference Cirencester, UK, December 17–19, 1997 Proceedings 6. pp. 131–142. Springer (1997)
[35] Inc., W.R.: Mathematica, Version 11, https://www.wolfram.com/mathematica, champaign, IL, 2023
[36] Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: International Workshop on Post-Quantum Cryptography. pp. 19–34. Springer (2011)
[37] Jaques, S., Schanck, J.M.: Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. In: Annual International Cryptology Conference. pp. 32–61. Springer (2019)
[38] Jutla, C.S.: On finding small solutions of modular multivariate polynomial equations. In: Advances in Cryptology — EUROCRYPT 1998, Lec-

ture Notes in Computer Science, vol. 1403, pp. 158–170. Springer Berlin Heidelberg (1998). https://doi.org/10.1007/bfb0054124
[39] Kirschmer, M., Voight, J.: Algorithmic enumeration of ideal classes for quaternion orders. SIAM Journal on Computing **39**(5), 1714–1747 (2010)
[40] Kohel, D., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
[41] Kohel, D.R.: Endomorphism rings of elliptic curves over finite fields. Ph.D. thesis, University of California, Berkeley (1996)
[42] Kutas, P., Merz, S.P., Petit, C., Weitkämper, C.: One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In: Advances in Cryptology–EUROCRYPT 2021, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I. pp. 242–271. Springer (2021)
[43] Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen **261**, 515–534 (1982)
[44] Leroux, A.: A new isogeny representation and applications to cryptography. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 3–35. Springer (2022)
[45] Maino, L., Martindale, C.: An attack on SIDH with arbitrary starting curve. Cryptology ePrint Archive, Paper 2022/1026 (2022), https://eprint.iacr.org/2022/1026
[46] Nakagawa, K., Onuki, H.: QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. Cryptology ePrint Archive (2023)
[47] Nitaj, A.: L'algorithme de Cornacchia. In: Exposition. Math. vol. 13.4, pp. 358–365 (1995)
[48] Page, A., Wesolowski, B.: The supersingular endomorphism ring and one endomorphism problems are equivalent. arXiv preprint arXiv:2309.10432 (2023)
[49] Petit, C.: Faster algorithms for isogeny problems using torsion point images. In: Advances in Cryptology–ASIACRYPT 2017, Hong Kong, China, December 3-7, 2017, Proceedings, Part II 23. pp. 330–353. Springer (2017)
[50] Petit, C., Lauter, K.: Hard and easy problems for supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/962 (2017), https://eprint.iacr.org/2017/962
[51] Quehen, V.d., Kutas, P., Leonardi, C., Martindale, C., Panny, L., Petit, C., Stange, K.E.: Improved torsion-point attacks on SIDH variants. In: Annual International Cryptology Conference. pp. 432–470. Springer (2021)
[52] Riesel, H., Oesterlé, J., Weinstein, A.: Prime numbers and computer methods for factorization, vol. 126. Springer (1994)
[53] Robert, D.: Breaking SIDH in polynomial time. Cryptology ePrint Archive, Paper 2022/1038 (2022), https://eprint.iacr.org/2022/1038
[54] Robert, D.: Evaluating isogenies in polylogarithmic time. Cryptology ePrint Archive, Paper 2022/1068 (2022), https://eprint.iacr.org/2022/1068
[55] de Saint Guilhem, C.D., Kutas, P., Petit, C., Silva, J.: SÉTA: Supersingular encryption from torsion attacks. Cryptology ePrint Archive, Paper 2019/1291 (2019), https://eprint.iacr.org/2019/1291

[56] Sawilla, R.E., Silvester, A.K., Williams, H.C.: A new look at an old equation. In: Algorithmic Number Theory: 8th International Symposium, ANTS-VIII Banff, Canada, May 17-22, 2008 Proceedings 8. pp. 37–59. Springer (2008)
[57] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
[58] Silverman, J.H.: The arithmetic of elliptic curves, vol. 106. Springer Science & Business Media (2009)
[59] Simon, D.: Quadratic equations in dimensions 4, 5 and more. Preprint (2005)
[60] Tani, S.: Claw finding algorithms using quantum walk. Theoretical Computer Science **410**(50), 5285–5297 (2009)
[61] The Sage Developers: SageMath, the Sage Mathematics Software System (Version 10.0) (2023), `https://www.sagemath.org`
[62] Vignéras, M.F.: Algèbres De Quaternions Sur Un Corps. Springer (1980)
[63] Voight, J.: Quaternion algebras. preprint **13**, 23–24 (2018)
[64] Wesolowski, B.: The supersingular isogeny path and endomorphism ring problems are equivalent. Cryptology ePrint Archive, Report 2021/919 (2021), `https://ia.cr/2021/919`
[65] Wesolowski, B.: Orientations and the supersingular endomorphism ring problem. In: Advances in Cryptology–EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30–June 3, 2022, Proceedings, Part III. pp. 345–371. Springer (2022)

A The order embedding problem

Besides the application of computing fixed-degree isogenies and solving Problem 4.1, the improved algorithms we provide for solving multivariate equations, and most prominently the trivariate Coppersmith approach, can further be utilised to a different algorithmic problem. In particular, we can examine how our methods from Sections 5 and 6 impact the search for an element of prescribed trace and norm inside a maximal order $\mathcal{O}$, i.e. the following problem.

Problem A.1. Let $\mathcal{O}$ be a maximal order in $B_{p,\infty}$ for some prime p and let $\mathfrak{D}$ be a quadratic order. Decide whether $\mathfrak{D}$ embeds into $\mathcal{O}$ and find this embedding if it exists.

An improvement to solving Problem A.1 will have several interesting implications. For one, this so-called *order embedding problem* is naturally connected to the problem of finding connecting ideals of a given norm. For example, it is easy to see that finding a connecting ideal to the endomorphism ring of the curve $E_{1728} : y^2 = x^3 + x$ of norm d is closely related to finding an embedding of the quadratic order $\mathbb{Z}[d\iota]$ where ι is the order-four automorphism on E_{1728}. Furthermore, we will gain more insight into Wesolowski's reductions in [65]. More

precisely, the order embedding problem is the missing link in relating the *Uber isogeny problem* [55, Prob. 5.1] and the *endomorphism ring problem* [65, Prob. 6].

We provide a heuristic method to embed orders of small discriminant in polynomial time. We implemented the method and according to our experimental results, the approach works for quadratic orders of discriminant up to $p^{0.8}$.

Recall that the *Uber isogeny problem* is informally the following. Let $\mathfrak{D}$ be a quadratic order. One is then given two $\mathfrak{D}$-oriented curves and one has to find a connecting ideal class between them. It was introduced by De Feo et al. since the key recovery problem in many isogeny-based schemes can be reduced to this problem [55]. A particular example is the key recovery in Commutative Supersingular Isogeny Diffie–Hellman (CSIDH) [12] and its relation to the general isogeny problem: If the discriminant of the quadratic order is large enough, we expect $\mathfrak{D}$ to be embedded in every maximal order. Hence, finding the desired ideal class would solve the pure isogeny problem of finding any isogeny between the two given curves.

For simplicity, we will assume that the element we are looking for has trace zero, i.e. we would like to embed $\mathbb{Z}[\sqrt{-d}]$ into $\mathcal{O}$. First one can compute the $\mathbb{Z}$-lattice of trace zero elements which is known to be a rank-3 lattice of determinant p^2. If $d < p^{2/3}$, usually one can find this element by computing the shortest element in the lattice. The interesting case is when d is substantially bigger than $p^{2/3}$. Since we are working with a rank-3 lattice, the trivariate approaches described in Section 6.2 can be applied. For efficient computations, we are only interested in polynomial-time algorithms and will hence refrain from investigating the complexity of first guessing one variable and then applying one of our bivariate approaches; deducing a running time should nevertheless be straightforward. The results presented below are heuristic but more rigorous bounds could potentially be achieved using different variations of Coppersmith's algorithm.

Experimental results For our experiments, we generated problem instances in the following way. First, we computed a random maximal order in $B_{p,\infty}$. This can be accomplished by starting from a standard maximal order and taking a random walk of length $\log p$. Then we computed a basis for the trace 0 part of the order and computes a reduced basis of this lattice. From this basis we generated the corresponding quadratic form. Then we chose random x_1, x_2, x_3 of bounded size and checked whether the Coron or Bauer–Joux algorithms could recompute the solution.

An alternative way could be to fix some order $\mathbb{Z}[\sqrt{-d}] =: \mathcal{O}_0$ and find a maximal order containing it. This can be accomplished by embedding $\mathcal{O}_0$ into the quaternion algebra $B_{p,\infty}$ via finding rational solutions (x, y, z) to the equation $x^2 + py^2 + pz^2 = d$. Given d in factored form, we can use the algorithm from [59] which is conveniently implemented in PARI/GP [4] to solve the equation over $\mathbb{Q}$. It remains to compute a maximal order containing this element. Thus, we have constructed a maximal order which we know is oriented by $\mathfrak{D}$. However, this approach seemed to return solutions with a particular structure, so for our

experiment we decided to use the first approach. The main reason is that if just choose to put it in the order and find a maximal order containing that order requires factoring and is impractical for experiments. An alternative method is the way keys are generated in Séta [55] but then one of the x_i was 0.

We experimented with 3 different primes of varying sizes and we ran multiple instances for each order size. In Table 3 we present our findings on when and how often we succeeded in finding the embeddings.

Size (D)	# **succ.**	**Size** (D)	# **succ.**	**Size** (D)	# **succ.**
2^{186}	100	2^{317}	100	2^{485}	100
2^{187}	100	2^{318}	99	2^{487}	100
2^{188}	100	2^{319}	43	2^{489}	95
2^{189}	58	2^{320}	11	2^{491}	23
2^{190}	7	2^{321}	4	2^{493}	4
2^{191}	1	2^{322}	1	2^{495}	3
2^{192}	0	2^{323}	0	2^{497}	0

Table 3. Experiments for a 256-bit, 434-bit, and 610-bit prime p, respectively, 100 instances run for each discriminant size.

Based on our experiments, we conjecture that the approach outlined in this appendix works for discriminants of size $p^{0.8}$.