

Efficient algorithms for the detection of (N, N) -splittings and endomorphisms

Maria Corte-Real Santos¹[0000–0003–2651–8951]*, Craig Costello²[0000–0001–5423–7714], and Sam Frengley³[0000–0002–8904–6253]†

¹ ENS de Lyon, CNRS, UMPA, UMR 5669, Lyon, France
`maria.corte_real_santos@ens-lyon.fr`

² Queensland University of Technology
`craig.costello@qut.edu.au`

³ University of Bristol, Bristol, UK
`sam.frengley@bristol.ac.uk`

Abstract. We develop an efficient algorithm to detect whether a superspecial genus 2 Jacobian is optimally (N, N) -split for each integer $N \leq 11$. Incorporating this algorithm into the best-known attack against the superspecial isogeny problem in dimension 2 (due to Costello and Smith) gives rise to significant cryptanalytic improvements. Our implementation shows that when the underlying prime p is 100 bits, the attack is sped up by a factor of 25; when the underlying prime is 200 bits, the attack is sped up by a factor of 42; and, when the underlying prime is 1000 bits, the attack is sped up by a factor of 160. Furthermore, we describe a more general algorithm to find endomorphisms of superspecial genus 2 Jacobians.

Keywords: Isogeny-based cryptography, genus 2, superspecial, cryptanalysis.

This article is an extended version of [14] in which we additionally present an algorithm to detect superspecial Jacobians that have real multiplication by a maximal order in a real quadratic field using similar techniques.

1 Introduction

Let C and C' be genus 2 curves with superspecial Jacobians. The general dimension 2 *superspecial isogeny problem* asks us to find an isogeny

$$\phi: J_C \rightarrow J_{C'},$$

*Parts of this work were supported by the UK EPSRC grant EP/S022503/1, and the European Research Council under grant No. 101116169 (AGATHA CRYPTY).

†Funded by the Woolf Fisher and Cambridge Trusts and by Céline Maistret’s Royal Society Dorothy Hodgkin Fellowship.

of principally polarised (p.p.) abelian surfaces, where J_C and $J_{C'}$ are the Jacobians of C and C' respectively.

We say that the Jacobian J_C of a genus 2 curve C is *split* (over K) if there exists a separable (polarised) K -isogeny of p.p. abelian surfaces $J_C \rightarrow E_1 \times E_2$ where E_1/K and E_2/K are elliptic curves.

The best known algorithm for solving the superspecial isogeny problem is due to Costello and Smith [16]. It consists of two stages. The first stage computes pseudorandom walks away from the two input Jacobians to find paths to products of two supersingular elliptic curves, i.e., $\varphi: J_C \rightarrow E_1 \times E_2$ and $\varphi': J_{C'} \rightarrow E'_1 \times E'_2$. Assuming the pseudorandom walks quickly converge to the uniform distribution, the first stage runs in $\tilde{O}(p)$ classical bit operations, since the proportion of superspecial abelian surfaces that are isomorphic to a product of elliptic curves is $O(1/p)$. The second stage calls the $\tilde{O}(p^{1/2})$ Delfs-Galbraith algorithm [19] to find paths between E_1 and E'_1 and between E_2 and E'_2 . These are then glued together to obtain the path $\pi: E_1 \times E_2 \rightarrow E'_1 \times E'_2$ connecting φ and φ' in order to output the full solution $\phi := \widehat{\varphi}' \circ \pi \circ \varphi$. It follows that the entire algorithm runs in $\tilde{O}(p)$ classical bit operations on average, with the cost dominated by the first step: finding paths to products of elliptic curves.

Isogeny-based cryptography in dimension 2. The product-finding algorithm [16] that we accelerate in this work solves the general superspecial isogeny problem, which underlies the security of various isogeny-based protocols in dimension 2. An example of such a scheme is the dimension 2 analogue of the Charles-Goren-Lauter hash function [12], which was proposed by Takashima [66] and later extended by Castryck, Decru and Smith [10].

The 2022 breaks of SIDH and SIKE [9,51,57] revealed that understanding higher dimensional isogenies is essential to navigate the isogeny graphs in dimension 1. More recently, there has been a line of works leveraging the techniques used in the attacks to propose new cryptosystems that exploit isogeny computations in higher dimensions [18,1,13]. Although the hard problems underlying these schemes are not directly impacted by the algorithm that is optimised in this paper, we believe the trend towards instantiating schemes in higher dimensions will only make the dimension 2 supersingular isogeny problem more relevant to practitioners as the field of isogeny-based cryptography continues to mature.

Based on the present knowledge of attacks in dimension 2, we believe it is reasonable to speculate that the complexity of the product-finding algorithm may eventually be used as an upper-bound on the classical hardness of attacking many schemes that are currently conceivable, even when the underlying instances of the isogeny problem are special cases of its general formulation (provided no superior algorithm for the special problem is found, of course). For example, consider the dimension 2 analogue of the Sigma protocol that proves knowledge of an isogeny of a specified degree (see [24] for the latest on this protocol). In dimension 1, the best known classical attack on this protocol is the van Oorschot–Wiener (vOW) meet-in-the-middle algorithm [69]. In dimension 2, however, the $\tilde{O}(p)$ product-finding algorithm will solve the general problem at least as fast

as the van Oorschot–Wiener meet-in-the-middle algorithm [69], and is likely to become the preferred algorithm⁴ for large enough p .

Computing endomorphisms. Finding endomorphisms of supersingular elliptic curves is closely linked to solving the supersingular isogeny problem. Indeed, Wesolowski showed that the problem of finding an isogeny between a pair of supersingular elliptic curves E/\mathbb{F}_{p^2} and E'/\mathbb{F}_{p^2} is equivalent (under reductions of polynomial expected time, assuming the generalised Riemann hypothesis) to the problem of computing their endomorphism rings [71]. Recent work by Wesolowski and Page further showed that access to an oracle that returns a single non-trivial endomorphism $\phi \in \text{End}(E)$ corresponding to a given supersingular curve E is sufficient to solve the supersingular isogeny problem [55].

It is therefore natural to ask whether an analogous equivalence occurs beyond the case of elliptic curves and, in particular, in the case of dimension 2. As it currently stands, however, the literature on algorithms for computing the endomorphism ring of superspecial p.p. abelian surfaces is limited. To our knowledge, there is not yet a well-established algorithm for this computation (let alone a mature complexity analysis). In the ordinary case several such algorithms have been studied [30], [37, §6.1], [65].

Nevertheless, we expect that a connection between the hardness of the endomorphism ring and isogeny problems in dimension 2 is likely to exist. In particular, developing an efficient algorithm for computing the endomorphism ring of a superspecial p.p. abelian surface is an important open problem for furnishing the security picture of higher-dimensional isogeny-based cryptography. The new routine we present in Section 7 can be viewed as a small step in this direction, since its termination gives *partial* information about the corresponding endomorphism ring – it returns a real multiplication (RM) subfield of the endomorphism algebra. More strongly, we note that a p.p. abelian surface A having RM by the “degenerate” quadratic algebra \mathcal{O}_{N^2} of discriminant N^2 is the same as A being (N, N) -split (see [46, §2]). The Costello–Smith algorithm shows that this is the bottleneck step in solving the isogeny problem in dimension 2, and it is therefore natural to ask:

Question 1. Can the detection of an RM action by an order \mathcal{O}_D of (small) discriminant $D > 0$ be realised as the bottleneck towards solving the isogeny problem in dimension 2?

Note that for sufficiently large discriminants D (with respect to p) such an action is guaranteed to exist [36, Theorem 2.4] (this is conditional, see [36, Remark 2.6]). For further discussion on the role of real multiplication in the isogeny

⁴For a fixed memory bound w and single processor, taking $n = p^{3/4}$ [29, §4.1] in [69, Equation 4] gives an asymptotic runtime of $O(p^{9/8})$ on a single core. Moreover, parallel processors running vOW must read from, and write to, the huge central storage database (which hampers parallel performance in practice), while product-finding is memory-free and parallelises *perfectly*.

graph of superspecial p.p. abelian surfaces, the reader is encouraged to consult [11] and [35, Section 6.2].

We note that the runtime complexity of the algorithm `SplEndid` described in Remark 10 is unknown and that the analysis we present in this paper is entirely empirical. Our intention in Section 7 is not only to show that our original algorithm in [14] can be extended to more general endomorphisms, but also to use this generalised algorithm to produce numerical data (see Section 8.2) that prompts a number of interesting but unresolved questions in dimension 2.

Contributions. We begin with an implementation of the algorithm described above for finding paths to products of elliptic curves. This includes a streamlined version of the Takashima–Yoshida algorithm [67, §5.5] for computing chains of Richelot isogenies. With this optimised algorithm, we provide a toolbox for exploring the $(2, 2)$ -isogeny graph. The expansion properties of the $(2, 2)$ -isogeny graph are not well understood, and our implementation is well suited to exploring this. For example, one could hope to provide evidence towards [27, Conj. 4.10]. Understanding the expansion properties of this graph is crucial to gaining a deeper insight into the hardness of the general isogeny problem in dimension 2.

This lays the foundation for the main contribution of this work: a new algorithm that speeds up the search for paths to products of elliptic curves. At the heart of our algorithm is the work of Kumar [46], who gives explicit parametrisations of the moduli space of genus 2 curves whose Jacobians are split by an (N, N) -isogeny. When we step to a new node in the Richelot isogeny graph, these parametrisations allow us to efficiently test whether *any* of the (N, N) -isogenous neighbours are isomorphic to a product of supersingular elliptic curves without computing any expensive (N, N) -isogenies. For example, over a field whose characteristic is a 100-bit prime, an optimised Richelot isogeny (see Section 3) requires 1176 \mathbb{F}_p -multiplications. This is the cost of taking a single step in the Richelot isogeny graph, which reveals only one neighbour and is thus the per-node cost of running the attack described in [16]. However, using the new algorithm we describe in Section 5 with $N = 3$, we are able to test whether any of the $(3, 3)$ -isogenous neighbours are split with a total of 767 \mathbb{F}_p -multiplications. Since there are 40 such neighbours, the per-node cost of simultaneously searching these neighbours is less than 20 \mathbb{F}_p -multiplications each. The upshot is that when attacking an instance of the superspecial isogeny problem, we can sift through a larger proportion of superspecial Jacobians per unit time, thus reaching an elliptic curve product with fewer \mathbb{F}_p -multiplications.

Motivated by the close relationship between the supersingular isogeny problem and the problem of finding endomorphisms in the case of elliptic curves, in Section 7 we briefly turn to the problem of finding endomorphisms on superspecial Jacobians of genus 2 curves. We present an algorithm that finds certain types of endomorphisms of small degree in the superspecial isogeny graph. More precisely, leveraging the work by Elkies and Kumar [23], the methods developed

for splitting detection can be repurposed to develop an algorithm that detects whether the input Jacobian has real multiplication by a quadratic order.

In Section 8 we present the results of experiments conducted on our accelerated product-finding algorithm, as well as our endomorphism detection algorithm. For the product-finding algorithm, we report on a number of experiments conducted over both small primes (where instances of the superspecial isogeny problem can be solved) and large primes of cryptographic size. Applying our accelerated algorithm to find paths to elliptic products when $p = 2^{31} - 1$, we solve 10 instances of the problem using an average of $2^{33.0}$ multiplications in \mathbb{F}_p for an average wall time of $2^{16.3}$ seconds. Our optimised version of the original algorithm from [16] requires an average of $2^{36.8}$ multiplications in \mathbb{F}_p for an average runtime of $2^{20.5}$ seconds to solve the same 10 instances. In Table 1 we give a snapshot of the improvements that were observed in our implementation for a number of large primes of varying bitlength. We see that the relative speedup improves as the prime p grows in size (see Section 8 for more details).

p (bits)	Walks in $\Gamma_2(2; p)$ without additional searching [16] (optimised in Section 3)	Walks in $\Gamma_2(2; p)$ w. split searching in $\Gamma_2(N; p)$ This work	\mathbb{F}_p -mults. per node	improv. factor
	\mathbb{F}_p -mults. per node	set $N \in \{ \dots \}$		
50	579	{2, 3}	35	16.5x
100	1176	{2, 3}	48	24.5x
150	1575	{3, 4}	54	29.2x
\vdots	\vdots	\vdots	\vdots	\vdots
950	9772	{4, 6}	69	141.6x
1000	11346	{4, 6}	71	159.8x

Table 1. An abbreviated version of Table 6. See Section 8 for further explanation.

Indeed, for primes of at least 150 bits, we argue in Section 6.3 that (heuristically) Algorithm 4 requires an expected number of

$$\left(\frac{14 \log_2(p) + 34490}{5 \cdot 664} \right) p + O(\log_2(p))$$

\mathbb{F}_p -multiplications before encountering a product of elliptic curves. Under the same heuristics, our optimised version of the algorithm in [16] would require a larger expected $\left(\frac{12 \log_2(p) + 129}{5} \right) p + O(\log_2(p))$ \mathbb{F}_p -multiplications.

Let D be a fundamental discriminant and let \mathcal{O}_D be the maximal order in $\mathbb{Q}(\sqrt{D})$. In Section 8.2 we report on experiments in which we use our endomor-

phism detection algorithm to study the distribution of superspecial p.p. abelian surfaces in the $(2, 2)$ -isogeny graph with $\mathcal{O}_D \subset \text{End}(A)$ (or more precisely the Rosati fixed space, see Section 2) for p up to 23 bits. We compare this to a heuristic given by Florit–Smith [27] for split nodes occurring in random walks in the $(2, 2)$ -isogeny graph.

All the source code accompanying this paper is written in Magma [3] and can be found at

<https://github.com/mariascrs/SplitSearcher>.

Finally, we note that our algorithm for detecting (N, N) -splittings may be of interest outside of our target application of the dimension 2 superspecial isogeny problem. For example, it answers a question posed by Castryck and Decru [9, §11] for $N \leq 11$.

Related work. At a high level, our improvements to the dimension 2 superspecial isogeny attack can be viewed as an analogue of those recently given by Corte-Real Santos, Costello and Shi [15] to the Delfs–Galbraith attack [19] in dimension 1. Indeed, both attacks use random walks to find special nodes in the graph to reduce the (remainder of the) algorithm to a comparatively easier isogeny problem: the special nodes in the Delfs–Galbraith algorithm are the isomorphism classes of elliptic curves defined over \mathbb{F}_p , while the special nodes in the Costello–Smith algorithm are the isomorphism classes of products of elliptic curves. The key to the improvements in [15] was an efficient method for determining whether modular polynomials have subfield roots without computing any such roots explicitly. This allows many nodes to be simultaneously searched over without being visited by means of expensive isogeny computations. The key to the improvements in this paper stem from Kumar’s parametrisations of the moduli space of genus 2 curves whose Jacobians are split by an (N, N) -isogeny [46]. In a similar vein to [15], we show that these can be used to simultaneously search over many neighbours without visiting the corresponding nodes in the isogeny walks.

It is worth noting that, relatively speaking, the improvements found in this work are significantly larger than the improvements reported in [15] in the dimension 1 case. At first glance of Section 5, it seems our batch (N, N) -split searching requires a lot more computation than the analogous batch N -isogenous subfield curve searching in [15]. However, in dimension 2 we are processing $O(N^3)$ neighbours simultaneously (see Equation (1)), while the subfield search in dimension 1 is batch testing $O(N)$ neighbours each time. For primes of size 50 to 800 bits, [15, Table 6] report speedups ranging from $3.2\times$ to $17.6\times$, while the speedups we found for primes of these same sizes (see Table 6) range from $16.5\times$ to $116.3\times$.

Outline. After giving the necessary background in Section 2, we detail our optimised version of the original $F_2(2; p)$ walk from [16] in Section 3. In Section 4 we recall standard results concerning moduli spaces for genus 2 curves with split

Jacobians and Kumar’s formulae [46]. In Section 5 we present the main contribution of this work: an efficient algorithm to detect (N, N) -splittings. We give the full algorithm and discuss our implementation in Section 6. Using the methods developed in the previous two sections, in Section 7 we present our algorithm for efficient endomorphism detection. Finally, we present the experimental results in Section 8 before we conclude by mentioning some possible avenues for improving the algorithm.

2 Background

We give a brief account of abelian surfaces and fix notation. Readers looking for an in-depth discussion of higher dimensional abelian varieties and their application in isogeny based cryptography are encouraged to consult [10], [16], and [29].

Let A be an abelian surface (i.e., an abelian variety of dimension 2) defined over a field K and write \widehat{A} for the dual abelian variety. A pair (A, λ) is said to be a *polarised abelian surface* if $\lambda: A \rightarrow \widehat{A}$ is an isogeny (i.e., a surjective finite morphism of group varieties). We say that (A, λ) is *principally polarised* (p.p.) if λ is an isomorphism.

If C/K is a smooth projective curve we write J_C/K for the Jacobian of C , the abelian variety whose points parametrise degree zero divisors on C up to linear equivalence. Throughout the article we will suppress the implicit choice of (principal) polarisation on A . In particular, when $A = J_C$ is the Jacobian of a (smooth projective) curve then A is equipped with the (canonical) principal polarisation arising from the theta divisor and when $A = E_1 \times E_2$ is a product of elliptic curves then A is equipped with the product polarisation.

Let (A, λ) and (A', λ') be p.p. abelian surfaces. An isogeny $\phi: A \rightarrow A'$ is said to be an *isogeny* of p.p. abelian surfaces if there exists an integer $m \geq 1$ such that $\widehat{\phi} \circ \lambda' \circ \phi = [m]\lambda$. If $N \geq 2$ is an integer coprime to the characteristic of K , then for any abelian variety A we have the N -Weil pairing $A[N] \times \widehat{A}[N] \rightarrow \mu_N$. When A is equipped with a principal polarisation this gives rise to the N -Weil pairing

$$e_N: A[N] \times A[N] \rightarrow \mu_N.$$

We say that a subgroup $G \subseteq A[N]$ is *isotropic* (with respect to the N -Weil pairing) if $e_N(P, Q) = 1$ for all $P, Q \in G$. We say G is *maximal isotropic* if moreover there is no isotropic subgroup G' with $G \subsetneq G' \subseteq A[N]$.

Given a maximal isotropic subgroup $G \subseteq A[N]$, the abelian surface $A' = A/G$ comes equipped with a principal polarisation λ' such that $\phi: A \rightarrow A'$ is an isogeny of p.p. abelian surfaces and $\phi^*\lambda' = [m]\lambda$ for some integer m . We say that a subgroup $G \subseteq A[N]$ is an (N, N) -subgroup if it is maximal isotropic (with respect to e_N) and isomorphic (as an abstract group) to $(\mathbb{Z}/N\mathbb{Z})^2$. In this case we say that ϕ is an (N, N) -isogeny. The number of (N, N) -subgroups of

$A[N]$ is equal to

$$\mathfrak{D}_N := N^3 \prod_{\substack{\text{primes} \\ \ell|N}} \frac{1}{\ell^3} (\ell + 1)(\ell^2 + 1). \quad (1)$$

In particular, when N is prime we have $\mathfrak{D}_N = (N^2 + 1)(N + 1)$. See e.g., [8, Lemma 2] (see also [16, Lemma 2] and [29, Proposition 3(2)] when N is a prime or prime power, respectively).

Let (A, λ) be a p.p. abelian surface and let $\text{End}(A)$ denote the endomorphism ring of A . The principal polarisation λ induces an involution on $\text{End}(A)$ known as the *Rosati involution* via $\phi \mapsto \phi^\dagger = \lambda^{-1} \circ \widehat{\phi} \circ \lambda$. We write $\text{End}^\dagger(A)$ for the subring of $\text{End}(A)$ consisting of endomorphisms which are fixed by the Rosati involution.

We say that a positive integer D is a discriminant if $D \equiv 0, 1 \pmod{4}$. Let \mathcal{O}_D be the quadratic ring of discriminant D . That is, if D is not a square then $\mathcal{O}_D \subset \mathbb{Q}(\sqrt{D})$ is the order of discriminant D , and if $D = N^2$ then we have $\mathcal{O}_{N^2} \cong \{(a, b) \in \mathbb{Z}^2 : a \equiv b \pmod{N}\}$. We say that a p.p. abelian surface A has *real multiplication by \mathcal{O}_D* (or *RM by \mathcal{O}_D* for short) if there exists an embedding $\iota: \mathcal{O}_D \hookrightarrow \text{End}^\dagger(A)$. Note that if $D = N^2$ and J_C is the Jacobian of a genus 2 curve, then J_C having RM by \mathcal{O}_D is equivalent to being (N, N) -isogenous to a product of elliptic curves (see [46, §2] or Lemma 1).

2.1 Superspecial abelian surfaces

As discussed by Castryck, Decru, and Smith [10, §2], for cryptographic applications the most natural generalisation of the set of supersingular elliptic curves to dimension 2 is the set of *superspecial* p.p. abelian surfaces.

Definition 1. *We say a p.p. abelian surface $A/\overline{\mathbb{F}}_p$ is supersingular if the Newton polygon of A is a line of slope $\frac{1}{2}$. We say A is superspecial if the Hasse-Witt matrix $M \in \mathbb{F}_p^{2 \times 2}$ vanishes identically.*

If A is superspecial, then it is supersingular. The converse is not necessarily true when $\dim(A) \geq 2$. The condition for superspeciality is a natural generalisation of the fact that when $p > 3$ an elliptic curve is supersingular if and only if it has trace of Frobenius congruent to 0 modulo p . An alternative characterisation is that A is isomorphic (as an abstract abelian variety) to a product of supersingular elliptic curves.

It can be shown that every superspecial p.p. abelian surface $A/\overline{\mathbb{F}}_p$ is $\overline{\mathbb{F}}_p$ -isomorphic (as a p.p. abelian variety) to a p.p. abelian surface defined over \mathbb{F}_{p^2} (see [39, Theorem 1]) and moreover this abelian surface may be chosen to have full \mathbb{F}_{p^2} -rational 2-torsion when p is odd (see [10, §2]).

The dimension 2 superspecial isogeny problem may be stated precisely as:

Problem 1 (Dimension 2 superspecial isogeny problem). Given a pair of superspecial p.p. abelian surfaces A and A' defined over \mathbb{F}_{p^2} , find an $\overline{\mathbb{F}}_p$ -isogeny $A \rightarrow A'$.

2.2 The superspecial isogeny graph

We now describe the superspecial isogeny graph, and re-frame Problem 1 as a path finding problem.

Let $\mathcal{S}_2(p)$ denote the set of $\overline{\mathbb{F}}_p$ -isomorphism classes of superspecial p.p. abelian surfaces. Since every superspecial p.p. abelian surface admits a model over \mathbb{F}_{p^2} , the set $\mathcal{S}_2(p)$ is finite. In fact, it can be shown that it has size $O(p^3)$ [10, Theorem 1]. For each integer N coprime to p , we define $\Gamma_2(N; p)$ as the directed weighted multigraph on vertex set $\mathcal{S}_2(p)$, whose edges are $\overline{\mathbb{F}}_p$ -isomorphism classes of (N, N) -isogenies (weighted by the number of distinct kernels yielding isogenies in the class). The graph $\Gamma_2(N; p)$ is \mathfrak{D}_N -regular, where \mathfrak{D}_N is given by Equation (1) (taking into account multiplicities of each edge).

Though primitives constructed using superspecial p.p. abelian surfaces, such as the Castryck–Decru–Smith hash function [10], assume the rapid convergence of random walks in the graphs $\Gamma_2(N; p)$ to the uniform distribution, it is important to note that these expansion properties are not well understood. The superspecial isogeny graph is connected (see e.g., [54,43]), however, as discussed by Florit and Smith [27, §4.3], the graphs $\Gamma_2(N; p)$ do not fit into the definition of an expander graph as they are directed multigraphs. However, one can still obtain upper bounds on the eigenvalues of these graphs to determine whether $\Gamma_2(N; p)$ is Ramanujan, i.e. has optimal expansion. Jordan–Zaytman [43] give the first counterexample: $\Gamma_2(2; 11)$ is not Ramanujan. Florit–Smith provide evidence that the same behaviour occurs for $\Gamma_2(2; p)$ where $11 \leq p \leq 201$, therefore suggesting that the superspecial $(2, 2)$ -isogeny graph fails to be Ramanujan [27, Appendix A]. It would also be interesting to study the expansion properties of $\Gamma_2(N; p)$ for $N > 2$. Despite the lack of *optimal* expansion, Florit–Smith conjecture [27, Conjecture 4.10] that $\Gamma_2(N; p)$ still has *good enough* expansion for cryptographic purposes.

Every p.p. abelian surface is isomorphic to either the Jacobian of a curve of genus 2, or to a product of two elliptic curves with the product polarisation. In the latter case, if the abelian surface is superspecial, then the elliptic curves will be supersingular. Therefore, $\mathcal{S}_2(p)$ is equal to the disjoint union of the following two sets:

$$\begin{aligned} \mathcal{J}_2(p) &:= \{A \in \mathcal{S}_2(p) : A \cong J_C \text{ for some genus 2 curve } C\} \text{ and} \\ \mathcal{E}_2(p) &:= \{A \in \mathcal{S}_2(p) : A \cong E_1 \times E_2 \text{ for some } E_1, E_2 \in \mathcal{S}_1(p)\}, \end{aligned}$$

where the isomorphisms are of p.p. abelian varieties over $\overline{\mathbb{F}}_p$. It can be shown that $\#\mathcal{J}_2(p) = \frac{1}{2880}p^3 + O(p^2)$ and $\#\mathcal{E}_2(p) = \frac{1}{288}p^2 + O(p)$ (combine [64, Theorem V.4.1(c)] with [4, Theorem 3.10(b)] or [40, Theorem 3.3], see [10, Theorem 1] for details). In particular $\#\mathcal{E}_2(p)/\#\mathcal{S}_2(p) = 10/p + O(1/p^2)$.

Important to our work will be the ratio of nodes $A \in \mathcal{E}_2(p)$ to nodes visited while performing a random walk on $\Gamma_2(N; p)$. A natural first guess would be that this ratio matches the proportion of such nodes in the entire graph, i.e., $\sim 10/p$. However, Florit–Smith show that all but $O(p)$ of the products of elliptic curves have reduced automorphism group of order 2, and deduce that the expected proportion of products in a random walk is $\sim \frac{1}{2} \cdot \frac{10}{p} = \frac{5}{p}$ [27, §6.2].

As in the dimension 1 case, we can view the dimension 2 isogeny problem as a path finding problem in the superspecial isogeny graph.

Problem 2. Given superspecial p.p. abelian surfaces A and A' defined over \mathbb{F}_{p^2} , find a walk in $\Gamma_2(N; p)$ connecting them (when $p \nmid N$).

2.3 Attacking the general isogeny problem in dimension 2

The best known algorithm for solving Problem 2 exploits the properties of the subset $\mathcal{E}_2(p) \subseteq \mathcal{S}_2(p)$ and is depicted in Algorithm 1. Given two (\mathbb{F}_p -isomorphism classes of) p.p. abelian surfaces A and $A' \in \mathcal{J}_2(p)$, Steps 1 and 2 find paths $\varphi: A \rightarrow E_1 \times E_2$ and $\varphi': A' \rightarrow E'_1 \times E'_2$, where both $E_1 \times E_2 \in \mathcal{E}_2(p)$ and $E'_1 \times E'_2 \in \mathcal{E}_2(p)$. As $\#\mathcal{J}_2(p) = O(p^3)$ and $\#\mathcal{E}_2(p) = O(p^2)$, we expect to complete both of these steps using $\tilde{O}(p)$ operations in \mathbb{F}_p . Steps 3 and 4 then solve the dimension 1 isogeny problem on input of E_1 and E'_1 and on input of E_2 and E'_2 to output the paths $\psi_1: E_1 \rightarrow E'_1$ and $\psi_2: E_2 \rightarrow E'_2$ in the supersingular elliptic curve N -isogeny graph. Both of these steps terminate using on average $\tilde{O}(\sqrt{p})$ operations in \mathbb{F}_p [19]. If $\text{length}(\psi_1) \equiv \text{length}(\psi_2) \pmod{2}$, we can use these to construct a product path $\pi: E_1 \times E_2 \rightarrow E'_1 \times E'_2$, as described in [16, Lemma 3]. The desired path between A and A' is then $\phi := \widehat{\varphi}' \circ \pi \circ \varphi$.⁵ Overall, the cost of the algorithm is $\tilde{O}(p)$ bit operations.

For the rest of this paper we focus on improving the concrete complexity of Steps 1 and 2 of this attack, i.e., on finding paths to the product surfaces, since this is the bottleneck step that determines the concrete complexity of Algorithm 1.

Algorithm 1 Computing isogeny paths in $\Gamma_2(N; p)$ [16]

Input: A and A' in $\mathcal{S}_2(p)$

Output: A path $\phi: A \rightarrow A'$ in $\Gamma_2(N; p)$

- 1: Find a path φ from A to some $E_1 \times E_2$ in $\mathcal{E}_2(p)$
 - 2: Find a path φ' from A' to some $E'_1 \times E'_2$ in $\mathcal{E}_2(p)$
 - 3: Find a path $\psi_1: E_1 \rightarrow E'_1$ using (elliptic curve) path finding
 - 4: Find a path $\psi_2: E_2 \rightarrow E'_2$ using (elliptic curve) path finding
 - 5: **if** $\text{length}(\psi_1) \not\equiv \text{length}(\psi_2) \pmod{2}$ **then**
 - 6: **return** \perp
 - 7: **else**
 - 8: Construct a path $\pi: E_1 \times E_2 \rightarrow E'_1 \times E'_2$ using ψ_1, ψ_2 as in [16, Lemma 3]
 - 9: **return** the path $\phi := \widehat{\varphi}' \circ \pi \circ \varphi$ from A to A'
-

⁵If $\text{length}(\psi_1) \not\equiv \text{length}(\psi_2) \pmod{2}$, we fail and return \perp . Note, however, only three runs of Algorithm 1 are required to successfully return path ϕ . Indeed, if we instead run Algorithm 1 to find paths $\psi_1: E_1 \rightarrow E'_1$, $\psi_{2,1}: E_2 \rightarrow E$, and $\psi_{2,2}: E \rightarrow E'_2$, where $E: y^2 = x^3 + x$ has an endomorphism of degree 2, say τ , then we can set $\psi_2 = \psi_{2,2} \circ \psi_{2,1}$ if $\text{length}(\psi_1) \equiv \text{length}(\psi_{2,1} \circ \psi_{2,2}) \pmod{2}$ and $\psi_2 = \psi_{2,2} \circ \tau \circ \psi_{2,1}$, otherwise.

Applications to cryptanalysis. In the security analysis of their hash function [10], Castryck–Decru–Smith correctly argue that, since the steps taken by their hash function correspond entirely to “good extensions” (see Section 3.2), the path returned by [16, Algorithm 1] (which does not only consist of good extensions) is therefore not a valid preimage [10, Footnote 11]. However, more recent work by Florit and Smith [27, §6.2 - 6.4] shows that collisions in the Castryck–Decru–Smith hash function can be constructed once a walk to an elliptic product is known. So long as we assume our walks approximate the random distribution on $\Gamma_2(2; p)$ (more on this in Remark 1), then we consider it prudent to use the complexity of the product-finding algorithms to classify the security of a given instance of the CDS hash function, even if preimage resistance is the governing security property.

As will become apparent in Section 6, our acceleration of the Costello–Smith algorithm will return a $(2^n N, 2^n N)$ -isogeny (for some n). However, for many cryptographic protocols in isogeny-based cryptography, the secret isogeny will be of a specified degree, usually a prime power ℓ^k . Though an algorithm that transforms a $(2^n N, 2^n N)$ -isogeny to a (ℓ^k, ℓ^k) -isogeny has yet to be developed, for example by generalising the KLPT algorithm [44] to dimension 2, we find it prudent to conjecture such an algorithm exists, rather than betting the security of primitives on the converse.

3 Optimised product finding in $\Gamma_2(2; p)$

In this section we describe an optimised instantiation of the product finding algorithm from [16] in the case of dimension 2.

Our instantiation uses pseudo-random walks in the superspecial subgraph of the Richelot isogeny graph [28, Definition 1] and exploits a streamlined version of Takashima and Yoshida’s Richelot isogeny algorithm [67] to take efficient steps therein.

3.1 Taking a step in $\Gamma_2(2; p)$

We start by deriving a streamlined version of Takashima and Yoshida’s Richelot isogeny algorithm [67, Algorithm 2] that will be used as the basis for pseudo-random walks in the superspecial subgraph of $\Gamma_2(2; p)$. On input of the six-tuple $\mathbf{a} = (a_0, \dots, a_5) \in (\mathbb{F}_{p^2})^6$ defining⁶ the genus 2 curve

$$C/\mathbb{F}_{p^2} : y^2 = (x - a_0) \cdots (x - a_5),$$

the algorithm outputs the six-tuple $\mathbf{a}' = (a'_0, \dots, a'_5) \in (\mathbb{F}_{p^2})^6$ that defines

$$C'/\mathbb{F}_{p^2} : y^2 = (x - a'_0) \cdots (x - a'_5),$$

⁶For odd p , superspecial abelian surfaces always have full \mathbb{F}_{p^2} -rational 2-torsion (cf. [10, §2]), which in particular implies that the a_i are defined over \mathbb{F}_{p^2} .

where $\phi: J_C \rightarrow J_{C'}$ is the Richelot isogeny whose non-trivial kernel is precisely the three points $((x - a_i)(x - a_{i+1}), 0)$ in J_C with $i \in \{0, 2, 4\}$.

The main modifications we have made to their algorithm are:

- We assume that both the equations for C and C' are indeed given by the sextic polynomials whose six roots are rational elements of \mathbb{F}_{p^2} . This avoids the case distinctions made by Takashima and Yoshida that allow for quintic inputs and outputs (i.e., one of the a_i and/or a'_j being at infinity), which are unnecessary for our purposes (they occur with negligible probability, and after a change of coordinates we may assume that C and C' are defined by sextics).
- We do not keep track of the leading coefficient of the sextic, since this merely determines which quadratic twist we are on, which is irrelevant for our application because twists correspond to the same node in $I_2(2; p)$. This means we avoid the final inversion in Line 33 of [67, Algorithm 2].
- Each of the three iterations of their main loop involve separate inversion and square root computations. In each case we merge the inversion and square root into one combined inverse-and-square-root computation (see Line 7 of Algorithm 2) using the trick described in [58].

On top of a small, fixed, number of field multiplications, Algorithm 2 computes a Richelot isogeny using 3 calls to `InvSqrt`, which is essentially the same cost as a square root in \mathbb{F}_{p^2} (i.e., 2 exponentiations in \mathbb{F}_p). This means our streamlined version saves all of the four additional \mathbb{F}_{p^2} inversions reported by Takashima and Yoshida [67, §5.5]. Otherwise, the notation and description of the algorithm is essentially unchanged: the indices in Line 3 of Algorithm 2 are taken modulo 6, and the indices in Line 5 are taken modulo 3.

Algorithm 2 `Rlsog()`: A Richelot isogeny in the general case

Input: $\mathbf{a} = (a_0, \dots, a_5) \in (\mathbb{F}_{p^2})^6$ defining $C/\mathbb{F}_{p^2}: y^2 = (x - a_0) \cdots (x - a_5)$.

Output: $\mathbf{a}' = (a'_0, \dots, a'_5) \in (\mathbb{F}_{p^2})^6$ defining $C'/\mathbb{F}_{p^2}: y^2 = (x - a'_0) \cdots (x - a'_5)$, where $\phi: J_C \rightarrow J_{C'}$ is a Richelot isogeny whose kernel corresponds to the three quadratic splittings $(x - a_i)(x - a_{i+1})$ for $i = 0, 2, 4$; and `split`, a boolean that is true if the image of ϕ is in $\mathcal{E}_2(p)$.

```

1: Initialise  $\lambda \leftarrow [\mathbf{a}[0] \cdot \mathbf{a}[1], \mathbf{a}[2] \cdot \mathbf{a}[3], \mathbf{a}[4] \cdot \mathbf{a}[5]]$ ,  $\theta \leftarrow []$ ,  $\mathbf{a}' \leftarrow []$ 
2: for  $j = 0$  to 2 do
3:    $\rho \leftarrow [\mathbf{a}[2j+2] - \mathbf{a}[2j+4], \mathbf{a}[2j+3] - \mathbf{a}[2j+5], \mathbf{a}[2j+2] - \mathbf{a}[2j+5], \mathbf{a}[2j+3] - \mathbf{a}[2j+4]]$ 
4:    $\theta[j] \leftarrow \rho[0] + \rho[1]$ 
5:    $\nu \leftarrow \lambda[j+1] - \lambda[j+2]$ 
6:    $\delta \leftarrow \rho[0] \cdot \rho[1] \cdot \rho[2] \cdot \rho[3]$ 
7:    $(\mu, \kappa) \leftarrow \text{InvSqrt}(\theta_j, \delta)$ 
8:    $(\mathbf{a}'[2j], \mathbf{a}'[2j+1]) \leftarrow ((\nu + \kappa) \cdot \mu, (\nu - \kappa) \cdot \mu)$ 
9: split  $\leftarrow (\lambda[0] \cdot \theta[0] + \lambda[1] \cdot \theta[1] + \lambda[2] \cdot \theta[2]) = 0$ 
10: return  $(\mathbf{a}', \text{split})$ 

```

Alternatives for computing $(2^n, 2^n)$ -isogenies. There are numerous ways to compute chains of $(2, 2)$ -isogenies that would be fit for our purposes, but we are yet to find one that can appreciably outperform repeated calls to Algorithm 2. Recall that each such call computes a $(2, 2)$ -isogeny using a fixed number of \mathbb{F}_p -multiplications on top of three calls to the merged inversion-and-square root computation (i.e., `InvSqrt`). Castryck and Decru’s multiradical variant of a Richelot isogeny also requires at least three square root computations in \mathbb{F}_{p^2} [8, §4.2], so the most we could expect to gain using their formulae is in the constant number of additional \mathbb{F}_p -operations (assuming any field inversions required in their case can also be absorbed into the square root calls). Kunzweiler’s efficient $(2^n, 2^n)$ -isogeny algorithm [47] could also be used in our scenario, but in testing this algorithm against ours we observed that, on average, ours performs between 3x and 5x faster for the two primes considered by Kunzweiler. Note, Kunzweiler’s formulae were derived with a different target application (i.e., G2SIDH) in mind, meaning computing a chain of $(2, 2)$ -isogenies of fixed length n is most efficient when $2^n \mid p+1$. In our algorithm, we compute chains of length much larger than any such n and, as a result, this comparison is unfair to [47]. Our comparison is to ensure that we are not sacrificing efficiency in our context.

3.2 Walking in the superspecial subgraph of $\Gamma_2(2; p)$

We now turn to describing walks in the superspecial subgraph of $\Gamma_2(2; p)$ that take steps using the `Rlsog` algorithm developed above. To ensure that these walks are non-backtracking and avoid short cycles, the output of `Rlsog` must first be permuted so that the quadratic splitting implicit to its ordering (see §3.1) corresponds to a *good extension* of the previous $(2, 2)$ -isogeny (i.e., a $(2, 2)$ -isogeny whose kernel intersects trivially with the kernel of the dual of the previous $(2, 2)$ -isogeny).

Kernel permutations corresponding to good extensions. Following Castryck, Decru and Smith [10], there are 8 non-equivalent permutations of our a_i which correspond to good extensions of the previous $(2, 2)$ -isogeny. Our walks are deterministically defined by pseudorandom bitstrings. Each step uses three bits to choose which of the 8 good extensions defines our next $(2, 2)$ -isogeny. Using [10, Proposition 3], we define the function $\mathbf{a} \leftarrow \text{PermuteKernel}(\mathbf{a}, \text{bits})$ by

$$\mathbf{a} \leftarrow \begin{cases} (\mathbf{a}[0], \mathbf{a}[2], \mathbf{a}[1], \mathbf{a}[4], \mathbf{a}[3], \mathbf{a}[5]), & \text{bits} = 0|0|0; \\ (\mathbf{a}[0], \mathbf{a}[2], \mathbf{a}[1], \mathbf{a}[5], \mathbf{a}[3], \mathbf{a}[4]), & \text{bits} = 0|0|1; \\ (\mathbf{a}[0], \mathbf{a}[3], \mathbf{a}[1], \mathbf{a}[4], \mathbf{a}[2], \mathbf{a}[5]), & \text{bits} = 0|1|0; \\ (\mathbf{a}[0], \mathbf{a}[3], \mathbf{a}[1], \mathbf{a}[5], \mathbf{a}[2], \mathbf{a}[4]), & \text{bits} = 0|1|1; \\ (\mathbf{a}[0], \mathbf{a}[4], \mathbf{a}[1], \mathbf{a}[2], \mathbf{a}[3], \mathbf{a}[5]), & \text{bits} = 1|0|0; \\ (\mathbf{a}[0], \mathbf{a}[4], \mathbf{a}[1], \mathbf{a}[3], \mathbf{a}[2], \mathbf{a}[5]), & \text{bits} = 1|0|1; \\ (\mathbf{a}[0], \mathbf{a}[5], \mathbf{a}[1], \mathbf{a}[3], \mathbf{a}[2], \mathbf{a}[4]), & \text{bits} = 1|1|0; \\ (\mathbf{a}[0], \mathbf{a}[5], \mathbf{a}[1], \mathbf{a}[2], \mathbf{a}[3], \mathbf{a}[4]), & \text{bits} = 1|1|1. \end{cases}$$

Remark 1. Under a mild conjecture on the associated eigenvalues, Florit and Smith [27] show that despite Richelot isogeny graphs not having optimal expansion, walks of length $O(\log p)$ still approximate the stationary distribution on $\Gamma_2(2; p)$ [27, Theorem 6.1]. This statement is implicitly assuming that walks are unrestricted, i.e., that each step can take any one of the 15 outgoing Richelot isogenies. In choosing to restrict each step in $\Gamma_2(2; p)$ to the 8 good edges with the aim of avoiding fruitless cycles, we are under the implicit assumption that these walks also rapidly approximate the stationary distribution. All of our experiments over small primes produced results that support this assumption (see Section 8), and Florit and Smith also comment in its favour [27, §6.4]. Nevertheless, if future research provides evidence to the contrary, modifying our walks to include the 6 other extensions is straightforward. In this case we could either aim to prohibit certain sequences of isogenies that cycle back to prior nodes, or (since we abandon walks after a small number of steps – see below) simply tolerate the possibility of revisiting prior nodes. Even if a walk did cycle back and hit a prior node, in general we would have a 14^{-n} chance of continuing along the same path for n steps thereafter.

Pseudorandom walks in the superspecial subgraph of $\Gamma_2(2; p)$. A given step of our pseudorandom walk can now be defined as $\mathbf{a} \leftarrow \text{Step}(\mathbf{a}, \text{bits})$, where the function `Step` is simply given by

$$\text{Step}(\mathbf{a}, \text{bits}) = \text{Rlsog}(\text{PermuteKernel}(\mathbf{a}, \text{bits})).$$

Recall (from Lines 1 and 2 of Algorithm 1) that our goal is to find a path φ from $A \in \mathcal{S}_2(p)$ to some $E_1 \times E_2 \in \mathcal{E}_2(p)$. In principle, one could continue walking deterministically from the input node $A \in \mathcal{S}_2(p)$ for as long as it takes to find the splitting $E_1 \times E_2 \in \mathcal{E}_2(p)$, but the length of this path would be $O(p)$. To ensure a compact description of the solution, we instead take a relatively small number of steps from $A \in \mathcal{S}_2(p)$ before abandoning a walk, returning back to $A \in \mathcal{S}_2(p)$, and starting again.

Our implementation uses `Magma`'s inbuilt function `SHA1`: $\{0, 1\}^* \rightarrow \{0, 1\}^{160}$ to generate pseudorandom walks consisting of 160 Richelot isogeny steps as follows. We start by setting $H_0 := \text{StartingSeed}(\mathbf{a})$, where $\mathbf{a} \in (\mathbb{F}_{p^2})^6$ defines the input node $A \in \mathcal{S}_2(p)$, and where `StartingSeed` merely concatenates and parses the 12 \mathbb{F}_p components of \mathbf{a} in order to be fed as input into `SHA1`. We then define the function `Hash`: $\{0, 1\}^* \rightarrow \{0, 1\}^{480}$ as `Hash`: $s \mapsto \text{SHA1}(s) \parallel \text{SHA1}^2(s) \parallel \text{SHA1}^3(s)$, where $\text{SHA1}^2(s)$ denotes $\text{SHA1}(\text{SHA1}(s))$, etc. Our first walk in $\Gamma_2(2; p)$ is defined by $H_1 = \text{Hash}(H_0)$; these 480 bits are used (three bits at a time) to give 160 steps away from $A \in \mathcal{S}_2(p)$, at which point we return back to $A \in \mathcal{S}_2(p)$ and repeat the process by using $H_{i+1} = \text{Hash}(H_i)$ for $i = 1, 2, \dots$, until one of our calls to `Rlsog` returns `split = true`, at which point our walks have hit a node in $\mathcal{E}_2(p)$. To proceed to the elliptic curve path finding in Steps 3 and 4 of Algorithm 1, the j -invariants of the elliptic curves in the product of the final $(2, 2)$ -isogeny are determined using [10, §6.2]. This concludes the description

of our implementation of the generic product finding algorithm from [16] that works entirely in $\Gamma_2(2;p)$.

Choice of Optimisations. In our search for product curves we use optimised walks in $\Gamma_2(2;p)$, rather than adopting Castryck and Decru’s multiradical isogenies [8] to walk in $\Gamma_2(3;p)$. Indeed, their hash function built from multiradical $(3,3)$ -isogenies between superspecial genus 2 Jacobians outperforms its $(2,2)$ -counterpart by a factor ≈ 9 . We first note that the bulk of the Castryck–Decru speedup comes from their hash function processing 3 trits of entropy per $(3,3)$ -isogeny, rather than 3 bits of entropy processed by a $(2,2)$ -isogeny. In our application, however, entropy is irrelevant and we are only interested in the raw cost of taking one step in the graph. Nevertheless, Castryck and Decru still report a $\approx 2.7\times$ speedup for a multiradical $(3,3)$ -isogeny (which is dominated by 3 cube roots over \mathbb{F}_{p^2}) compared to a multiradical $(2,2)$ -isogeny (which is dominated by 3 square roots over \mathbb{F}_{p^2}), with this factor coming directly from the relative performance of cube roots and square roots in \mathbb{F}_{p^2} in `Magma`. In our implementation, we optimised explicit computation of the square roots in \mathbb{F}_{p^2} in terms of \mathbb{F}_p exponentiations and multiplications using the tricks in [58, §5.3], and we are unaware of analogous (or any) tricks in the cube root case that could outperform the square root computation.

Furthermore, we use walks in $\Gamma_2(2;p)$ that do not store or recycle any information from previous steps. Indeed, we could not see an obvious way to (re)use any of the three square roots in Line 7 of Algorithm 2 to compute the other 7 good extensions. We remark that this is a feature of our choice to walk using only good extensions, and we could in fact recycle these square roots to compute some of the *bad* extensions. If it turns out that there *is* a way to compute all 8 of the image tuples \mathbf{a} in appreciably fewer operations than calling the `RIsoG` algorithm on all 8 kernels individually, then one could define an octonary tree in an analogous fashion to the binary tree from [15].

4 Humbert surfaces

We give a brief review of some well known facts about genus 2 curves with split Jacobians, p.p. abelian surfaces with real multiplication, and their moduli. The reader wishing for a more in-depth discussion is encouraged to consult e.g., [6, §2], [23], [33], [45], [46], [68].

Let K be a field of characteristic not equal to 2. Let \mathcal{M}_2 denote the variety whose points $[C] \in \mathcal{M}_2(\bar{K})$ correspond to the \bar{K} -isomorphism classes of genus 2 curves C/\bar{K} .

4.1 The Igusa–Clebsch invariants of a genus 2 curve

From the invariant theory of the binary sextic, we may associate to any genus 2 curve C/K its Igusa–Clebsch invariants $I_2(C)$, $I_4(C)$, $I_6(C)$, and $I_{10}(C)$ (here the subscript denotes the weight of the invariant). Moreover the isomorphism

class of C/\overline{K} is uniquely determined by its Igusa–Clebsch invariants (see e.g., [41,52]). This induces a K -birational morphism $\mathcal{M}_2 \hookrightarrow \mathbb{P}(2, 4, 6, 10)$ given by associating to $[C]$ the Igusa–Clebsch invariants $[I_2(C) : I_4(C) : I_6(C) : I_{10}(C)]$.

Explicitly, if C/K is a genus 2 curve given by a Weierstrass equation

$$C: y^2 = (x - a_0) \cdots (x - a_5)$$

where $a_0, \dots, a_5 \in K$, we define:

$$I_2(C) := \sum_{15} (01)^2 (23)^2 (45)^2, \quad I_4(C) := \sum_{10} (01)^2 (12)^2 (20)^2 (34)^2 (45)^2 (53)^2,$$

$$I_6(C) := \sum_{60} (01)^2 (12)^2 (20)^2 (34)^2 (45)^2 (53)^2 (03)^2 (14)^2 (25)^2, \quad \text{and}$$

$$I_{10}(C) := \prod_{i < j} (a_i - a_j)^2,$$

where, for any permutation $\sigma \in S_6$, we let (ij) denote the difference $(a_{\sigma(i)} - a_{\sigma(j)})$. Here the sums are taken over all distinct expressions in the a_i as σ ranges over S_6 ; the subscripts denote the number of expressions in each sum.

4.2 Optimal splittings of Jacobians of a genus 2 curves

Let C be a curve of genus 2 defined over a field K . Recall that we say the Jacobian J_C of C is *split* (over K) if there exists a (polarised) separable K -isogeny $\phi: J_C \rightarrow E_1 \times E_2$ where E_1/K and E_2/K are elliptic curves⁷.

To work explicitly with subvarieties of \mathcal{M}_2 which parametrise genus 2 curves with split Jacobians, we will restrict our focus to Jacobians which are split by an (N, N) -isogeny. However, without imposing further conditions on the isogeny, our subvarieties will not be irreducible. Following Bruin–Doerksen [6, §2], we make the following definition:

Definition 2. *Let K be a field, C/K be a curve of genus 2, and E/K be an elliptic curve. We say that a cover $\psi: C \rightarrow E$ of degree N is optimal if N is coprime to the characteristic of K and ψ does not factor through a non-trivial unramified covering.*

We say that a polarised separable isogeny $\phi: J_C \rightarrow E_1 \times E_2$ is an optimal (polarised) (N, N) -splitting if ϕ is an (N, N) -isogeny and the covering $C \rightarrow E_1$ induced by ϕ and the Abel–Jacobi map is optimal. In this case J_C is said to be optimally (N, N) -split.

In our application N will be an integer ≤ 11 and K will be the finite field \mathbb{F}_{p^2} for some prime number $p \gg 11$, so the assumption that ϕ is separable will be automatically satisfied.

In fact every splitting factors through an optimal (N, N) -splitting, more precisely:

⁷The convention that ϕ is separable contrasts with, e.g., [6, Definition 2.1].

Proposition 1. *If J_C is split (over K) then there exists an integer $N \geq 2$ such that J_C is optimally (N, N) -split (over K).*

Proof. We closely follow [6, Proposition 2.8]. Since J_C is split, there exists a separable K -isogeny $\phi: J_C \rightarrow E_1 \times E_2$ where E_1/K and E_2/K are elliptic curves. Since ϕ is separable, there exists an elliptic curve E'_1/K such that the morphism $C \rightarrow E_1$ induced by the Abel–Jacobi map and ϕ factors through an optimal cover $\psi: C \rightarrow E'_1$. By [6, Lemma 2.6], ψ gives rise to an optimal (N, N) -splitting $J_C \rightarrow E'_1 \times E'_2$, where E'_2 is an elliptic curve and N is the degree of ψ . \square

4.3 The surfaces $Y_-(D)$ and \mathcal{H}_D

We now recall several definitions about Hilbert modular and Humbert surfaces. The reader concerned only with detecting splittings can safely skip to Lemma 1.

We first recall Theorem 1 which asserts the existence of a surface $Y_-(D)$ which parametrises genus 2 Jacobians with RM by \mathcal{O}_D , or more precisely genus 2 Jacobians equipped with a real multiplication $\iota: \mathcal{O}_D \hookrightarrow \text{End}^\dagger(A)$. For more details, the reader should consult [20,56] or [38, Theorem 4.9] when D is not a square, and [46, §2] when D is a square. We call the surface $Y_-(D)$ the *Hilbert modular surface of discriminant D* .

Theorem 1. *Let $D > 1$ be a discriminant (i.e., D is an integer congruent to 0 or 1 modulo 4) which is coprime to the characteristic of K . Then there exists a surface $Y_-(D)$, defined over K , whose \bar{K} -points are in bijective correspondence with (isomorphism classes of) pairs (J_C, ι) where C is a genus 2 curve and $\iota: \mathcal{O}_D \hookrightarrow \text{End}^\dagger(A)$.*

Given a point $(J_C, \iota) \in Y_-(D)$ we may forget ι and obtain a morphism $Y_-(D) \rightarrow \mathcal{M}_2$ given by $(J_C, \iota) \mapsto [C]$. We write \mathcal{H}_D for the image of this morphism, which we call the *Humbert surface of discriminant D* . The Humbert surface remembers only that the endomorphism ring of the Jacobian J_C contains \mathcal{O}_D (but forgets the choice of embedding).

Lemma 1. *Let C/K be a genus 2 curve and let $N > 1$ be coprime to the characteristic of K . There exists an optimal (N, N) -splitting $J_C \rightarrow E_1 \times E_2$ if and only if $[C] \in \mathcal{H}_{N^2}(\bar{K}) \subset \mathcal{M}_2(\bar{K})$. In particular, the Jacobian of C is split over \bar{K} if and only if there exists an integer $N \geq 2$ such that $[C] \in \mathcal{H}_{N^2}(\bar{K})$.*

Proof. The first claim follows from the discussion in [46, §2]. The second claim follows from Proposition 1. \square

Let $D > 1$ be an integer which satisfies either:

- (D1) $D = N^2$ where $2 \leq N \leq 11$, or
- (D2) $D < 100$ is a fundamental discriminant.

Under both conditions **(D1)** and **(D2)** explicit models for the surfaces \mathcal{H}_D are known. Elkies–Kumar [23] gave explicit models for the surface $Y_-(D)$ for every fundamental discriminant $D < 100$. These calculations were extended by Kumar [46] who computed explicit models of the surfaces $Y_-(N^2)$ for each integer $N \leq 11$. In each of these cases the Humbert surfaces \mathcal{H}_D are rational (i.e., birational to \mathbb{A}^2), and they presented the surfaces $Y_-(D)$ as a double covers of \mathcal{H}_D together with the moduli interpretation of \mathcal{H}_D . More specifically, they compute rational functions $\mathcal{I}_2^{(D)}(r, s)$, $\mathcal{I}_4^{(D)}(r, s)$, $\mathcal{I}_6^{(D)}(r, s)$, $\mathcal{I}_{10}^{(D)}(r, s)$ which (after an appropriate projective rescaling) may be taken to lie in $\mathbb{Z}[r, s]$ and for which the following diagram commutes

$$\begin{array}{ccc} \mathbb{A}^2 & \overset{\varphi_D}{\dashrightarrow} & \mathbb{P}(2, 4, 6, 10) \\ \downarrow & & \uparrow \\ \mathcal{H}_D & \longrightarrow & \mathcal{M}_2 \end{array}$$

Here the maps on the left and right are birational and the rational map φ_D is given by $(r, s) \mapsto [\mathcal{I}_2^{(D)}(r, s) : \mathcal{I}_4^{(D)}(r, s) : \mathcal{I}_6^{(D)}(r, s) : \mathcal{I}_{10}^{(D)}(r, s)]$.

Remark 2. Genus 2 curves C/K with split Jacobians have appeared many times elsewhere in the literature. Indeed when $N \leq 4$ generic families were known classically from work of Legendre, Jacobi, Hermite, Grousat, Burkhardt, Brioschi, and Bolza (as discussed in [46]). More recently, Shaska [59] gave a method for general N for computing the surface $Y_-(N^2)$ together with a curve $C/K(Y_-(N^2))$ such that J_C is (N, N) -isogenous to a product $E_1 \times E_2$. Explicit computations have been performed for $2 \leq N \leq 5$ by Shaska, Magaard, Volklein, Wijesiri, Wolf, and Woodland [60,62,61,50,63] and by Gaudry–Schost [34], Bröker–Lauter–Howe–Stevenhagen [5], Bruin–Doerksen [6], and Djukanović [22,21]. If J_C is (N, N) -isogenous over K to a product of elliptic curves $E_1 \times E_2$ there exists an anti-symplectic Galois equivariant isomorphism $E_1[N] \cong E_2[N]$ (see e.g., [6, Proposition 2.8]). For $N > 11$ this has been considered by Fisher when $N = 13, 17$ [25,26] and Frengley when $N = 12, 14, 15$ [31,32]. However, while they give the generic elliptic curves E_1 and E_2 , they do not give the genus 2 curve C .

4.4 The image of the morphism $\mathcal{H}_{N^2} \rightarrow \mathcal{M}_2$

Recall that we have a map $\mathcal{H}_{N^2} \rightarrow \mathcal{M}_2 \rightarrow \mathbb{P}(2, 4, 6, 10)$ given by the Igusa–Clebsch invariants. The (Zariski closure of) the image of this map is a projective surface given by the vanishing of a polynomial $F_N \in \mathbb{Z}[I_2, I_4, I_6, I_{10}]$ which is homogeneous with respect to the weights.

If K is a field of characteristic coprime to $2N$, the Jacobian of a genus 2 curve C/K is optimally (N, N) -split over \bar{K} if and only if

$$F_N(I_2(C), I_4(C), I_6(C), I_{10}(C)) = 0.$$

For $2 \leq N \leq 5$ the polynomial F_N was computed by Bruin–Doerksen [6,7, Theorem 1.2] and Shaska, Magaard, Volklein, Wijesiri, Wolf, and Woodland [61,63,50].

Such equations may be computed from Kumar’s formulae [46]. For each $N \leq 5$ we interpolate the image of φ_{N^2} modulo a small number of primes of approximately 128 bits. Lifting these equations to characteristic zero with the LLL algorithm gives a candidate for F_N .

Since F_N is an irreducible polynomial and the image of φ_{N^2} is an irreducible variety, we verify the result in `Magma` by checking that F_N vanishes at the equations defining φ_{N^2} . These polynomials are available in the code accompanying this article, and their properties are summarised in Table 2.

Remark 3. As pointed out to us by Benjamin Smith, for a generic genus 2 curve $C : y^2 = (x - a_0) \dots (x - a_5)$ the polynomial $F_2(I_2(C), I_4(C), I_6(C), I_{10}(C))$ is (up to a scaling factor) equal to the square of the product of the determinants of the 15 Richelot kernels. This gives a connection to the classical work of Bolza [2, p. 51] where this is the invariant which Bolza calls R^2 .

N	Weighted degree of F_N	Number of monomials in F_N	Average bitlength of the coefficients of F_N
2	30	34	~ 16.6
3	80	318	~ 64.3
4	180	2699	~ 197
5	480	43410	~ 617

Table 2. The number of monomials in the defining equation F_N for the image of \mathcal{H}_{N^2} in $\mathbb{P}(2, 4, 6, 10)$ and the total number of bytes required to (naively) store the coefficients of each F_N .

5 Efficient detection of (N, N) -splittings

In this section we present an algorithm to efficiently detect whether, at each step, the p.p. abelian surface J_C is (N, N) -isogenous (over $\overline{\mathbb{F}}_p$) to a product of elliptic curves, without ever computing an (N, N) -isogeny. In this way we are able to use resultants and gcd computations, rather than inefficient computations of (N, N) -isogenies, therefore avoiding all N^{th} -root calculations and the need to work in extension fields when the N -torsion is not fully \mathbb{F}_{p^2} -rational.

A natural starting point to perform this detection is to exploit the equations F_N for the image of the morphism $\mathcal{H}_{N^2} \rightarrow \mathbb{P}(2, 4, 6, 10)$ (see Section 4.4). Indeed, if a genus 2 curve C/\mathbb{F}_{p^2} is (N, N) -split, then $F_N(I_2(C), I_4(C), I_6(C), I_{10}(C)) = 0$. However, as demonstrated in Table 2, both the number of monomials in F_N and the bitlength of its coefficients grow rapidly with N . As a result, computing and storing F_N for $N > 5$ is challenging. Instead, we will use techniques in

elimination theory to determine whether $[C]$ lies on the (Zariski closure of) the image of φ_{N^2} . Indeed, even for $N \leq 5$, evaluating the image at the Igusa–Clebsch invariants of C will not outperform this method.

Lemma 2. *Let $D > 1$ be a discriminant satisfying either (D1) or (D2) and let C/K be a genus 2 curve defined over a field K of characteristic not dividing $2D$. Suppose that the Igusa–Clebsch invariants $I_2(C)$, $I_4(C)$, $I_6(C)$, and $I_{10}(C)$ are non-zero. Write $\alpha_1(C) := \frac{I_4(C)}{I_2(C)^2}$, $\alpha_2(C) := \frac{I_2(C)I_4(C)}{I_6(C)}$, and $\alpha_3(C) := \frac{I_4(C)I_6(C)}{I_{10}(C)}$. If there exist $r_0 \in \bar{K} \cup \{\infty\}$ and $s_0 \in \bar{K}$ satisfying*

$$\begin{cases} \alpha_1(C) = \frac{\mathcal{I}_4^{(D)}(r_0, s_0)}{\mathcal{I}_2^{(D)}(r_0, s_0)^2}, \\ \alpha_2(C) = \frac{\mathcal{I}_2^{(D)}(r_0, s_0)\mathcal{I}_4^{(D)}(r_0, s_0)}{\mathcal{I}_6^{(D)}(r_0, s_0)}, \\ \alpha_3(C) = \frac{\mathcal{I}_4^{(D)}(r_0, s_0)\mathcal{I}_6^{(D)}(r_0, s_0)}{\mathcal{I}_{10}^{(D)}(r_0, s_0)} \end{cases}$$

then $[C] \in \mathcal{H}_D(\bar{K})$. In particular, if $D = N^2$ then J_C is optimally (N, N) -split over \bar{K} . Here $\mathcal{I}_w^{(D)}(r, s)$ are as in Section 4.3.

Proof. The rational map $\psi: \mathbb{P}(2, 4, 6, 10) \dashrightarrow \mathbb{A}^3$ given by $[I_2 : I_4 : I_6 : I_{10}] \mapsto \left(\frac{I_4}{I_2^2}, \frac{I_2 I_4}{I_6}, \frac{I_4 I_6}{I_{10}}\right)$ is birational with inverse $(\alpha_1, \alpha_2, \alpha_3) \mapsto \left[1 : \alpha_1 : \frac{\alpha_1}{\alpha_2} : \frac{\alpha_1^2}{\alpha_2 \alpha_3}\right]$. Moreover on the open subvariety of $\mathbb{P}(2, 4, 6, 10)$ where I_2, I_4, I_6 , and I_{10} are nonzero the map ψ restricts to an isomorphism onto its image. The claim follows from the discussion preceding the lemma. \square

Remark 4. It is common in the literature (e.g., [6,42]) to choose the affine patch with coordinates the *absolute invariants* $\frac{6(I_2^2 - 16I_4)}{I_2^3}$, $\frac{-12(5I_2^3 - 176I_2I_4 + 384I_6)}{I_2^3}$, and $\frac{3888I_{10}}{I_2^2}$. Our choice is *ad hoc* and made to optimise the algorithms in Section 5.2. In particular, the choice in Lemma 2 yields polynomials $P_{i,j}$ in Lemma 3 of smaller degree. Choosing an affine patch of $\mathbb{P}(2, 4, 6, 10)$ so that the analogous polynomials to $P_{i,j}$ in Lemma 3 have minimal degree would likely lead to improved performance of our algorithm.

Remark 5. In the code accompanying this article we provide a function `InvariantsFromWeierstrassPoints` that, on input of the 6-tuple $\mathbf{a} = (a_0, \dots, a_5) \in (\mathbb{F}_{p^2})^6$ of Weierstrass points, computes the 3-tuple $\boldsymbol{\alpha}(C) = (\alpha_1(C), \alpha_2(C), \alpha_3(C)) \in (\mathbb{F}_{p^2})^3$ using a total of 291 multiplications and one (merged) inversion in \mathbb{F}_p . This is the first step of Algorithm 4.

If $D > 1$ is a discriminant satisfying either (D1) or (D2), we define polynomials $f_k^{(D)}(r, s) \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][r, s]$ by

$$\begin{aligned} f_1^{(D)}(r, s) &= \mathcal{I}_4^{(D)}(r, s) - \alpha_1 \mathcal{I}_2^{(D)}(r, s)^2, \\ f_2^{(D)}(r, s) &= \mathcal{I}_2^{(D)}(r, s)\mathcal{I}_4^{(D)}(r, s) - \alpha_2 \mathcal{I}_6^{(D)}(r, s), \\ f_3^{(D)}(r, s) &= \mathcal{I}_4^{(D)}(r, s)\mathcal{I}_6^{(D)}(r, s) - \alpha_3 \mathcal{I}_{10}^{(D)}(r, s). \end{aligned}$$

The following proposition follows immediately from Lemma 2.

Proposition 2. *Let $D > 1$ be a discriminant satisfying either (D1) or (D2) and suppose that C/K is a genus 2 curve with non-zero Igusa–Clebsch invariants. If there exist $r_0 \in \bar{K} \cup \{\infty\}$ and $s_0 \in \bar{K}$ such that for each $w \in \{2, 4, 6, 10\}$ we have $\mathcal{I}_w^{(D)}(r_0, s_0) \neq 0$ and $f_k^{(D)}(r_0, s_0) = 0$ then $[C] \in \mathcal{H}_D(\bar{K})$. In particular if $D = N^2$ then J_C is optimally (N, N) -split over \bar{K} .*

We describe a method for determining whether, given a genus 2 curve $C/\bar{\mathbb{F}}_p$ with superspecial Jacobian, there exists a point $P \in \mathbb{A}^2(\bar{\mathbb{F}}_p)$ such that the polynomials $f_k^{(D)}(r, s)$ vanish at P when $D = N^2$ and $N \leq 11$ (Section 5.2) or a fundamental discriminant $D \leq 100$ (Section 7.1). Our approach utilises the equations of Kumar [46] and Elkies–Kumar [23]. Moreover, we determine lower bounds on the costs in terms of \mathbb{F}_p -multiplications.

5.1 The resultants of f_i and f_j

Fix a discriminant $D > 1$ satisfying either (D1) or (D2). For each distinct pair $i, j \in \{1, 2, 3\}$, define polynomials⁸

$$R_{i,j}^{(D)}(s) := \text{res}_r(f_i^{(D)}(r, s), f_j^{(D)}(r, s)) \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][s].$$

If C/K is a genus 2 curve then, since resultants are invariant under ring homomorphisms, by the elimination property of the resultant (see e.g., [17, §3.6 Lemma 1]) the specialisations $(R_{i,j}^{(D)})_{[C]}(s) \in K[s]$, given by evaluating the coefficients of $R_{i,j}^{(D)}(s)$ at $\alpha_1(C)$, $\alpha_2(C)$, and $\alpha_3(C)$, vanish at the s -coordinate of any common solution to the specialised polynomials $(f_j^{(D)})_{[C]}(r, s)$.

However, these resultants (generically) have factors which correspond to unwanted solutions (i.e., where one of the polynomials $\mathcal{I}_w^{(D)}$ vanishes). We make this more precise in the following lemma.

Lemma 3. *Let $L = \mathbb{Q}(\alpha_1, \alpha_2, \alpha_3)$ and let $D > 1$ be a discriminant satisfying (D1) or (D2). When $i \neq j$, there exist polynomials $Q_{i,j}^{(D)} \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][s]$ dividing $R_{i,j}^{(D)}$ with the following property: for each pair $r_0, s_0 \in \bar{L}$ such that $f_k^{(D)}(r_0, s_0) = 0$ for $k = 1, 2, 3$ and $Q_{i,j}^{(D)}(s_0) = 0$, then $\mathcal{I}_w^{(D)}(r_0, s_0) = 0$ for some $w \in \{2, 4, 6, 10\}$. Moreover, the polynomials*

$$P_{i,j}^{(D)} = R_{i,j}^{(D)} / Q_{i,j}^{(D)} \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][s]$$

are coprime.

Proof. This follows from a direct calculation in Magma. □

⁸If necessary, we swap the roles of Elkies–Kumar or Kumar’s r and s so that the polynomials $P_{i,j}^{(D)}$ from Lemma 3 are of lowest degree (as noted in the accompanying code). It would be interesting to find a birational transformation of \mathbb{A}^2 which minimises $\deg P_{i,j}^{(D)}$.

Applying [17, §3.6 Corollary 7] we have:

Proposition 3. *Let C/K be a genus 2 curve such that $I_w^{(D)}(C) \neq 0$ for each $w \in \{2, 4, 6, 10\}$ and let $D > 1$ be a discriminant satisfying **(D1)** or **(D2)**.*

If there exist $r_0, s_0 \in \bar{K}$ such that $(f_i^{(D)})_{[C]}(r_0, s_0) = 0$ for each $i = 1, 2, 3$ then the degree of $\gcd((P_{1,2}^{(D)})_{[C]}, (P_{2,3}^{(D)})_{[C]})$ is at least 1.

Conversely if $s_0 \in \bar{K}$ is a root of $\gcd((P_{1,2}^{(D)})_{[C]}, (P_{2,3}^{(D)})_{[C]})$ then there exist $r_0, r'_0 \in \bar{K} \cup \{\infty\}$ such that $(f_1^{(D)})_{[C]}(r_0, s_0) = (f_2^{(D)})_{[C]}(r_0, s_0) = 0$ and $(f_2^{(D)})_{[C]}(r'_0, s_0) = (f_3^{(D)})_{[C]}(r'_0, s_0) = 0$.

In the electronic data attached to this article we give the polynomials $P_{i,j}^{(D)} \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][s]$ for each pair $j \neq k$.

5.2 An algorithm to detect (N, N) -split Jacobians

We now present our algorithm to efficiently detect whether the Jacobian of a genus 2 curve C/\mathbb{F}_{p^2} is (N, N) -split for some integer $2 \leq N \leq 11$. In Proposition 4 we then give an upper bound on the number of \mathbb{F}_p -multiplications required by the algorithm.

Precomputation step. Let $D = N^2$. We reduce the coefficients of the polynomials $P_{1,2}^{(D)}, P_{2,3}^{(D)} \in \mathbb{Z}[\alpha_1, \alpha_2, \alpha_3][s]$ from Lemma 3 modulo p to obtain polynomials $\tilde{P}_{1,2}^{(D)}, \tilde{P}_{2,3}^{(D)} \in \mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3][s]$, which are stored.

Evaluation and gcd step. Our approach is summarised in Algorithm 3. To test whether a given genus 2 curve C/\mathbb{F}_{p^2} with superspecial Jacobian is (N, N) -split, we specialise the coefficients of $\tilde{P}_{1,2}^{(D)}, \tilde{P}_{2,3}^{(D)}$ at $\alpha(C) = (\alpha_1(C), \alpha_2(C), \alpha_3(C))$, by running the algorithm `EvalCoeffs`, to obtain the polynomials $(\tilde{P}_{1,2}^{(D)})_{[C]}, (\tilde{P}_{2,3}^{(D)})_{[C]} \in \mathbb{F}_{p^2}[s]$. The `EvalCoeffs` algorithm takes as input $\tilde{P}_{i,j}^{(D)}$ and the invariants $\alpha(C)$, and evaluates the coefficients of the polynomial at these invariants (see the proof of Proposition 4 for more details).

We then compute the gcd of $(\tilde{P}_{1,2}^{(D)})_{[C]}$ and $(\tilde{P}_{2,3}^{(D)})_{[C]}$ using the “inversion-free gcd” algorithm `InvFreeGCD` from [15, Algorithm 1], modified to output the gcd explicitly, rather than a boolean.

If this gcd has degree ≥ 1 then it has a root $s_0 \in \bar{\mathbb{F}}_p$ and (by Proposition 3) there exist $r_0, r'_0 \in \bar{\mathbb{F}}_p \cup \{\infty\}$ such that $(f_1^{(D)})_{[C]}(r_0, s_0) = (f_2^{(D)})_{[C]}(r_0, s_0) = 0$ and $(f_2)_{[C]}(r'_0, s_0) = (f_3^{(D)})_{[C]}(r'_0, s_0) = 0$. By Proposition 2 to verify that J_C is (N, N) -split it suffices to show that we may take $r_0 = r'_0$ such that $\mathcal{I}_w^{(D)}(r_0, s_0) \neq 0$ for each $w \in \{2, 4, 6, 10\}$. We verify the first condition by computing the gcd of $(f_1^{(D)})_{[C]}(r, s_0), (f_2^{(D)})_{[C]}(r, s_0), (f_3^{(D)})_{[C]}(r, s_0)$, and if it has degree ≥ 1 computing a root $r_0 \in \bar{\mathbb{F}}_p$. We verify the second condition by checking that $\mathcal{I}_w^{(D)}(r_0, s_0) \neq 0$ for each $w \in \{2, 4, 6, 10\}$ – we abbreviate this to the function `lgNonzero`.

Remark 6. If J_C is optimally (N, N) -split then Algorithm 3 will return **true** with high probability. In this case $[C]$ is an \mathbb{F}_{p^2} -point on \mathcal{H}_{N^2} . Since $\varphi_{N^2}: \mathbb{A}^2 \dashrightarrow \mathcal{H}_{N^2}$ is birational (over \mathbb{F}_p) it is an isomorphism outside a closed \mathbb{F}_p -subvariety $X \subseteq \mathcal{H}_{N^2}$ of dimension 1. But from the Weil conjectures $\#\mathcal{H}_{N^2}(\mathbb{F}_{p^2}) = O(p^4)$ and $\#X(\mathbb{F}_{p^2}) = O(p^2)$. In particular except in $O(1/p^2)$ of cases there exist $r_0, s_0 \in \overline{\mathbb{F}}_p$ satisfying the conditions of Proposition 3.

Algorithm 3 $\text{IsSplit}(\alpha(C), \tilde{P}_{1,2}^{(N^2)}, \tilde{P}_{2,3}^{(N^2)}, N)$:

Input: A tuple $\alpha(C) = (\alpha_1(C), \alpha_2(C), \alpha_3(C))$, the polynomials $\tilde{P}_{1,2}^{(N^2)}, \tilde{P}_{2,3}^{(N^2)} \in \mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3][r]$, and an integer $2 \leq N \leq 11$.

Output: A boolean which is true if J_C is optimally (N, N) -split.

```

1:  $(\tilde{P}_{1,2})_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{P}_{1,2}^{(N^2)}, \alpha(C))$ 
2:  $(\tilde{P}_{2,3})_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{P}_{2,3}^{(N^2)}, \alpha(C))$ 
3:  $g \leftarrow \text{InvFreeGCD}((\tilde{P}_{1,2})_{[C]}, (\tilde{P}_{2,3})_{[C]})$ 
4: if  $\deg g \geq 1$  then
5:    $s_0 \leftarrow \text{ComputeRoot}(g)$ 
6:    $(\tilde{f}_1)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_1^{(N^2)}, \alpha(C))$ 
7:    $(\tilde{f}_2)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_2^{(N^2)}, \alpha(C))$ 
8:    $(\tilde{f}_3)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_3^{(N^2)}, \alpha(C))$ 
9:    $h \leftarrow \text{InvFreeGCD}(\text{InvFreeGCD}((\tilde{f}_1)_{[C]}(r, s_0), (\tilde{f}_2)_{[C]}(r, s_0)), (\tilde{f}_3)_{[C]}(r, s_0))$ 
10:  if  $\deg h \geq 1$  then
11:     $r_0 \leftarrow \text{ComputeRoot}(h)$ 
12:    bool  $\leftarrow \text{lgNonzero}(r_0, s_0)$ 
13:    if bool == true then
14:      return true
15: return false

```

The cost of Algorithm 3. We now determine an upper bound for the number of \mathbb{F}_p -multiplications required for the online part of this method (i.e., ignoring the cost of precomputation). In the analysis that follows we assume that Karatsuba multiplication is used in \mathbb{F}_{p^2} , hence we cost one \mathbb{F}_{p^2} -multiplication as three \mathbb{F}_p -multiplications.

Proposition 4. *Let $N \in \{2, \dots, 11\}$ be an integer, and let $\text{mons}(N)$ be the set of monomials in $\alpha_1, \alpha_2, \alpha_3$ appearing in the coefficients of $\tilde{P}_{1,2}^{(N^2)}$ and $\tilde{P}_{2,3}^{(N^2)}$ (which lie in $\mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3]$). For each $i = 1, 2, 3$, let*

$$d_i(N) = \max(\{\text{degree of } \alpha_i \text{ in } m \mid m \in \text{mons}(N)\}).$$

The cost of steps 1–3 in Algorithm 3 (with input N) is at most

$$3(d_1(N) + d_2(N) + d_3(N)) + 6m(N) + 2M(N) + \frac{3}{2}(d_P(N) + 2)(d_P(N) + 3) - 27$$

\mathbb{F}_p -multiplications, where $d_P(N) = \deg \tilde{P}_{1,2}^{(N^2)} + \deg \tilde{P}_{2,3}^{(N^2)}$, $m(N) = \#\text{mons}(N)$, and $M(N)$ is the number of monomials in $\alpha_1, \alpha_2, \alpha_3$ appearing in the coefficients of $\tilde{P}_{1,2}^{(N^2)}$ and $\tilde{P}_{2,3}^{(N^2)}$ counting repetitions.

Proof. We first evaluate the coefficients of $\tilde{P}_{1,2}^{(N^2)}, \tilde{P}_{2,3}^{(N^2)} \in \mathbb{F}_{p^2}[\alpha_1, \alpha_2, \alpha_3][s]$ at the normalised invariants $\alpha_1(C), \alpha_2(C), \alpha_3(C) \in \mathbb{F}_{p^2}$ using our evaluation algorithm `EvalCoeffs` on each polynomial. This runs as follows. We first compute powers $\alpha_1(C)^2, \dots, \alpha_1(C)^{d_1(N)}$ where $d_1(N)$ is the maximum degree of α_1 appearing in $\text{mons}(N)$ (as defined in the statement of the proposition). Similarly, we compute powers of $\alpha_2(C)$ and $\alpha_3(C)$ up to $d_2(N)$ and $d_3(N)$ respectively. This step is performed using $d_1(N) + d_2(N) + d_3(N) - 3$ multiplications in \mathbb{F}_{p^2} .

From these powers, we obtain the monomials appearing in the coefficients of $(\tilde{P}_{1,2}^{(N^2)})_{[C]}(s)$ and $(\tilde{P}_{2,3}^{(N^2)})_{[C]}(s)$ in at most $2m(N)$ \mathbb{F}_{p^2} -multiplications, where $m(N) = \#\text{mons}(N)$. We then require $2M(N)$ \mathbb{F}_p -multiplications (and $2M(N)$ additions) to construct the coefficients of $(\tilde{P}_{1,2}^{(N^2)})_{[C]}$ and $(\tilde{P}_{2,3}^{(N^2)})_{[C]}$.

The final step computes the gcd of $(\tilde{P}_{1,2}^{(N^2)})_{[C]}$ and $(\tilde{P}_{2,3}^{(N^2)})_{[C]}$ using `InvFreeGCD`. This requires $\frac{3}{2}(d_P(N) + 2)(d_P(N) + 3) - 18$ \mathbb{F}_p -multiplications by [15, Proposition 2]. \square

The cost from Proposition 4 depends only on N . Therefore, for each $2 \leq N \leq 11$, we can determine the total number of \mathbb{F}_p -multiplications required for the detection per node revealed in $\mathcal{S}_2(p)$ for any prime p . We give these costs in Table 3.

Noting that, when $N \neq N'$ we may have non-empty intersection $\text{mons}(N) \cap \text{mons}(N')$, our implementation of Algorithm 4 stores all evaluated monomials to avoid repeated computations. In particular, the upper bound in Proposition 4 is often not sharp.

Remark 7. We note that, in practice, when our algorithm enters the if loop on Line 4 in Algorithm 3, we have yet to encounter a case where Steps 5–14 fail to return `true`. In these cases the bound in Proposition 4 yields a bound on the cost of Algorithm 3. It is however possible to construct examples of polynomials for which they would be necessary – e.g., $f_1(r, s) = r - 1$, $f_2(r, s) = s - r(r + 1)(r - 1)$, and $f_3(r, s) = r + 1$. It would be interesting to put this observation on rigorous footing by showing that with overwhelming probability the roots r_0 and r'_0 guaranteed by Proposition 3 are equal.

Alternative approach for $N = 10$ and 11 . When $N = 10, 11$, several megabytes are required to store the coefficients of the polynomials $\tilde{P}_{i,j}^{(N^2)}$. Rather than computing the resultants $R_{1,2}^{(N^2)}$ and $R_{2,3}^{(N^2)}$ and dividing out by the generic factors described in Lemma 3 to obtain $P_{1,2}^{(N^2)}, P_{2,3}^{(N^2)}$ as a precomputation, the approach we pursued was to instead perform these two steps during the online phase. Even still, our experiments (which were reinforced by the cost analysis above) revealed that performing the detection for $N = 10, 11$ is suboptimal in

N	$d_1(N)$	$d_2(N)$	$d_3(N)$	$m(N)$	$M(N)$	$d_P(N)$	Total $\#\mathbb{F}_p$ mults.	Total $\#\mathbb{F}_p$ mults. per node revealed
2	1	2	1	6	23	6	175	12.5
3	2	3	2	11	97	16	767	19.2
4	6	8	6	78	1136	35	4882	46.9
5	6	10	6	64	2500	92	18818	120.6
6	7	11	7	91	4118	114	29188	52.1
7	10	14	10	190	24779	294	182641	456.6
8	16	24	16	433	73454	340	325606	395.2
9	12	16	12	271	69648	540	582474	539.3
10	24	32	24	1005	260178	606	1082007	495.4
11	28	38	28	1345	669432	1120	3237198	2211.2

Table 3. Values of $d_1(N)$, $d_2(N)$, $d_3(N)$, $m(N)$, $M(N)$, and $d_P(N)$ for $N \in \{2, \dots, 11\}$. The final columns respectively list the number of \mathbb{F}_p -multiplications in Proposition 4 and the ratio of multiplications to the number of (N, N) -isogenous p.p. abelian surfaces.

our application to `SplitSearcher` (shown in Algorithm 4) and slows the overall search down, even when the characteristic of the field is very large. Thus, we leave the further optimisation of these computations as future work.

6 The full algorithm

In Section 3 we discussed our optimised implementation of the product-finding attack [16] that works entirely in the Richelot isogeny graph $\Gamma_2(2; p)$. In this section, we present `SplitSearcher`, which leverages our efficient detection of (N, N) -splittings from Section 5.2 to improve on the concrete complexity of product-finding when solving the dimension 2 isogeny problem.

6.1 `SplitSearcher`

Each time we take a step using a Richelot isogeny, we will use the methods from the previous section to detect whether the current node is (N, N) -isogenous to a product of elliptic curves, for some subset of integers in $2 \leq N \leq 11$. Using the algorithm from Section 5.2 makes this check much more efficient than, say, walking in $\Gamma_2(N; p)$; each node we step to would require computing an (N, N) -isogeny which, at minimum, requires three N -th roots in \mathbb{F}_{p^2} [8].

Each time we take a step and arrive at a new abelian surface, A , we are in one of two cases: either A is isomorphic to a product of elliptic curves, in which case the algorithm terminates, or A is isomorphic to the Jacobian of a genus 2

curve C/\mathbb{F}_{p^2} . In the latter case, `SplitSearcher` calls Algorithm 3 to detect whether A is (N, N) -split for certain $2 \leq N \leq 11$. The set of N 's for which this detection is performed is chosen to minimise the number of \mathbb{F}_p -multiplications per node revealed (either by stepping on them in $\Gamma_2(2; p)$ or inspecting them via our splitting detection) in $\mathcal{S}_2(p)$. Since it only depends on the prime p , determining this optimal list of N 's is performed during precomputation.

If Algorithm 3 determines that A is (N, N) -split, the elliptic curves E_1 and E_2 can be recovered by applying [49, Algorithm 4] to compute all (N, N) -isogenies from A (alternatively, E_1 and E_2 may be recovered from Kumar's equations [46] by solving for r_0 and s_0 in Proposition 2). As both of these costs are negligible and do not affect the cost of finding such a splitting, we may view this as a post-computation step and exclude it from our multiplication counts.

A precise formulation of the full algorithm for finding paths to elliptic curve products is given by Algorithm 4. Along with the target abelian surface $A \in \mathcal{S}_2(p)$, the auxiliary inputs into the algorithm are the polynomials $\tilde{P}_{1,2}^{(N^2)}, \tilde{P}_{2,3}^{(N^2)} \in \mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3][r]$ (see Lemma 3), and the optimal set $\mathcal{N} \subseteq \{2, \dots, 11\}$ (see Section 6.2). The hash function on Line 4 is assumed to be of the form $\text{Hash}: \{0, 1\}^* \rightarrow \{0, 1\}^{3\ell}$, where ℓ is a positive integer, since we use three bits of entropy each time we call the Richelot isogeny (i.e., `Step` algorithm) in Line 13. In practice we choose ℓ to be large enough that we can expect to find an elliptic product in walks of ℓ steps, but not *too* large, since storing walks of up to ℓ steps requires more storage on average. Once the 3ℓ bits of entropy have been consumed, the hash function is called again and the walk is restarted from $\mathbf{a}_{\text{start}}$ (more on this in Remark 8). The output returned by Algorithm 4 is of the form (path, N) , where `path` is a sequence of $3k$ bits (with $k \leq \ell$) and N is an integer: the $3k$ bits define a sequence of k Richelot isogenies and the integer N specifies the final (N, N) -isogeny whose image is in $\mathcal{E}_2(p)$.

Remark 8. In a real-world attack, we would expect to return to Line 4 of Algorithm 4 an exponential number of times before the algorithm terminates. Thus, there are a number of ways one could recycle information computed in the early stages of each walk to avoid recomputing them over and over again. One solution that is easy to implement in view of Algorithm 4 would be to store a hash table whose entries each correspond to the (hash of the) Igusa–Clebsch invariants of any node that is visited and checked for (N, N) -splittings. Upon returning to a given node and finding a collision in the hash table, the walk could simply avoid the tests for (N, N) -splittings between Lines 8 and 11. Another approach would be to build a table of the six-tuples \mathbf{a} that are computed after the first t Richelot steps have been taken, alongside the label of the $3t$ -bit string that took us there. Each time we return back to Line 4 and iterate the hash function, we simply check to see if the first $3t$ bits are already in the table and, if so, we can skip straight to \mathbf{a} .

Finally, as is mentioned in [16], parallelising the search for product curves is trivial. For P processors, we would simply compute P unique short walks from our target surface $A \in \mathcal{S}_2(p)$ and send each of the corresponding image surfaces A_1, \dots, A_P to a unique processor as its assigned input surface.

Algorithm 4 SplitSearcher: finding paths to elliptic curve products

Input: $\mathbf{a}_{\text{start}} = (a_0, \dots, a_5) \in (\mathbb{F}_{p^2})^6$ defining a genus 2 curve C/\mathbb{F}_{p^2} with superspecial Jacobian, and a set $\mathcal{N} \subseteq \{2, 3, \dots, 11\}$.

Output: A pair (path, N) where path is a path $\varphi: J_C \rightarrow J_{C'}$ in $\Gamma_2(2; p)$ and N is an integer such that $J_{C'}$ is optimally (N, N) -split.

```

1: split ← false
2: H ← StartingSeed( $\mathbf{a}_{\text{start}}$ ) §3.1
3: while not split do
4:   ( $H, i, \text{path}, \mathbf{a}$ ) ← (Hash( $H$ ), 0,  $\{\emptyset\}, \mathbf{a}_{\text{start}}$ )
5:   while  $i < \ell$  and not split do
6:     if  $\mathcal{N} \neq \emptyset$  then
7:        $\alpha(C) \leftarrow$  InvariantsFromWeierstrassPoints( $\mathbf{a}$ ) Remark 5
8:       for  $N \in \mathcal{N}$  do
9:         split ← IsSplit( $\alpha(C), \tilde{P}_{1,2}^{(N^2)}, \tilde{P}_{2,3}^{(N^2)}, N$ ) Algorithm 3
10:        if split then
11:          return (path,  $N$ )
12:        bits ←  $H[3i] \parallel H[3i+1] \parallel H[3i+2]$ 
13:         $\mathbf{a}, \text{split} \leftarrow$  Step( $\mathbf{a}, \text{bits}$ ) §3.2
14:        path ← path  $\parallel$  bits
15:         $i \leftarrow i + 1$ 
16: return (path, 2)

```

6.2 Determining the optimal set \mathcal{N}

Recall that, when we step to a new p.p. abelian surface $A \in \mathcal{S}_2(p)$, we want to determine if it is (N, N) -split for a set $\mathcal{N} \subseteq \{2, \dots, 11\}$ of N . We wish to determine the optimal subset $\mathcal{N} \subseteq \{2, \dots, 11\}$, i.e., the subset which minimises the number of \mathbb{F}_p -multiplications per node revealed in the graph. The first step towards determining this ‘multiplications-per-node’ ratio is to count the number of nodes in $\mathcal{S}_2(p)$ that are inspected inside the for loop of Algorithm 4 with a finite set of integers $\mathcal{N} \subseteq \mathbb{Z}_{\geq 2}$. A first attempt would be to simply count the number of neighbours a node $A \in \mathcal{S}_2(p)$ has in $\Gamma(N; p)$, i.e., \mathfrak{D}_N given by Equation (1) in Section 2. However, this is an overcount as we now detail.

Suppose we take a non-backtracking walk

$$A_0 \xrightarrow{\phi_0} A_1 \xrightarrow{\phi_1} \dots \xrightarrow{\phi_{n-1}} A_n \xrightarrow{\phi_n} \dots \quad (2)$$

in $\Gamma_2(2; p)$ and we inspect (N, N) -splittings for $N \in \mathcal{N}$. If $0 \leq m \leq n$ are integers, let $\phi_{m,n}$ denote the $(2^{n-m}, 2^{n-m})$ -isogeny $\phi_{m-1} \circ \dots \circ \phi_n$ and let $\widehat{\phi_{m,n}}$ denote $\widehat{\phi_{m,n}}$.

Firstly, if both N and $2^k N$ are contained in \mathcal{N} (for $k \geq 1$), then any abelian surfaces (N, N) -isogenous to A_n are automatically $(2^k N, 2^k N)$ -isogenous to A_{n+k} . Therefore, we restrict to only considering subsets \mathcal{N} which do not contain pairs of integers $M \neq N$ with $N = 2^k M$.

This restriction is not sufficient to stop double-counting nodes. Indeed, suppose $N \in \mathcal{N}$ with $N = 2M$. Then any abelian surface (N, N) -isogenous to A_n

will be (M, M) -isogenous to A_{n+1} . In particular such an abelian surface will also be (N, N) -isogenous to A_{n+2} . To rule out such scenarios, we introduce the following restriction on our paths.

Definition 3. Let $\mathcal{N} \subseteq \mathbb{Z}_{\geq 2}$ be a finite set of integers and let \mathcal{P} be a walk of $(2, 2)$ -isogenies in $\Gamma_2(2; p)$ as in (2).

Let $M, N \in \mathcal{N}$ and suppose that there exist integers $m, n \geq 0$ and (M, M) - and (N, N) -isogenies $\psi_M: A_m \rightarrow B$ and $\psi_N: A_n \rightarrow B$. We say that \mathcal{P} resists collisions for M, N if there exists an integer $i \geq 0$ and an isogeny $\Psi: A_i \rightarrow B$ such that $\psi_M = \Psi \circ \phi_{m,i}$ and $\psi_N = \Psi \circ \phi_{n,i}$.

We say that \mathcal{P} resists collisions for \mathcal{N} if it resists collisions for every pair $M, N \in \mathcal{N}$.

We are now able to state precisely the number of nodes checked between Lines 8–11 of Algorithm 4, assuming our paths resist collisions for the set \mathcal{N} .

Lemma 4. Let $\mathcal{N} \subseteq \mathbb{Z}_{\geq 2}$ be a finite set of integers such that, if \mathcal{N} is non-empty, there do not exist distinct $M, N \in \mathcal{N}$ with $N = 2^k M$ for any $k \geq 1$.

Let \mathcal{P} be a path in $\Gamma_2(2; p)$ which resists collisions for \mathcal{N} . The number of nodes inspected per step by running Algorithm 4 in \mathcal{P} is at least

$$\text{nodes}_{\mathcal{N}} := \begin{cases} \sum_{N \in \mathcal{N}} \mathfrak{D}'_N & \text{if } \mathcal{N} \text{ contains a power of } 2, \\ \sum_{N \in \mathcal{N}} \mathfrak{D}'_N + 1 & \text{otherwise} \end{cases}$$

where

$$\mathfrak{D}'_N = \mathfrak{D}_N - \sum_{\substack{1 \leq k \\ 2^k | N}} \mathfrak{D}_{N/2^k}$$

and \mathfrak{D}_N is the number of neighbours of a node in $\Gamma_2(N; p)$, given in Equation (1). Equality holds for steps taken after $\max_{N \in \mathcal{N}} (2 \log_2(N))$ steps.

Remark 9. It is important to note that the assumption that \mathcal{P} resists collisions for \mathcal{N} is mild in practice. Indeed, when \mathcal{N} contains only odd integers the assumption simplifies to requiring that, in a walk in the $(2, 2)$ -isogeny graph, any abelian surface (N, N) -isogenous to A_n is not (M, M) -isogenous to A_m for some m . The set \mathcal{N} will consist only of integers ≤ 11 and our walks have length $O(\log(p))$. A collision of this sort therefore implies that A_n has an endomorphism of degree $O(\log(p))$. Heuristically there should be very few such abelian surfaces. Indeed in the dimension 1 case, by Proposition B.3 in the unpublished appendix to [48], the proportion of supersingular elliptic curves with an endomorphism of degree at most $O(\log(p))$ is $O(\log(p)^{3/2}/p)$.

Proof. Suppose we have taken the following walk in $\Gamma_2(2; p)$

$$A_0 \rightarrow A_1 \rightarrow \cdots \rightarrow A_n \rightarrow A_{n+1} \rightarrow \cdots,$$

applying Algorithm 4.

First note that if \mathcal{N} contains a power of 2, then each successive p.p. abelian surface A_i is known not to be a product of elliptic curves. By hypothesis, there do not exist distinct $M, N \in \mathcal{N}$ with $N = 2^k M$ for any $k \geq 1$. Therefore, since \mathcal{P} resists collisions for \mathcal{N} , for each *distinct* $M, N \in \mathcal{N}$ the p.p. abelian surfaces (M, M) -isogenous to A_m are not (N, N) -isogenous to A_n for all $m, n \geq 0$. In particular it suffices to show that the number of p.p. abelian surfaces (N, N) -isogenous to A_i , but not (N, N) -isogenous to A_j for each $j < i$, is equal to \mathfrak{D}_N .

The claim follows immediately when N is odd, since the walk takes place in $\Gamma_2(2; p)$. If N is even, write $N = 2^\ell M$ where $\ell \geq 1$ and M is odd. In this case, for each $1 \leq k \leq \ell$, any p.p. abelian surface $(2^{\ell-k} M, 2^{\ell-k} M)$ -isogenous to A_{n-k} is (N, N) -isogenous to both A_{n-2k} and A_n . Therefore, $\mathfrak{D}_{N/2^k}$ surfaces (N, N) -isogenous to A_n are (N, N) -isogenous to A_{n-2k} .

The claim follows by summing over $1 \leq k \leq \ell$. Note that equality holds if $n - 2k \geq 0$ for each $1 \leq k \leq \ell$, i.e., we have taken at least 2ℓ steps. \square

We use the lemma above to determine, for each prime p , an *optimal* set \mathcal{N} for which we perform the detection of (N, N) -splittings during Algorithm 4.

Let c_{step} be the number of \mathbb{F}_p -multiplications required to take a step in $\Gamma_2(2; p)$ using Algorithm 2, and let c_{ig} be the number of \mathbb{F}_p -multiplications required to compute $\alpha(C)$ using `InvariantsFromWeierstrassPoints` (see Remark 5). Finally, letting $c_{\text{split}}(N)$ be the total number of \mathbb{F}_p -multiplications required by Algorithm 3 (see Proposition 4 and Remark 7), we obtain the following lemma.

Lemma 5. *For a subset $\mathcal{N} \subseteq \{2, 3, \dots, 11\}$, the number of \mathbb{F}_p -multiplications required to run Steps 7-15 of Algorithm 4 is at most*

$$\text{cost}_{\mathcal{N}} := \begin{cases} c_{\text{step}} + c_{\text{ig}} + \sum_{N \in \mathcal{N}} c_{\text{split}}(N) & \text{if } \mathcal{N} \neq \emptyset, \\ c_{\text{step}} & \text{otherwise.} \end{cases}$$

Proof. Given input defining a genus 2 curve C/\mathbb{F}_{p^2} if $\mathcal{N} = \emptyset$ then Steps 7-15 of Algorithm 4 require a single call to `Step(a, bits)`, taking c_{step} \mathbb{F}_p -multiplications.

Otherwise, Step 7 calls `InvariantsFromWeierstrassPoints` taking c_{ig} multiplications in \mathbb{F}_p . For each $N \in \mathcal{N}$, the contents of the for-loop (i.e., Steps 8-11) require $c_{\text{split}}(N)$ multiplications in \mathbb{F}_p . Finally Steps 12-15 call `Step(a, bits)`, again requiring c_{step} \mathbb{F}_p -multiplications. \square

We consider subsets of $\{2, \dots, 11\}$ satisfying the hypotheses of Lemma 4. As a precomputation, amongst these subsets we determine the optimal set \mathcal{N} for Algorithm 4 by choosing \mathcal{N} to minimise the number of \mathbb{F}_p -multiplications per node revealed (either visited by the Richelot walk or revealed by `lsSplit`). That is, we choose the \mathcal{N} that minimises the ratio $\frac{\text{cost}_{\mathcal{N}}}{\text{nodes}_{\mathcal{N}}}$.

6.3 A bound on the cost of the SplitSearcher algorithm

We now discuss a heuristic upper bound for the concrete cost of finding a splitting of a genus 2 Jacobian using the `SplitSearcher` algorithm combined with an optimised walk in $\Gamma_2(2; p)$.

First recall that our function `InvariantsFromWeierstrassPoints` terminates with 291 \mathbb{F}_p -multiplications and 1 \mathbb{F}_p inversion. Bounding this inversion by $2\log_2(p)$ \mathbb{F}_p -multiplications (i.e., by the worst case where the binary expansion of the exponent consists only of 1's), we have $c_{\text{ig}} \leq 291 + 2\log_2(p)$.

We now assume that the cost of `IsSplit` is bounded by the cost of its first 3 steps (see Proposition 4 and Table 3 bounds depending only on N , and Remark 7 for a justification). Finally `RIsog` requires 63 \mathbb{F}_p -multiplications and 3 calls to `InvSqrt` which costs at most $22 + 4\log_2(p)$ \mathbb{F}_p -multiplications (with the $\log_2(p)$ terms arising from 2 exponentiations). In particular, `RIsog` costs at most $129 + 12\log_2(p)$ \mathbb{F}_p -multiplications.

For primes of at least 150 bits, the set $\mathcal{N} = \{4, 6\}$ is the optimal set discussed in Section 6.2, and we obtain an upper bound of

$$\frac{14\log_2(p) + 34490}{664} \tag{3}$$

\mathbb{F}_p -multiplications per node revealed (assuming the heuristics from Remarks 6 and 9). If we assume that the proportion of product nodes (among nodes inspected by Algorithm 4) is equal⁹ to $5/p$ we would expect that Algorithm 4 requires

$$\left(\frac{14\log_2(p) + 34490}{5 \cdot 664}\right)p + O(\log_2(p)) \tag{4}$$

\mathbb{F}_p -multiplications before encountering a product node.

7 Efficient detection of endomorphisms

In this section, we show how our algorithm `SplitSearcher` can be extended to detect endomorphisms. Recall that for the detection of (N, N) -splittings we work with the Humbert surface \mathcal{H}_{N^2} of square discriminant N^2 . The key observation that sets up the work in this section is the following: the methods from Section 5 apply directly to surfaces \mathcal{H}_D , even when D is not a perfect square. Using this observation, we obtain an algorithm that efficiently detects endomorphisms of degree D for fundamental discriminants $D < 100$.

Though these algorithms are constructed in a manner nearly identical to Sections 5 and 6, we provide details here for completeness and to set the stage for the experimental data presented in Section 8.

7.1 An algorithm to detect endomorphisms

In Section 5.2, we presented a method to efficiently detect whether the Jacobian J_C of a genus 2 curve C is (N, N) -isogenous to a product of elliptic curves, by determining whether $[C]$ lies on the Humbert surface \mathcal{H}_D defined in Section 4.3.

⁹This is the expected proportion of product nodes in a random walk in $\Gamma_2(2, p)$, see [27, §6.2]. However, preliminary experiments (see Table 5 and Section 8.2) indicate that in our walk (taking only *good extensions*) the proportion may be closer to $1/p$.

Now consider instead the Humbert surface \mathcal{H}_D of fundamental discriminant D . By Theorem 1 the Jacobian J_C has RM by \mathcal{O}_D (over \bar{K}) if and only if $[C] \in \mathcal{H}_{N^2}(\bar{K})$. In this case, we use the explicit formulæ for the maps φ_D given by Elkies and Kumar [23] for $1 < D < 100$.

Indeed, let $f_i^{(D)}(r, s)$ and $P_{i,j}^D$ be as in Proposition 2 and Lemma 3. By Proposition 2, to show that J_C has RM by \mathcal{O}_D , it suffices to show there exists a point $P \in \mathcal{A}_2(\bar{\mathbb{F}}_p)$ such that $f_1^{(D)}(r, s)$, $f_2^{(D)}(r, s)$ and $f_3^{(D)}(r, s)$ vanish at P . This is analogous to the situation in Section 5 and, given a genus 2 curve $C/\bar{\mathbb{F}}_p$, we can use the methods described in Section 5.1 to perform this detection. As in Section 5.2 let $\tilde{P}_{i,j}^{(D)}$ be the polynomial obtained from $P_{i,j}^{(D)}$ by reducing its coefficients mod p .

By direct analogy to `lsSplit` (see Algorithm 3) we have the algorithm `DetectEnd`, which we present in Algorithm 5.

Algorithm 5 `DetectEnd`($\alpha(C), \tilde{P}_{1,2}^{(D)}, \tilde{P}_{2,3}^{(D)}, D$):

Input: A tuple $\alpha(C) = (\alpha_1(C), \alpha_2(C), \alpha_3(C))$, the polynomials $\tilde{P}_{1,2}^{(D)}, \tilde{P}_{2,3}^{(D)} \in \mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3][r]$, and a discriminant $D > 1$ satisfying either **(D1)** or **(D2)**.

Output: A boolean that is true if J_C has RM by \mathcal{O}_D .

```

1:  $(\tilde{P}_{1,2})_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{P}_{1,2}^{(D)}, \alpha(C))$ 
2:  $(\tilde{P}_{2,3})_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{P}_{2,3}^{(D)}, \alpha(C))$ 
3:  $g \leftarrow \text{InvFreeGCD}((\tilde{P}_{1,2})_{[C]}, (\tilde{P}_{2,3})_{[C]})$ 
4: if  $\deg g \geq 1$  then
5:    $s_0 \leftarrow \text{ComputeRoot}(g)$ 
6:    $(\tilde{f}_1)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_1^{(D)}, \alpha(C))$ 
7:    $(\tilde{f}_2)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_2^{(D)}, \alpha(C))$ 
8:    $(\tilde{f}_3)_{[C]} \leftarrow \text{EvalCoeffs}(\tilde{f}_3^{(D)}, \alpha(C))$ 
9:    $h \leftarrow \text{InvFreeGCD}(\text{InvFreeGCD}((\tilde{f}_1)_{[C]}(r, s_0), (\tilde{f}_2)_{[C]}(r, s_0)), (\tilde{f}_3)_{[C]}(r, s_0))$ 
10:  if  $\deg h \geq 1$  then
11:     $r_0 \leftarrow \text{ComputeRoot}(h)$ 
12:    bool  $\leftarrow \text{lgNonzero}(r_0, s_0)$ 
13:    if bool == true then
14:      return true
15: return false
```

As with Proposition 4, we have the following proposition which gives the cost of `DetectEnd`.

Proposition 5. *Let $1 < D < 100$ be a fundamental discriminant, and let $\text{mons}(D)$ be the set of monomials in $\alpha_1, \alpha_2, \alpha_3$ appearing in the coefficients of $\tilde{P}_{1,2}^{(D)}$ and $\tilde{P}_{2,3}^{(D)}$ (which lie in $\mathbb{F}_p[\alpha_1, \alpha_2, \alpha_3]$). For each $i = 1, 2, 3$, let*

$$d_i(D) = \max(\{\text{degree of } \alpha_i \text{ in } m \mid m \in \text{mons}(D)\}).$$

The cost of steps 1–3 in Algorithm 5 (with input D) is at most

$$3(d_1(D) + d_2(D) + d_3(D)) + 6m(D) + 2M(D) + \frac{3}{2}(d_P(D) + 2)(d_P(D) + 3) - 27$$

\mathbb{F}_p -multiplications, where $d_P(D) = \deg \tilde{P}_{1,2}^{(D)} + \deg \tilde{P}_{2,3}^{(D)}$, $m(D) = \#\text{mons}(D)$, and $M(D)$ is the number of monomials in $\alpha_1, \alpha_2, \alpha_3$ appearing in the coefficients of $\tilde{P}_{1,2}^{(D)}$ and $\tilde{P}_{2,3}^{(D)}$ counting repetitions.

In Table 4, we give the values of $d_1(D), d_2(D), d_3(D), m(D), M(D)$, and $d_P(D)$ for some $1 < D < 100$. Combining this with Proposition 5, we obtain the number of \mathbb{F}_p multiplications required to run `DetectEnd` for a fixed D .

D	$d_1(D)$	$d_2(D)$	$d_3(D)$	$m(D)$	$M(D)$	$d_P(D)$	Total $\#\mathbb{F}_p$ mults.
5	1	2	1	6	15	4	114
8	3	5	3	21	117	11	639
12	2	3	2	13	154	22	1280
13	3	5	3	21	228	22	1488
17	8	12	8	121	2076	44	8178
21	8	12	8	121	2492	44	9010
24	5	7	5	56	1956	69	11940
28	9	13	9	154	6795	88	26865
29	6	8	6	78	1981	66	11501
33	10	14	10	190	10375	132	49100
37	6	10	6	64	2896	110	25199
41	10	14	10	190	13402	176	75812
44	13	18	13	316	20040	154	78819
53	8	12	8	131	10380	154	58341
56	14	20	14	350	30534	220	137544
57	14	22	14	320	42309	308	231276
60	15	24	15	415	45272	264	199702
65	12	16	12	271	48240	352	286704
69	10	14	10	190	24775	264	157298
73	16	24	16	433	97830	484	553422
76	17	23	17	518	106063	418	480608
77	10	16	10	196	21193	264	150176
85	20	32	20	715	143863	396	530408
88	16	22	16	459	110859	506	612465
89	20	28	20	695	185100	572	869622
92	19	26	19	645	124159	440	546062
93	20	32	20	715	124527	396	491736
97	28	40	28	1311	572417	924	2440564

Table 4. Values of $d_1(D), d_2(D), d_3(D), m(D), M(D)$, and $d_P(D)$ for fundamental discriminants $D < 100$. The final column lists the number of \mathbb{F}_p -multiplications from Proposition 5.

Remark 10. Combining our optimised algorithm for taking steps in the $(2, 2)$ -isogeny graph with Algorithm 5 (which detects whether the current node has RM by \mathcal{O}_D), we obtain a new algorithm by direct analogy with Algorithm 4, which we call **SplEndid**. More precisely, **SplEndid** takes as input of a genus 2 curve C/\mathbb{F}_{p^2} with superspecial Jacobian and a set \mathcal{D} of discriminants satisfying either **(D1)** or **(D2)** and outputs a path $J_C \rightarrow J_{C'}$ in $\Gamma_2(2; p)$ such that $J_{C'}$ has an RM by \mathcal{O}_D for some $D \in \mathcal{D}$.

The output of **SplEndid** gives non-trivial information about the endomorphism ring of $J_{C'}$. As discussed in Section 1, the utility of this information in the broader picture of the superspecial isogeny problem is not known. Nevertheless, to begin to compare the dimension 2 isogeny problem and the problem of finding an endomorphism on a superspecial Jacobian, it seems natural to compare the relative performances of **SplitSearcher** and **SplEndid** as p grows. For **SplitSearcher** the set \mathcal{N} passed into the algorithm containing the integers N on which the detection is performed can be chosen to be (heuristically) *optimal*. However, in the case of **SplEndid** it is unclear how to obtain the optimal set of discriminants D . Indeed, we are not yet aware of any heuristics on the proportion of nodes in the graph that have RM by \mathcal{O}_D , and thus do not know how many nodes we “reveal” when running **DetectEnd** at each step for a specific D . In Section 8.2 we give some experimental evidence towards an appropriate heuristic.

8 Experimental results

We present some experimental data for the **SplitSearcher** algorithm in §8.1 and for the **SplEndid** algorithm in §8.2. We conclude the paper with some possible improvements to these algorithms in §8.3.

8.1 SplitSearcher results

We conducted experiments over both small and large primes, and the results are reported in Tables 5 and 6, respectively.

The small prime experiments were conducted so that we could run multiple instances of the full $\tilde{O}(p)$ search for product curves to completion. The four Mersenne primes of the form $p = 2^m - 1$ with $m \in \{13, 17, 19, 31\}$ were chosen as the field characteristics, and instances of the product-finding problem were generated by taking a chain of 40 randomised Richelot isogenies away from the superspecial abelian surface¹⁰ corresponding to $C/\mathbb{F}_p: y^2 = x^5 + x$. For the three smaller primes, 256 instances were generated, while for $p = 2^{31} - 1$, we generated 10 such instances; each instance is specified by a 6-tuple of Weierstrass points (see Section 3.1). All of the instances were solved once using the original walk in $\Gamma_2(2; p)$ described in Section 3. and again using our improved **SplitSearcher** algorithm described in Section 6. For all four of these primes, the set $\mathcal{N} = \{2, 3\}$

¹⁰The shapes of the primes chosen in both tables is of little consequence: we merely made consistent choices of the prime shape so that the same form of superspecial starting surface could be used throughout the experiments.

was optimal for use in `SplitSearcher`. In Table 5 we report the average number of steps taken in $\Gamma_2(2; p)$ for both algorithms, as well as the average number of \mathbb{F}_p -multiplications required to solve the problem. In the case of `SplitSearcher`, we additionally report the average number of nodes searched. This includes both the nodes that were walked on and those that were inspected using our (N, N) -splitting detection¹¹. As we might expect, this is always relatively close to the number of steps taken in the Richelot-only walk.

Prime p	Walks in $\Gamma_2(2; p)$ without additional searching [16] (optimised in Section 3)			Walks in $\Gamma_2(2; p)$ w. <code>SplitSearcher</code> in $\Gamma_2(N; p)$ This work			Improv. factor
	No. inst. solved	Av. steps taken	Av. \mathbb{F}_p mults.	Av. steps taken	Av. nodes covered	Av. \mathbb{F}_p mults.	
$2^{13} - 1$	256	6531	1839209	122	6536	188015	9.8x
$2^{17} - 1$	256	101812	33538079	2154	116305	3474579	9.7x
$2^{19} - 1$	256	475300	168095438	8593	464008	14104408	11.9x
$2^{31} - 1$	10	238694656	118336348672	4856252	262237639	8787389743	13.4x

Table 5. Solving the product-finding problem using Richelot isogeny walks in $\Gamma_2(2; p)$ only (left) vs. using Richelot isogeny walks in $\Gamma_2(2; p)$ together with `SplitSearcher` in $\Gamma_2(N; p)$ (right).

For cryptographically sized primes, we are unable to solve the product-finding problem, which is why Table 6 instead reports the number of nodes that were searched when the number of \mathbb{F}_p -multiplications was bounded at 10^8 . The main trend to highlight (in both tables) is that the speedup is increasing steadily as the prime grows in size: the number of \mathbb{F}_p -multiplications required for a single Richelot isogeny is proportional to the bitlength of p (due to the square root computations), while the number of \mathbb{F}_p -multiplications required to inspect the (N, N) -isogenous neighbours (after computing the Igusa–Clebsch invariants) remains fixed as p grows. This is also predicted by Equation 3, where the coefficient of the dominating $\log_2(p)$ term is $14/664$ versus 12.

Interestingly, as shown in Table 6 the set $\mathcal{N} = \{2, 3\}$ is optimal for the 50- and 100-bit primes, the set $\mathcal{N} = \{3, 4\}$ is optimal for the 150-bit prime, while the set $\mathcal{N} = \{4, 6\}$ takes over and reigns supreme for all other reported bitlengths. Our implementation can be used to obtain the same data for any other prime of interest, and the number of \mathbb{F}_p -multiplications used per node can be combined with the (average) number of nodes one expects to search through in order to get a very precise estimate on the concrete classical security of the superspecial isogeny problem.

¹¹Throughout this section we assume that the number of nodes revealed by `SplitSearcher` after s steps is equal to $s \cdot \text{nodes}_{\mathcal{N}}$. Indeed, as discussed in Remarks 6 and 9 an overcount should occur with very low probability. In particular, after $O(p)$ steps we would expect to overcount at most $o(p)$ nodes. This heuristic is also supported by the experiments reported in Table 5.

Prime p	$\log(p)$	Walks in $\Gamma_2(2; p)$ without additional searching [16] (optimised in Section 3)		Set $N \in \{\dots\}$	Walks in $\Gamma_2(2; p)$ with SplitSearcher in $\Gamma_2(N; p)$ This work		Improv. factor
		nodes per 10^8 mults.	\mathbb{F}_p -mults. per node		nodes per 10^8 mults.	\mathbb{F}_p -mults. per node	
$2^{11} \cdot 3^{24} - 1$	50	172712	579		2830951	35	16.5x
$2^{44} \cdot 3^{35} - 1$	100	85034	1176	{2, 3}	2076517	48	24.5x
$2^{27} \cdot 3^{77} - 1$	150	63492	1575	{3, 4}	1858912	54	29.2x
$2^{144} \cdot 3^{35} - 1$	200	42088	2376		1802816	55	43.2x
$2^{181} \cdot 3^{43} - 1$	250	34083	2934		1771608	56	52.4x
$5 \cdot 2^{193} \cdot 3^{66} - 1$	300	29317	3411		1745712	57	59.8x
$2^{201} \cdot 3^{94} - 1$	350	25581	3909		1719152	58	67.4x
$2^{231} \cdot 3^{106} - 1$	400	22753	4395		1694584	59	74.5x
$2^{204} \cdot 3^{155} - 1$	450	20729	4824		1672672	60	80.4x
$2^{113} \cdot 3^{244} - 1$	500	20239	4941		1667360	60	82.4x
$2^{293} \cdot 3^{162} - 1$	550	16835	5940	{4, 6}	1619552	62	95.8x
$5 \cdot 2^{299} \cdot 3^{188} - 1$	600	15679	6378		1599632	63	101.2x
$2^{404} \cdot 3^{155} - 1$	650	13848	7221		1562448	64	112.8x
$2^{83} \cdot 3^{389} - 1$	700	14530	6882		1580376	63	109.2x
$2^{477} \cdot 3^{172} - 1$	750	12046	8301		1517960	66	125.7x
$2^{107} \cdot 3^{437} - 1$	800	13228	7560		1548504	65	116.3x
$2^{166} \cdot 3^{431} - 1$	850	11968	8355		1515304	66	126.6x
$2^{172} \cdot 3^{459} - 1$	900	11427	8751		1500032	67	130.6x
$2^{536} \cdot 3^{261} - 1$	950	10233	9772		1443592	69	141.6x
$2^{721} \cdot 3^{176} - 1$	1000	8814	11346		1403752	71	159.8x

Table 6. The approximate number of multiplications required to search a single node using Richelot isogeny walks in $\Gamma_2(2; p)$ only (left) vs. using Richelot isogeny walks in $\Gamma_2(2; p)$ together with SplitSearcher in $\Gamma_2(N; p)$ (right). Note that the shape of the primes chosen has little effect on the multiplication counts since we expect to never find a splitting.

8.2 The distribution of RM nodes in the Richelot isogeny graph

Suppose that $D = N^2$ is a perfect square. Recall that the number $\overline{\mathbb{F}}_p$ -isomorphism classes of superspecial Jacobians and elliptic products is equal to $\#\mathcal{J}_2(p) = \frac{1}{2880}p^3 + O(p^2)$ and $\#\mathcal{E}_2(p) = \frac{1}{288}p^2 + O(p)$ respectively (as discussed in Section 2.2). Naively we predict that the expected number of steps in a random walk in $\Gamma_2(2; p)$ to arrive at a Jacobian which is (N, N) -isogenous to a product of elliptic curves is equal to $\frac{1}{\mathfrak{D}_N} \frac{\#\mathcal{J}_2(p)}{\#\mathcal{E}_2(p)} = \frac{p}{10\mathfrak{D}_N} + O(1)$, where \mathfrak{D}_N is as defined in Equation (1). On account of the fact that the automorphism group of a product $E_1 \times E_2$ of elliptic curves contains $C_2 \times C_2$, and is therefore not isomorphic to C_2 (which it is for a generic superspecial p.p. abelian surface) Florit–Smith [27, §6.2] argue for a more precise heuristic of $\frac{p}{5\mathfrak{D}_N} + O(1)$.

By simply extending this function to a general discriminant D , we arrive at the heuristic that the expected number steps in a random walk in $\Gamma_2(2; p)$ to arrive at an abelian surface with RM by \mathcal{O}_D is

$$\frac{p}{\mathfrak{D}_D} + O(1) \tag{5}$$

for some function $\tilde{\mathfrak{D}}_D = O(D^{3/2})$ which interpolates $\tilde{\mathfrak{D}}_{N^2} = 5\mathfrak{D}_N$.

A heuristic for $\tilde{\mathfrak{D}}_D$. The following heuristic description of the function $\tilde{\mathfrak{D}}_D$ was kindly provided to us by Jonathan Love. Suppose that D is a fundamental discriminant and that \mathcal{O}_D has narrow class number 1. Write $K = \text{Frac}(\mathcal{O}_D)$ for the field of fractions of \mathcal{O}_D . By combining [53, Theorem 2.5.35] and the Eichler mass formula ([70, Theorem 26.1.5]) we obtain an approximation for the number of superspecial p.p. abelian surfaces equipped with a choice of RM action $\mathcal{O}_D \hookrightarrow \text{End}^\dagger(A)$ given by

$$\frac{\zeta_K(2)D^{3/2}}{8\pi^4}p^2 + O(p) \tag{6}$$

where $\zeta_K(s)$ is the Dedekind zeta function. (The difference between (6) and the true count of superspecial p.p. abelian surfaces with RM by \mathcal{O}_D is accounted for by the Eichler class number formula [70, Theorem 30.8.6].) If $D \ll p$ one expects heuristically that for almost all superspecial p.p. abelian surfaces A/\mathbb{F}_{p^2} which *can be equipped* with RM by \mathcal{O}_D there exist exactly 2 choices of embeddings $\mathcal{O}_D \hookrightarrow \text{End}^\dagger(A)$. In particular, the expected number of steps in a random walk in $\Gamma_2(2;p)$ to arrive at an abelian surface with RM by \mathcal{O}_D is heuristically given by the formula in Equation (5) with

$$\tilde{\mathfrak{D}}_D = \frac{1}{2} \cdot \frac{360\zeta_K(2)}{\pi^4} D^{3/2}. \tag{7}$$

Experimental distribution of RM nodes. In Figure 1 we report on some preliminary experiments giving the average number of steps required to find a surface with RM by \mathcal{O}_D . These experiments appear to suggest that the leading term in Equation (5) should be closer to $p/\tilde{\mathfrak{D}}_D^{\text{expt}}$ where

$$\tilde{\mathfrak{D}}_D^{\text{expt}} = \begin{cases} \frac{5}{4}\mathfrak{D}_N & \text{if } D = N^2, \text{ and} \\ \frac{1}{3} \cdot \frac{360\zeta_K(2)}{\pi^4} D^{3/2} & \text{if } D \text{ is fundamental.} \end{cases} \tag{8}$$

Note that this adjusted heuristic matches our experimental data closely, even when \mathcal{O}_D does not have narrow class number 1.

Remark 11. Note that our walks in $\Gamma_2(2;p)$ take only *good extensions*. It is not clear to us whether this should have an effect on the number of endomorphisms encountered in a random walk.

8.3 Possible improvements and questions

There have been a number of choices made throughout this paper which open up possible avenues for improvement. We conclude by giving a non-exhaustive list of such improvements.

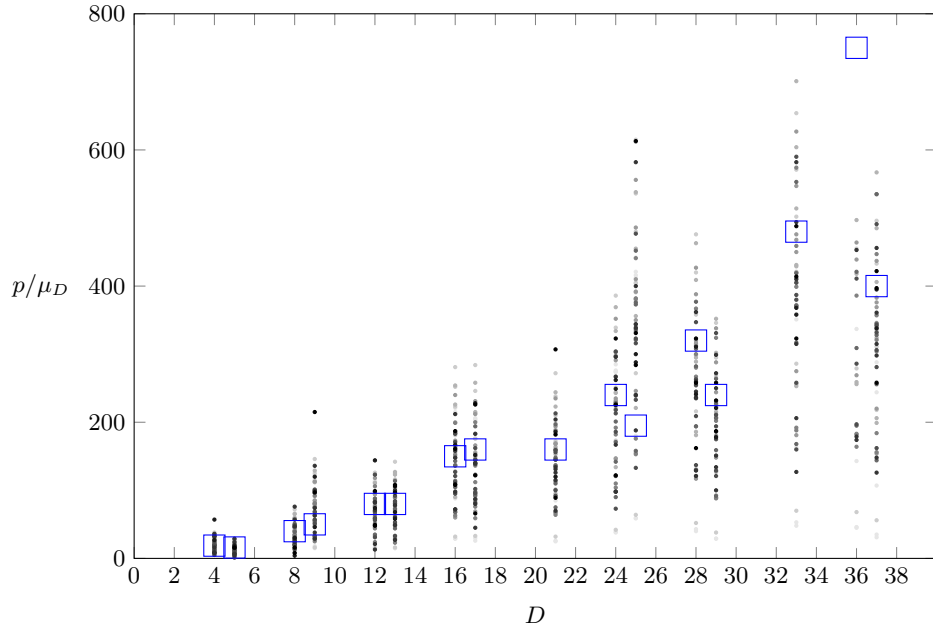


Fig. 1. The mean number of steps μ_D in $\Gamma_2(2; p)$ required to find a p.p. abelian surface with RM by \mathcal{O}_D for each discriminant $D < 40$ which is either fundamental or a perfect square. The vertical axis is normalised as p/μ_D to account for the heuristic dependence on p discussed in Section 8.2. Each plotted point corresponds to a separate prime p , and darker points correspond to larger values of p (which range from 12 to 23 bits). The squares are the values of $\tilde{\mathfrak{D}}_D^{\text{expt}}$.

1. The parametrisation of \mathcal{H}_{N^2} given by Kumar [46] may be altered through composition with a birational transformation of \mathbb{A}^2 . There may be better choices of parametrisations for our purposes, i.e., ones which minimise the degree of $P_{i,j}^{(N^2)}$. Furthermore, as detailed in Remark 4, there are many ways to normalise the Igusa–Clebsch invariants, though it is unclear to us which normalisations minimise the degrees that arise in the resultant computations.
2. Since the Weierstrass points of genus 2 curve with superspecial Jacobian are all \mathbb{F}_{p^2} -rational, it may be desirable to work with the Rosenhain invariants which may be computed more efficiently. To use our methods one would need to compute a birational model for the surface $\mathcal{H}_{N^2}(2)$ whose points parametrise optimally (N, N) -split Jacobians with full level 2 structure. One approach is described in [37].
3. It may be possible to improve the complexity of the evaluations performed by `EvalCoeffs` (see Section 5.2) by taking longer walks in the $(2, 2)$ -graph and then batching the evaluations using multi-point evaluation.
4. Knowledge of explicit equations for the surface \mathcal{H}_{N^2} for larger N would allow us to perform efficient detection of (N, N) -splittings beyond $N = 11$.

It may be possible to derive these from the pre-existing equations for the surfaces $Y_-(N^2)$ (which parametrise pairs of elliptic curves (N, N) -isogenous to a genus 2 Jacobian) in [25, Theorem 2.4], [26, Theorem 1.2], and [32, Theorem 1.1], or by extending Kumar’s computations.

5. As was pointed out to us by Thomas Decru, it is possible to detect $(2N, 2N)$ -splittings more efficiently by taking *partial* steps in the $(2, 2)$ -isogeny graph. Let C/\mathbb{F}_{p^2} be a genus 2 curve given by a Weierstrass equation $y^2 = (x - a_0) \cdots (x - a_5)$. While we cannot take a full step in $\Gamma_2(2; p)$ (recovering the factorisation of the Weierstrass sextic for each of the $(2, 2)$ -isogenous curves) without computing square roots, we can compute the Igusa–Clebsch invariants of *all* the neighbours of J_C using only a small number of \mathbb{F}_p -multiplications and a single batched inversion. In this case we may detect $(2N, 2N)$ -splittings of J_C by applying `lsSplit` to each of the $(2, 2)$ -neighbours of J_C . In the code attached to this article, this optimisation can be enabled by setting `split_after_22_flag = true`. In our implementation of this idea, and for the primes ranging between 50 and 1000 bits reported in Table 6, we observed additional improvement factors ranging between 1.3-1.6.

We conclude with a natural question arising from the experiments in Section 8.2. Recall that for an integer $N > 1$ we write $\Gamma_2(N; p)$ for the (N, N) -isogeny graph of superspecial p.p. abelian surfaces over $\overline{\mathbb{F}}_p$.

Question 2. Given a discriminant $D > 0$, what is the expected number of p.p. abelian surfaces with RM by \mathcal{O}_D encountered in a random walk in $\Gamma_2(N; p)$?

An answer to Question 2 is necessary to construct optimal sets \mathcal{D} of discriminants for the algorithm `SplEndid` defined in Remark 10.

Acknowledgements

We thank Thomas Decru, Tom Fisher, Jonathan Love, Aurel Page, Benjamin Smith, and John Voight for helpful conversations and comments on earlier versions of this paper. We also thank Thomas Decru for mentioning to us improvement 5 in Section 8.3 and Jonathan Love for the arguments in Section 8.2 pertaining to fundamental discriminants.

References

1. A. Basso, L. Maino, and G. Pope. Festa: Fast encryption from supersingular torsion attacks. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023*, pages 98–126, Singapore, 2023. Springer Nature Singapore.
2. O. Bolza. On binary sextics with linear transformations into themselves. *Amer. J. Math.*, 10(1):47–70, 1887.
3. W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory.

4. B. Brock. *Superspecial curves of genera two and three*. PhD thesis, Princeton University, 1994.
5. R. Bröker, E. W. Howe, K. E. Lauter, and P. Stevenhagen. Genus-2 curves and Jacobians with a given number of points. *LMS J. Comput. Math.*, 18(1):170–197, 2015.
6. N. Bruin and K. Doerksen. The arithmetic of genus two curves with $(4, 4)$ -split Jacobians. *Canad. J. Math.*, 63(5):992–1024, 2011.
7. N. Bruin and K. Doerksen. Electronic resources. <http://www.cecm.sfu.ca/~nbruin/splitigusa/>, 2011. Accessed Septemeber 2022.
8. W. Castryck and T. Decru. Multiradical isogenies. *Arithmetic, Geometry, Cryptography, and Coding Theory 2021*, 779:57, 2022.
9. W. Castryck and T. Decru. An efficient key recovery attack on SIDH. In *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 423–447. Springer, 2023.
10. W. Castryck, T. Decru, and B. Smith. Hash functions from superspecial genus-2 curves using Richelot isogenies. *Journal of Math. Crypt.*, 14(1):268–292, 2020.
11. D. X. Charles, E. Z. Goren, and K. E. Lauter. Families of Ramanujan graphs and quaternion algebras. In *Groups and symmetries*, volume 47 of *CRM Proc. Lecture Notes*, pages 53–80. Amer. Math. Soc., Providence, RI, 2009.
12. D. X. Charles, K. E. Lauter, and E. Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, 2009.
13. M. Chen, A. Leroux, and L. Panny. SCALLOP-HD: Group action from 2-dimensional isogenies. In Q. Tang and V. Teague, editors, *Public-Key Cryptography – PKC 2024*, pages 190–216, Cham, 2024. Springer Nature Switzerland.
14. M. Corte-Real Santos, C. Costello, and S. Frengley. An algorithm for efficient detection of (N, N) -splittings and its application to the isogeny problem in dimension 2. In *Public-key cryptography—PKC 2024. Part III*, volume 14603 of *Lecture Notes in Computer Science*, pages 157–189. Springer, Cham, 2024.
15. M. Corte-Real Santos, C. Costello, and J. Shi. Accelerating the Delfs–Galbraith algorithm with fast subfield root detection. In *Annual International Cryptology Conference*, pages 285–314. Springer, 2022.
16. C. Costello and B. Smith. The supersingular isogeny problem in genus 2 and beyond. In *PQ Crypto*, pages 151–168. Springer, 2020.
17. D. A. Cox, J. Little, and D. O’Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer, 2015.
18. P. Dartois, A. Leroux, D. Robert, and B. Wesolowski. SQIsignHD: New dimensions in cryptography. In M. Joye and G. Leander, editors, *Advances in Cryptology – EUROCRYPT 2024*, pages 3–32, Cham, 2024. Springer Nature Switzerland.
19. C. Delfs and S. D. Galbraith. Computing isogenies between supersingular elliptic curves over \mathbb{F}_p . *Designs, Codes and Cryptography*, 78(2):425–440, 2016.
20. Pierre Deligne and Georgios Pappas. Singularités des espaces de modules de Hilbert, en les caractéristiques divisant le discriminant. *Compositio Math.*, 90(1):59–79, 1994.
21. M. Djukanović. *Split Jacobians and Lower Bounds on Heights*. PhD thesis, Leiden University and L’Université de Bordeaux, 11 2017. <https://hdl.handle.net/1887/54944>.
22. M. Djukanović. Families of $(3, 3)$ -split Jacobians. *arXiv e-prints arXiv:1811.10075*, November 2018.
23. N. D. Elkies and A. Kumar. K3 surfaces and equations for Hilbert modular surfaces. *Algebra Number Theory*, 8(10):2297–2411, 2014.

24. L. De Feo, S. Dobson, S. D. Galbraith, and L. Zobernig. SIDH proof of knowledge. In *ASIACRYPT*. Springer, 2022.
25. T. Fisher. On families of 13-congruent elliptic curves. *arXiv e-prints arXiv:1912.10777*, December 2019.
26. T. Fisher. On pairs of 17-congruent elliptic curves. *arXiv e-prints arXiv:2106.02033*, June 2021.
27. E. Florit and B. Smith. Automorphisms and isogeny graphs of abelian varieties, with applications to the superspecial Richelot isogeny graph. In *Arithmetic, Geometry, Cryptography, and Coding Theory 2021*, 2021.
28. Enric Florit and Benjamin Smith. An atlas of the Richelot isogeny graph. In *Theory and Applications of Supersingular Curves and Supersingular Abelian Varieties*, volume B90 of *RIMS Kôkyûroku Bessatsu*, pages 195–219. Res. Inst. Math. Sci. (RIMS), Kyoto, 2022.
29. E. V. Flynn and Y. B. Ti. Genus two isogeny cryptography. In *PQ Crypto*, pages 286–306. Springer, 2019.
30. D. Freeman and K. Lauter. Computing endomorphism rings of Jacobians of genus 2 curves over finite fields. In *Algebraic geometry and its applications*, volume 5 of *Ser. Number Theory Appl.*, pages 29–66. World Sci. Publ., Hackensack, NJ, 2008.
31. S. Frengley. *Explicit moduli spaces for curves of genus 1 and 2*. PhD thesis, University of Cambridge, December 2023.
32. S. Frengley. On 12-congruences of elliptic curves. *Int. J. Number Theory*, 20(2):565–601, 2024.
33. G. Frey and E. Kani. Curves of genus 2 covering elliptic curves and an arithmetical application. In *Arithmetic algebraic geometry (Texel, 1989)*, volume 89 of *Progr. Math.*, pages 153–176. Birkhäuser Boston, Boston, MA, 1991.
34. P. Gaudry and É. Schost. On the invariants of the quotients of the Jacobian of a curve of genus 2. In *Applied algebra, algebraic algorithms and error-correcting codes (Melbourne, 2001)*, volume 2227 of *Lecture Notes in Comput. Sci.*, pages 373–386. Springer, Berlin, 2001.
35. E. Z. Goren and J. R. Love. Supersingular elliptic curves, quaternion algebras and applications to cryptography. *arXiv e-prints arXiv:2410.06123*, October 2024.
36. Eyal Z. Goren and Kristin E. Lauter. Evil primes and superspecial moduli. *Int. Math. Res. Not.*, pages Art. ID 53864, 19, 2006.
37. D. Gruenewald. Computing Humbert surfaces and applications. In *Arithmetic, geometry, cryptography and coding theory 2009*, volume 521 of *Contemp. Math.*, pages 59–69. Amer. Math. Soc., Providence, RI, 2010.
38. H. Hida. *Hilbert modular forms and Iwasawa theory*. Oxford Mathematical Monographs. The Clarendon Press, Oxford University Press, Oxford, 2006.
39. T. Ibukiyama and T. Katsura. On the field of definition of superspecial polarized abelian varieties and type numbers. *Compositio Math.*, 91(1):37–46, 1994.
40. T. Ibukiyama, T. Katsura, and F. Oort. Supersingular curves of genus two and class numbers. *Compositio Math.*, 57(2):127–152, 1986.
41. J. Igusa. Arithmetic variety of moduli for genus two. *Ann. of Math. (2)*, 72:612–649, 1960.
42. J. Igusa. On Siegel modular forms of genus two. *Amer. J. Math.*, 84:175–200, 1962.
43. B. W. Jordan and Y. Zaytman. Isogeny graphs of superspecial abelian varieties and generalized Brandt matrices. *arXiv preprint arXiv:2005.09031*, 2020.
44. D. Kohel, K. Lauter, C. Petit, and J. Tignol. On the quaternion-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17(A):418–432, 2014.
45. R. M. Kuhn. Curves of genus 2 with split Jacobian. *Trans. Amer. Math. Soc.*, 307(1):41–49, 1988.

46. A. Kumar. Hilbert modular surfaces for square discriminants and elliptic subfields of genus 2 function fields. *Research in the Mathematical Sciences*, 2(1):1–46, 2015.
47. S. Kunzweiler. Efficient computation of $(2^n, 2^n)$ -isogenies. *Des. Codes Cryptogr.*, 92(6):1761–1802, 2024.
48. J. Love and D. Boneh. Supersingular curves with small noninteger endomorphisms. *Open Book Series*, 4(1):7–22, 2020. Appendices available at <https://arxiv.org/pdf/1910.03180.pdf>.
49. D. Lubicz and D. Robert. Fast change of level and applications to isogenies. In *ANTS-XV*, 2022.
50. K. Magaard, T. Shaska, and H. Völklein. Genus 2 curves that admit a degree 5 map to an elliptic curve. *Forum Math.*, 21(3):547–566, 2009.
51. L. Maino, C. Martindale, L. Panny, G. Pope, and B. Wesolowski. A direct key recovery attack on SIDH. In *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 448–471. Springer, 2023.
52. J. Mestre. Construction de courbes de genre 2 a partir de leurs modules. In *Effective methods in algebraic geometry*, volume 94 of *Progr. Math.*, pages 313–334. Springer, 1990.
53. M.-H. Nicole. *Superspecial abelian varieties, theta series and the Jacquet–Langlands correspondence*. PhD thesis, McGill University, 2005. <https://escholarship.mcgill.ca/concern/theses/f1881m434>.
54. F. Oort. A stratification of a moduli space of abelian varieties. In *Moduli of abelian varieties (Texel Island, 1999)*, volume 195 of *Progr. Math.*, pages 345–416. Birkhäuser, Basel, 2001.
55. A. Page and B. Wesolowski. The supersingular endomorphism ring and one endomorphism problems are equivalent. In *Advances in cryptology—EUROCRYPT 2024. Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pages 388–417. Springer, Cham, 2024.
56. M. Rapoport. Compactifications de l’espace de modules de Hilbert–Blumenthal. *Compositio Math.*, 36(3):255–335, 1978.
57. D. Robert. Breaking SIDH in polynomial time. In *EUROCRYPT 2023*, volume 14008 of *Lecture Notes in Computer Science*, pages 472–503. Springer, 2023.
58. M. Scott. A note on the calculation of some functions in finite fields: Tricks of the trade. *Cryptology ePrint Archive*, 2020.
59. T. Shaska. Curves of genus 2 with (n, n) decomposable Jacobians. *Journal of Symbolic Computation*, 31(5):603–617, 2001.
60. T. Shaska. *Curves of genus two covering elliptic curves*. University of Florida, 2001.
61. T. Shaska. Genus 2 fields with degree 3 elliptic subfields. *Forum Math.*, 16(2):263–280, 2004.
62. T. Shaska and H. Völklein. Elliptic subfields and automorphisms of genus 2 function fields. In *Algebra, arithmetic and geometry with applications*, pages 703–723. Springer, 2004.
63. T. Shaska, G. S. Wijesiri, S. Wolf, and L. Woodland. Degree 4 coverings of elliptic curves by genus 2 curves. *Albanian J. Math.*, 2(4):307–318, 2008.
64. J. H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer, Dordrecht, second edition, 2009.
65. C. Springer. Computing the endomorphism ring of an ordinary abelian surface over a finite field. *Journal of Number Theory*, 202:430–457, 2019.
66. K. Takashima. Efficient algorithms for isogeny sequences and their cryptographic applications. In *Mathematical modelling for next-generation cryptography*, pages 97–114. Springer, 2018.

67. K. Takashima and R. Yoshida. An algorithm for computing a sequence of Richelot isogenies. *Bulletin of the Korean Mathematical Society*, 46(4):789–802, 2009.
68. G. van der Geer. *Hilbert modular surfaces*, volume 16 of *Ergebnisse der Mathematik und ihrer Grenzgebiete (3) [Results in Mathematics and Related Areas (3)]*. Springer-Verlag, Berlin, 1988.
69. P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptol.*, 12(1):1–28, 1999.
70. John Voight. *Quaternion algebras*, volume 288 of *Graduate Texts in Mathematics*. Springer, Cham, [2021] ©2021.
71. B. Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science—FOCS 2021*, pages 1100–1111. IEEE Computer Society, Los Alamitos, CA, 2022.