Efficient supersingularity testing over $\mathbb{F}_p$ and CSIDH key validation

Gustavo Banegas[1], Valerie Gilchrist[2,1,*], Benjamin Smith[1]

[1]Inria and Laboratoire d'Informatique de l'École polytechnique, Institut Polytechnique de Paris, Palaiseau, France
[2]University of Waterloo, Canada

Abstract *Many public-key cryptographic protocols, notably non-interactive key exchange (NIKE), require incoming public keys to be validated to mitigate some adaptive attacks. In CSIDH, an isogeny-based post-quantum NIKE, a key is deemed legitimate if the given Montgomery coefficient specifies a supersingular elliptic curve over the prime field. In this work, we survey the current supersingularity tests used for CSIDH key validation, and implement and measure two new alternative algorithms. Our implementation shows that we can determine supersingularity substantially faster, and using less memory, than the state-of-the-art.*

Keywords: Isogenies, Key validation, Supersingularity, Elliptic Curves

1 INTRODUCTION

The security of many public-key cryptosystems assumes that public keys are honestly generated: that is, that public keys have not been manipulated by adversaries. *Key validation* is the process of determining whether an incoming public key was plausibly constructed following the protocol.

The simplest example of this is in non-interactive key exchange (NIKE). Consider classic static Diffie–Hellman in a finite field: the system parameters fix a prime modulus p and a generator g in $\mathbb{F}_p$ for a subgroup $G = \langle g \rangle \subset \mathbb{F}_p^\times$ of prime order r. Alice samples a long-term secret integer a, and binds to the corresponding public key $A = g^a$. An honest Bob computes his keypair $(B = g^b, b)$, and the shared secret is $S = A^b = B^a$. However, if G is a proper subgroup of $\mathbb{F}_p^\times$, then a dishonest Bob can choose some h in $\mathbb{F}_p^\times \setminus G$, of order $s \mid (p-1)/r$, and transmit the malformed public key $B' = B \cdot h$. The shared secret as computed by Alice is now $S' = (B')^a = S \cdot h^a$, while Bob derives the original $S = A^b$. The success or failure of subsequent encrypted communication tells Bob whether $S = S'$, and hence whether a is $0 \pmod{s}$.

To avoid leaking information on her long-term private key to adaptive adversaries, then, Alice must validate incoming public keys as being honestly generated. In the example above, this amounts to checking that Bob's public key really is an element of G; this can be done by checking that $B^r = 1$. In elliptic-curve Diffie–Hellman key exchange, key validation amounts to an analogous scalar multiplication by a (tiny or trivial) cofactor, *plus* a simple check that Bob's public key B really does encode a point on the curve.

Moving now to the post-quantum setting, the best-established NIKE candidate is CSIDH [4], a key exchange scheme based on the action of the ideal class group of $\mathbb{Z}[\sqrt{-p}]$ on the isogeny class of supersingular elliptic curves over $\mathbb{F}_p$ (recall that $\mathcal{E}/\mathbb{F}_p$ is *supersingular* if $\#\mathcal{E}(\mathbb{F}_p) = p + 1$; otherwise, it is *ordinary*). The CSIDH group action has also been used to construct other post-quantum public-key cryptosystems, including signatures [2, 8, 9], threshold schemes [10], and oblivious transfer [16].

Validating CSIDH public keys is therefore important for long-term post-quantum security. The fundamental problem is: we are given an element A in $\mathbb{F}_p$ corresponding to an elliptic curve with a *Montgomery model*

$$\mathcal{E}_A/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)\,, \tag{1}$$

and we must determine if $\mathcal{E}_A$ is in the orbit of the group action. By [4, Proposition 8], to validate A it suffices to

1. check that $A^2 \neq 4$ (that is, $\mathcal{E}_A$ is an elliptic curve and not a singular cubic), a trivial task; and
2. check that $\mathcal{E}_A$ is supersingular, a nontrivial and mathematically interesting task.

Key validation is not the bottleneck in CSIDH key exchange—it typically takes under 5% of the runtime—but it is still a critical problem to be solved efficiently. The main issue is supersingularity testing over $\mathbb{F}_p$, and we will focus entirely on this problem, ignoring the rest of the CSIDH cryptosystem and its derivatives (we refer the reader to [4, 5, 1] for further details and discussion). The only relevant detail is that in CSIDH,

$$p = 4\prod_{i=1}^{n} \ell_i - 1$$

*Corresponding Author: Valerie Gilchrist, vgilchrist@uwaterloo.ca

*Author list in alphabetical order; see `https://www.ams.org/profession/leaders/CultureStatement04.pdf`. This research was funded in part by the French Agence Nationale de la Recherche through ANR CIAO (ANR-19-CE48-0008) and the European Commission through H2020 SPARTA, `https://www.sparta.eu/`. Date of this document: 2022-07-26.

where the ℓ_i are small primes: this ensures that supersingular curves over $\mathbb{F}_p$ have rational points of order ℓ_i for $1 \leq i \leq n$, which can be used to compute the group action for ideals of smooth norm in a particularly efficient way. In particular, $p \equiv 3 \pmod 4$.

In this work we consider four algorithms for testing supersingularity over $\mathbb{F}_p$ in the context of CSIDH key validation. The algorithms are listed in Table 1. We have implemented and benchmarked each of them for the 512-bit prime p defined by the CSIDH-512 parameter set; our practical results are also summarized in Table 1.

Table 1: Supersingularity testing algorithms for elliptic curves over $\mathbb{F}_p$. For Algorithms 2 and 3, $p + 1 = 4 \prod_{i=1}^{n} \ell_i$. (The experimental setup for the performance measurements with CSIDH-512 parameters is detailed below.)

Test algorithm	Asymptotics		Supersingular input (valid keys)			Non-supersingular input (invalid keys)		
	Time ($\mathbb{F}_p$-ops)	Space ($\mathbb{F}_p$-elts)	MCycles: Mean	Median	Stack (B)	MCycles: Mean	Median	Stack (B)
Alg. 2 (Random point)	$O(n \log p)$	$O(1)$	63.4	62.2	2890	65.3	62.9	2890
Alg. 3 (Product tree)	$O((\log n)(\log p))$	$O(\log n)$	6.7	6.1	4344	1.7	1.6	3896
Alg. 5 (Sutherland)	$O(\log^2 p)$	$O(1)$	35.4	35.1	2696	0.8	0.4	2696
Alg. 6 (Doliskani)	$O(\log p)$	$O(1)$	4.5	4.7	3280	2.9	2.8	3264

Algorithms 2 and 3, detailed in §3, are elementary algorithms based on determining the order of a random point on $\mathcal{E}_A$. These are the algorithms used in existing CSIDH implementations, and we include them as a baseline for comparison. These are the only supersingularity tests here that rely on the special form of CSIDH primes (they require the factorization of $p + 1$).

Algorithm 5, described in §4, is our variant of Sutherland's test [22] for curves over $\mathbb{F}_{p^2}$, adapted and optimized for Montgomery models over $\mathbb{F}_p$. This test is based on distinguishing between the 2-isogeny graph structures of supersingular and ordinary elliptic curves over $\mathbb{F}_{p^2}$. While the asymptotic complexity of this algorithm is quadratic in $\log p$ (the others are linear or quasi-linear), it has very good constants, and we find that it performs surprisingly well in practice. It is extremely easy to implement, and requires very little memory. It is also the only one of the four tests considered here that always produces definitive proof of supersingularity.

Algorithm 6, described in §5, is our variant of Doliskani's test [12] for curves over $\mathbb{F}_{p^2}$, which uses Polynomial Identity Testing to distinguish between the p-th division polynomials of supersingular and ordinary curves. We have adapted and optimized this algorithm for Montgomery models over $\mathbb{F}_p$, drastically simplifying the division polynomial computation. This algorithm is particularly simple: it only requires a single scalar multiplication of a point over $\mathbb{F}_{p^2}$, followed by an easy field exponentiation. Algorithm 6 is a Monte Carlo algorithm with one-sided error: it may declare an ordinary curve supersingular, but the probability of this is in $O(1/p)$, which is virtually zero for cryptographic p: for the 512-bit p of CSIDH, a false-accept rate of 2^{-512} is more than covered by the claimed security level.

We will see that Algorithm 6 is the simplest and fastest supersingularity test for CSIDH public key validation.

Implementation and experimental results We implemented our algorithms in C for the 512-bit prime p of the CSIDH-512 parameter set, which was built from $n = 74$ small primes ℓ_i. For $\mathbb{F}_p$-arithmetic, we used the assembly code from the CTIDH library [1][1]. For $\mathbb{F}_{p^2}$-arithmetic, we used the "tricks" from [20], which we reproduce for easy reference in Appendix A. The implementation of Algorithm 3 was taken directly from the CTIDH library. Algorithms 2, 5, and 6 are our own implementations.

We ran our experiments on an Intel i7-10610U processor running at 4.90 GHz with TurboBoost and SpeedStep disabled, running Arch Linux with kernel 5.15.41-1-lts and GCC 12.1.0. Cycles were measured using the `bench` utility provided in the CSIDH code package. For our experiments, we generated 500 valid (supersingular) curves and 500 invalid (ordinary) curves. Table 1 presents the average and median runtime for each algorithm (in millions of cycles), and the maximum stack use (in bytes) for a complete run of the algorithm (including subroutines). We note that the algorithm used to compute square roots in $\mathbb{F}_{p^2}$ inside Algorithm 5 uses several temporary variables, which increases the stack footprint; this might be further optimized.

Elliptic curve models We focus on elliptic curves with Montgomery models, which are ubiquitous in isogeny-based cryptography. However, we discuss extensions of our methods to more general elliptic curves over $\mathbb{F}_p$ in §6.

Notation Throughout, p denotes a (large) odd prime, and q is a power of p. For every integer $m > 0$, we write

$$\operatorname{len}(m) := \lfloor \log_2 m \rfloor + 1 \qquad \text{(i.e., the bitlength of } m\text{)}.$$

[1]We used version `20210523`, available from `http://ctidh.isogeny.org/software.html`.

2 MONTGOMERY ARITHMETIC

We assume that the reader is familiar with basic elliptic curve arithmetic (see e.g. [21] for background). However, Algorithm 6 requires some fine detail on the Montgomery ladder algorithm [18], which is also a subroutine of Algorithms 2, 3, and CSIDH itself, so we take a moment to recall it here. (For further detail, see [7].)

In practice, most isogeny-based cryptosystems (including CSIDH) work with *Montgomery models*

$$\mathcal{E}_A : y^2 = x(x^2 + Ax + 1) .$$

We call the parameter A the *Montgomery coefficient*. We write $\oplus$ and $\ominus$ for addition and subtraction on $\mathcal{E}_A$.

Let P and Q be points on $\mathcal{E}_A$. Any three of $x(P)$, $x(Q)$, $x(P \ominus Q)$, and $x(P \oplus Q)$ determines the fourth, so we can define a *differential addition*

$$\mathtt{xADD} : (x(P), x(Q), x(P \ominus Q)) \longmapsto x(P \oplus Q)$$

and a pseudo-doubling operation

$$\mathtt{xDBL}_A : x(P) \longmapsto x([2]P) .$$

We can evaluate these maps on affine representatives for projective points as follows. Given points P and Q on $\mathcal{E}_A$, we write $(X_P : Z_P) = x(P)$, $(X_Q : Z_Q) = x(Q)$, $(X_\oplus : Z_\oplus) = x(P \oplus Q)$, and $(X_\ominus : Z_\ominus) = x(P \ominus Q)$ (recall that $x((X : Y : Z)) = (X : Z)$ if $Z \neq 0$, and $(1 : 0)$ if $Z = 0$). Now we can compute xADD using the formulæ

$$\begin{cases} X_\oplus = Z_\ominus \cdot [U + V]^2 , \\ Z_\oplus = X_\ominus \cdot [U - V]^2 \end{cases} \quad \text{where} \quad \begin{cases} U = (X_P - Z_P)(X_Q + Z_Q) , \\ V = (X_P + Z_P)(X_Q - Z_Q) . \end{cases} \tag{2}$$

Note that if we replace (X_P, Z_P) and (X_Q, Z_Q) with the projectively equivalent $(\lambda_P X_P, \lambda_P Z_P)$ and $(\lambda_Q X_Q, \lambda_Q Z_Q)$ in (2), then $(X_\oplus, Z_\oplus)$ becomes $((\lambda_P\lambda_Q)^2 X_\oplus, (\lambda_P\lambda_Q)^2 Z_\oplus)$.

Similarly, writing $(X_{[2]P} : Y_{[2]P} : Z_{[2]P})$ for $[2]P$, we can compute $\mathtt{xDBL}_A$ using

$$\begin{cases} X_{[2]P} = R \cdot S , \\ Z_{[2]P} = T \cdot (S + \frac{A+2}{4} T) \end{cases} \quad \text{where} \quad \begin{cases} R = (X_P + Z_P)^2 , \\ S = (X_P - Z_P)^2 , \\ T = 4X_P \cdot Z_P = Q - R . \end{cases} \tag{3}$$

If we replace (X_P, Z_P) with $(\lambda_P X_P, \lambda_P Z_P)$ in (3), then $(X_{[2]P}, Z_{[2]P})$ becomes $(\lambda_P^4 X_{[2]P}, \lambda_P^4 Z_{[2]P})$.

Algorithm 1 is the *Montgomery ladder*, which efficiently computes the map $(m, x(P)) \mapsto x([m]P)$.

Algorithm 1: The Montgomery ladder on the x-line $\mathbb{P}^1$ under $\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)$

Input: $A \in \mathbb{F}_q$, $m = \sum_{i=0}^{\beta-1} m_i 2^i$, and $(X_P, Z_P) \in \mathbb{F}_q^2$ with $X_P Z_P \neq 0$

Output: (X_m, Z_m) and (X_{m+1}, Z_{m+1}) in $\mathbb{F}_p(u)^2$ such that $(X_m : Z_m) = x([m]P)$ and $(X_{m+1} : Z_{m+1}) = x([m+1]P)$ where P is a point on $\mathcal{E}_A$ with $x(P) = (X_P : Z_P)$

Function Ladder(A, m, (X_P, Z_P))
- $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow ((1,0), (X_P, Z_P))$ // $(1 : 0) = x(0)$ and $(X_P : Z_P) = x(P)$
- **for** i *in* $(\beta - 1, \ldots, 0)$ **do** // Invariant: $\mathsf{R}_0 = x([\lfloor m/2^i \rfloor]P)$ and $\mathsf{R}_1 = x([\lfloor m/2^i \rfloor + 1]P)$
  - **if** $m_i = 0$ **then**
    - $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow (\mathtt{xDBL}_A(\mathsf{R}_0), \mathtt{xADD}(\mathsf{R}_0, \mathsf{R}_1, (X_P, Z_P)))$
  - **else**
    - $(\mathsf{R}_0, \mathsf{R}_1) \leftarrow (\mathtt{xADD}(\mathsf{R}_0, \mathsf{R}_1, (X_P, Z_P)), \mathtt{xDBL}_A(R_1))$
- **return** R_0 *and optionally* R_1 // $\mathsf{R}_0 = x([m]P)$ and $\mathsf{R}_1 = x([m+1]P)$

3 ELEMENTARY SUPERSINGULARITY TESTS

We begin by considering the two supersingularity tests that have been proposed for use with CSIDH [4]. These elementary tests are included as a point of reference when comparing performance with our new algorithms below; for detailed discussion, see [4, §5 and §8].

The goal of these two tests is to try to exhibit a point of order $N \mid (p + 1)$ with $N > 4\sqrt{p}$; then, the only multiple of N in the Hasse interval is $p + 1$, so we can conclude that the curve has order $p + 1$ and is therefore supersingular.

On the other hand, if we find a point whose order can be shown to *not* divide $p+1$, then we can immediately declare the curve ordinary. To efficiently determine (a divisor of) the order, we need to know the factorization of $p+1$, which in the case of CSIDH is $4\prod_{i=1}^{n} \ell_i$ with the ℓ_i very small.

Algorithm 2 proceeds in the simplest way: let u be a random element of $\mathbb{F}_p \setminus \{0\}$. Now u is the x-coordinate of a point P in $\mathcal{E}(\mathbb{F}_{p^2})$, which has exponent $p+1$. For each of the primes ℓ_i, we compute $Q_i = [(p+1)/\ell_i]P$.

- If $Q_i = 0$, then we learn nothing from ℓ_i;
- if $Q_i \neq 0$ but $[\ell_i]Q_i = 0$ then we know that ℓ_i divides the order of P;
- if $[\ell_i]Q_i \neq 0$ then the order of P cannot divide $p+1$, so we know the curve is ordinary.

Once we have accumulated enough ℓ_i that $\prod_i \ell_i > 4\sqrt{p}$, we can stop and declare the curve supersingular.

Algorithm 2 is quite simple to follow and has low memory requirements, but its expected runtime is rather slow. Indeed, each ℓ_i that we treat entails two `Ladder` calls, with $m = (p+1)/\ell_i$ and ℓ_i, and this adds up to approximately the cost of a `Ladder` with $m = p+1$. The asymptotic runtime is therefore $O(n \log p)$ $\mathbb{F}_p$-operations.

For a more concrete perspective: to minimise the total runtime before exceeding $4\sqrt{p}$ (or declaring ordinariness), we treat the ℓ_i from largest to smallest. If the curve is supersingular (which is the worst case for runtime), then we will need a little under half of the ℓ_i; that is, we expect an effort equivalent of around $n/2$ full-length `Ladder` calls over $\mathbb{F}_p$. Of course, in practice we can compute these point multiplications using precalculated optimal differential addition chains, rather than the ladder, but this is only a small improvement.

It is possible, though *extremely* improbable for cryptographic-size p, for Algorithm 2 to fail and return $\perp$. (This would imply that the random point has very small order.) If this happens, then we can simply re-run Algorithm 2 with a new random u.

Algorithm 2: Supersingularity testing for $\mathcal{E}/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)$ via random point multiplication [4, Algorithm 1]. Assumes the factorization $p+1 = 4\prod_{i=1}^{n} \ell_i$ is known, with $\ell_n > \cdots > \ell_1$.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False** or $\perp$
Function `IsSupersingular`(A)
  $u \leftarrow$ `Random`($\mathbb{F}_p \setminus \{0\}$)
  $N \leftarrow 1$
  for i *in* $(n, \dots, 1)$ **do**
    $(X, Z) \leftarrow$ `Ladder`(A, $(p+1)/\ell_i$, $(u, 1)$)
    $(X', Z') \leftarrow$ `Ladder`(A, ℓ_i, (X, Z))
    if $Z' \neq 0$ **then**
      return False
    if $Z \neq 0$ **then**
      $N \leftarrow N \cdot \ell_i$
    if $N > 4\sqrt{p}$ **then**
      return True
  return $\perp$

Algorithm 3 is a simple version of Algorithm 2 exploiting the fact that the various scalar multiplications are products of the same small primes, so we can compute them more efficiently using a classic product-tree structure. The algorithm proposed in [4, §8] traverses the product tree breadth-first. Algorithm 3, which is essentially the algorithm currently used in the CSIDH and CTIDH reference implementations, does this depth-first instead, handling the leaves corresponding to the largest ℓ_i first (the depth-first approach saves a lot of memory, and a little time too).

The product-tree approach is essentially a space-time tradeoff: we mutualise much of the effort of the scalar multiplications in the basic Algorithm 2, at the cost of storing the internal nodes of the product tree on the path to the current leaf. The depth of the tree is $\lceil \log_2 n \rceil$, so we have an asymptotic time complexity of $O(k \log n)$ $\mathbb{F}_p$-operations and a space complexity of $O(\log n)$ $\mathbb{F}_p$-elements.

Remark 1. *We warn the reader that the tests in this section do* not *work, or generalise, for elliptic curves over $\mathbb{F}_{p^2}$: supersingular curves over $\mathbb{F}_{p^2}$ do not have points of sufficiently large order to conclude on the group order.*

Algorithm 3: Supersingularity testing for $\mathcal{E}/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)$ via random point multiplication with an implicit product tree. Assumes the factorization $p + 1 = 4\prod_{i=1}^{n} \ell_i$ is known.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**

```
Function IsSupersingular(A)
    Function OrderRec(A, (X_Q, Z_Q), L, U, m)
        if U − L = 1 then                                  // At this point, Q = [(p+1)/ℓ_U]P
            if Z_Q = 0 then                                // In this case, we learn nothing
                return m
            (X_Q, Z_Q) ← Ladder(A, ℓ_U, (X_Q, Z_Q))
            if Z_Q = 0 then                                // ℓ_U divides the order of P
                return ℓ_U · m
            else                                           // [p+1]P ≠ 0:  not supersingular!
                return 0
        (X_L, Z_L) ← Ladder(A, ∏_{i=L+1}^{⌊(U+L)/2⌋} ℓ_i, (X_Q, Z_Q))
        m ← OrderRec(A, (X_L, Z_L), ⌊(U+L)/2⌋, U, m)
        if m > 4√p then
            return m
        (X_R, Z_R) ← Ladder(A, ∏_{i=⌊(U+L)/2+1⌋}^{R} ℓ_i, (X_Q, Z_Q))
        m ← OrderRec(A, (X_R, Z_R), L, ⌊(U+L)/2⌋, m)
        return m
    u ← Random(F_p)
    m ← OrderRec(A, (4u, 1), 0, n, 1)
    if m = 0 then
        return False
    else if m > 4√p then
        return True
    else
        return ⊥
```

4 ISOGENY VOLCANOES AND SUTHERLAND'S TEST

Our first non-elementary supersingularity test is Sutherland's algorithm [22], which actually detects supersingularity over $\mathbb{F}_{p^2}$. As such, we will need a slightly more evolved perspective on supersingularity before describing the algorithm. The facts stated here without proof are all covered in [21] (for supersingularity), and [15] and [13] (for isogeny graphs and volcanoes).

4.1 SUPERSINGULARITY IN GENERAL

Let $\mathcal{E} : y^2 = x^3 + a_2x^2 + a_4x + a_6$ be an elliptic curve over $\mathbb{F}_{p^e}$, and let $\pi : (x, y) \mapsto (x^{p^e}, y^{p^e})$ be the p^e-power Frobenius endomorphism. Like all endomorphisms, π satsifies a quadratic *characteristic polynomial* in the form

$$\chi_\pi(X) = X^2 - tX + p^e .$$

The integer t is called the *trace* of Frobenius, or simply the trace of $\mathcal{E}$; Hasse's theorem tells us that $|t| \le 2p^{e/2}$. Since the $\mathbb{F}_{p^e}$-rational points of $\mathcal{E}$ are precisely the points fixed by π, we have

$$\#\mathcal{E}(\mathbb{F}_{p^e}) = \chi_\pi(1) = p^e - t + 1 .$$

We say that $\mathcal{E}$ is supersingular if $p \mid t$; otherwise, $\mathcal{E}$ is ordinary. (In particular, in the case $e = 1$, Hasse's theorem implies that $\mathcal{E}$ is supersingular if and only if $t = 0$, if and only if $\#\mathcal{E}(\mathbb{F}_p) = p + 1$.) Equivalently, $\mathcal{E}$ is supersingular if $\mathcal{E}[p^r](\overline{\mathbb{F}}_p) = 0$ for all $r > 0$; for ordinary curves, $\mathcal{E}[p^r](\overline{\mathbb{F}}_p) \cong \mathbb{Z}/p^e\mathbb{Z}$.

If $\mathcal{E}$ is supersingular, then its multiplication-by-p endomorphism $[p]$ is purely inseparable, hence isomorphic to the p^2-power Frobenius isogeny, which is therefore isomorphic to an endomorphism. Hence, if $\mathcal{E}$ is supersingular, then the j-invariant of $\mathcal{E}$ must be in $\mathbb{F}_{p^2}$, and $\mathcal{E}$ is $\mathbb{F}_{p^e}$-isomorphic to a supersingular curve over $\mathbb{F}_{p^2}$. Thus, supersingularity testing over $\mathbb{F}_{p^e}$ reduces immediately to supersingularity testing over $\mathbb{F}_{p^2}$.

The ring $\mathbb{Z}[\pi]$ forms a subring of the endomorphism ring $\mathrm{End}(\mathcal{E})$. If $\mathcal{E}$ is ordinary, then $\mathrm{End}(\mathcal{E})$ is an order in the quadratic imaginary field $\mathbb{Q}(\pi) \cong \mathbb{Q}(\sqrt{t^2 - 4p^e})$. If $\mathcal{E}$ is supersingular, then $\mathrm{End}(\mathcal{E})$ is a maximal order in a quaternion algebra ramified at p and ∞.

4.2 2-ISOGENY GRAPHS

An *isogeny* $\phi : \mathcal{E} \to \mathcal{E}'$, is a non-zero morphism of elliptic curves. We say ϕ is a *d-isogeny* if the associated extension of function fields has degree d (we are only concerned with 2-isogenies in this paper).

If $\phi : \mathcal{E} \to \mathcal{E}'$ is a d-isogeny, then there exists a dual d-isogeny $\hat{\phi} : \mathcal{E}' \to \mathcal{E}$. The composition of two isogenies is another isogeny, and every elliptic curve has an isogeny to itself (the identity map, for example). Isogeny is therefore an equivalence relation: the set of all elliptic curves over $\mathbb{F}_{p^e}$ decomposes into *isogeny classes*. Isogenous curves have the same trace; in particular, supersingularity is preserved by isogeny.

The 2-isogeny graph of elliptic curves over $\mathbb{F}_{p^2}$ is the graph whose vertices are isomorphism classes of elliptic curves over $\mathbb{F}_{p^2}$, and where there is an edge between two vertices corresponding to (the isomorphism class of) each 2-isogeny between elliptic curves representing the vertices. While this graph is technically directed, away from the vertices of j-invariant 0 and 1728 it behaves exactly like an undirected graph.

The 2-isogeny graphs containing ordinary and supersingular curves have very different structures.

- The 2-isogeny graph of supersingular curves over $\mathbb{F}_{p^2}$ is a connected 3-regular graph.
- The 2-isogeny graph of ordinary curves over $\mathbb{F}_{p^2}$ decomposes into a set of *volcanoes*, each formed by a (possibly trivial) cycle whose vertices form the roots of a forest of regular binary trees, all of the same height.

Figure 1 illustrates an example of a 2-isogeny volcano. Following the terminology of [13], the cycle (at the top) is called the *crater*, and the leaves of the trees form the *floor*.

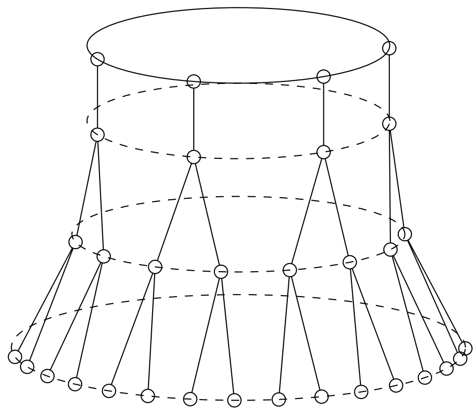

Figure 1: Example isogeny volcano.

As shown in [15], the volcano structure can be interpreted in terms of endomorphism rings. Let $\mathcal{E}/\mathbb{F}_{p^2}$ be an ordinary curve representing a vertex in a 2-isogeny volcano, let t be the trace of $\mathcal{E}$, and let $\Delta := t^2 - 4p^2$; then the algebra $\mathbb{Q}(\pi)$ generated by the Frobenius endomorphism π is isomorphic to the quadratic imaginary field $K := \mathbb{Q}(\sqrt{\Delta})$. Let Δ_0 be the fundamental discriminant of Δ, so $\Delta = c^2\Delta_0$ for some $c > 0$, which is the *conductor* (or index) of $\mathbb{Z}[\pi]$ in the maximal order $\mathcal{O}_K$ of K.

Now, if $\mathcal{E}$ is on the crater, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2 \cong \mathcal{O}_K \otimes \mathbb{Z}_2$; if $\mathcal{E}$ is on the floor, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2 \cong \mathbb{Z}[\pi] \otimes \mathbb{Z}_2$. In between, if $\mathcal{E}$ is d levels down from the crater, then $\mathrm{End}(\mathcal{E}) \otimes \mathbb{Z}_2$ has index 2^d in $\mathcal{O}_K \otimes \mathbb{Z}_2$; that is, the level d gives the 2-valuation of the conductor of $\mathrm{End}(\mathcal{E})$ in $\mathcal{O}_K$. The height h of the volcano is therefore bounded by the 2-valuation of c, which can never be greater than $\log_2(\sqrt{|t^2 - 4p^2|}) \le \log_2(\sqrt{4p^2}) = \log_2 p + 1$.

4.3 SUPERSINGULARITY TESTING WITH ISOGENY VOLCANOES

Sutherland's supersingularity test [22] determines the supersingularity of an elliptic curve $\mathcal{E}/\mathbb{F}_{p^2}$ by exploring the 2-isogeny graph around $\mathcal{E}$ and determining whether it is a volcano or not. If it is, then $\mathcal{E}$ must be ordinary; otherwise, it is supersingular.

We explore the graph using *modular polynomials*. The (classical) modular polynomial of level 2 is

$$\begin{aligned}\Phi_2(J_1, J_2) = {} & J_1^3 + J_2^3 - J_1^2J_2^2 + 1488(J_1^2J_2 + J_1J_2^2) - 162000(J_1^2 + J_2^2) \\ & + 40773375J_1J_2 + 8748000000(J_1 + J_2) - 157464000000000\,.\end{aligned}$$

This polynomial has the property that

$$\text{there exists a 2-isogeny } \mathcal{E}_1 \to \mathcal{E}_2 \iff \Phi_2(j(\mathcal{E}_1), j(\mathcal{E}_2)) = 0\,.$$

Hence, given a vertex $j(\mathcal{E})$ in the 2-isogeny graph, we can compute its neighbours by computing the roots of the cubic polynomial $\Phi_2(j_0, X)$ in $\mathbb{F}_{p^2}$.

Algorithm 4 is Sutherland's algorithm for curves $\mathcal{E}/\mathbb{F}_{p^2}$. We start three non-backtracking walks in the graph from the vertex of $\mathcal{E}$. If the graph is a volcano, then one of the walks must head towards the floor; if we find the floor (that is, a vertex with only one neighbour), then we can declare the curve ordinary. On the other hand, we know the maximal distance to the floor, if it exists; so if no walk finds the floor in less than $\log_2 p + 1$ steps, then we can declare the curve supersingular.

Initialising the three walks requires factoring the cubic $\Phi_2(j(\mathcal{E}), X)$. In every proceeding step, we can avoid backtracking by dividing the evaluated modular polynomial by $X - j'$, where j' is the j-invariant of the previous vertex. Therefore, each subsequent step in a walk requires factoring only a quadratic, which can be done easily via the quadratic formula; the cost of each step is therefore dominated by the computation of a square root in $\mathbb{F}_{p^2}$.

Algorithm 4: Sutherland's supersingularity test for elliptic curves over $\mathbb{F}_{p^2}$.

Input: $j \in \mathbb{F}_{p^2}$ (the j-invariant of an elliptic curve $\mathcal{E}/\mathbb{F}_{p^2}$)
Output: **True** or **False**

Function `IsSupersingular`(j)
- $\Phi \leftarrow$ `ClassicalModularPolynomial(2)` mod p — // $\Phi(X,Y) \in \mathbb{F}_p[X,Y]$
- $f \leftarrow$ `Evaluate`$(\Phi, (j, X))$ — // $f(X) \in \mathbb{F}_{p^2}[X]$, degree 3
- $J \leftarrow$ `Roots`$(f, \mathbb{F}_{p^2})$ — // Roots may be repeated
- **if** $\#J < 3$ **then**
  - **return False**
- $(j_1, j_2, j_3) \leftarrow J$
- $(j'_1, j'_2, j'_3) \leftarrow (j, j, j)$
- **for** $0 \le s \le \lfloor \log_2 p \rfloor$ **do** — // 2-isogeny walk steps
  - **for** $1 \le i \le 3$ **do** — // Three parallel walks
    - $f_i \leftarrow$ `Evaluate`$(\Phi, (j_i, X))$ — // $f_i(X) \in \mathbb{F}_{p^2}[X]$, degree 3
    - $f_i \leftarrow f_i/(X - j'_i)$ — // No backtracking
    - $J_i \leftarrow$ `Roots`$(f_i, \mathbb{F}_{p^2})$ — // Roots may be repeated
    - **if** $\#J_i \neq 2$ **then** — // We have hit the volcano floor
      - **return False**
    - $(j_i, j'_i) \leftarrow$ (`Random`$(J_i), j_i)$ — // Next step
- **return True**

4.4 AN IMPROVED VOLCANO TEST FOR CURVES OVER PRIME FIELDS

Sutherland suggests a speed-up for the case that the given curve is defined over $\mathbb{F}_p$—which is exactly the CSIDH setting. The key fact is that if we consider the subgraph of the supersingular 2-isogeny graph supported on vertices with j-invariants in $\mathbb{F}_p$, then this part looks like a particularly stumpy volcano:[2] that is, it consists of a cycle, with descending trees of height at most 1 (see [11] for a detailed treatment of this graph). When we allow vertices over $\mathbb{F}_{p^2}$, then we obtain the usual complete 3-regular graph. On the other hand, if $\mathcal{E}/\mathbb{F}_p$ is an ordinary curve, then it is in the upper part of the volcano of curves over $\mathbb{F}_{p^2}$, because the ring generated by the p^2-power Frobenius is (obviously) a subring of the ring generated by the p-power Frobenius.

Using these facts, when running Sutherland's algorithm from a vertex in $\mathbb{F}_p$, the "descending" path can quickly be determined: as soon as we encounter a j-invariant in $\mathbb{F}_{p^2}$ (which, in the supersingular case, will happen within two steps) we know that this is the only possibility for a "descending" walk, so we can continue that walk and stop the two others. This simple modification translates into a rough $3\times$ speedup when the input is in $\mathbb{F}_p$.

In fact, we can do even better. Sutherland bounds the length of a maximal descending walk by $\log_2 p + 1$, which is the maximum height of the 2-isogeny volcano containing any ordinary curve over $\mathbb{F}_{p^2}$. But as we noted above,

[2] In fact, this description corresponds to the graph whose vertices correspond to $\mathbb{F}_p$-isomorphism classes of supersingular curves over $\mathbb{F}_p$; using j-invariants corresponds to quotienting this structure by the involution mapping each curve to its quadratic twist, but in practice this makes no difference to the operation or correctness of the algorithm.

when working with curves defined over $\mathbb{F}_p$, we can quickly step into the $\mathbb{F}_{p^2}$-part of the 2-isogeny graph, and then continue there. Lemma 1 shows that the length of a "descending" walk in the $\mathbb{F}_{p^2}$-part of the graph is bounded by $\frac{1}{2}\log_2 p + 1$: that is, essentially half of Sutherland's bound. Combined with the modification above, this yields a $\approx 6\times$ speedup over the general algorithm.

Lemma 1. *Let $\mathcal{E}$ be an ordinary elliptic curve over $\mathbb{F}_p$. The height of the $\mathbb{F}_{p^2}$-part of the 2-isogeny volcano containing $\mathcal{E}$ is at most $\frac{1}{2}\lfloor\log_2 p\rfloor + 1$.*

Proof. Let π_p be the p-power Frobenius endomorphism of $\mathcal{E}$, and t_p its trace. The height of the $\mathbb{F}_{p^2}$-part of the 2-isogeny volcano containing $\mathcal{E}$ is the 2-valuation of the conductor of $\mathbb{Z}[\pi_p^2]$ in $\mathbb{Z}[\pi_p]$. The discriminants of $\mathbb{Z}[\pi_p]$ and $\mathbb{Z}[\pi_p^2]$ are $\Delta_1 = t_p^2 - 4p$ and $\Delta_2 = (t_p^2 - 2p)^2 - 4p^2 = t_p^2\Delta_1$, respectively, so the relative conductor is $|t_p|$. But $|t_p| < 2\sqrt{p}$ by Hasse's theorem; hence, the 2-valuation of $|t_p|$ is bounded above by $\log_2(2\sqrt{p}) = \frac{1}{2}\log_2 p + 1$. □

Our second improvement is to streamline the isogeny step computations. Rather than computing roots of evaluations of the classical modular polynomial Φ_2, we can use Lemma 2 to compute explicit 2-isogenies.

Lemma 2. *Let $\mathcal{E}/\mathbb{F}_{p^2}$ be an elliptic curve defined by an equation $y^2 = x(x^2 + a_2x + a_4)$. Set $D := a_2^2 - 4a_4$, and choose a square root $\delta := \sqrt{D}$ in $\overline{\mathbb{F}}_p$. Then $\alpha := -(a_2 - \delta)/2$ is a root of $x^2 + a_2x + a_4$, there is a quotient isogeny*

$$\varphi : \mathcal{E} \longrightarrow \mathcal{E}/\langle(\alpha, 0)\rangle \cong \mathcal{E}' : y^2 = x(x^2 + a_2'x + a_4') \qquad \text{where} \qquad \begin{cases} a_2' = a_2 - 3\delta\,, \\ a_4' = a_2(a_2 + \delta)/2 - a_4 \end{cases} \tag{4}$$

defined over $\mathbb{F}_{p^2}(\delta)$, and the kernel of its dual isogeny $\widehat{\varphi}$ is generated by $(0, 0)$.

Proof. Follows on composing the normalized 2-isogeny given by Vélu's formulæ (see [23] or [15, §2.4]) with the map $(x, y) \mapsto (x + \delta, y)$. □

We can now define our variant of Sutherland's algorithm. The subroutine `SqrtFp2`, given α in $\mathbb{F}_{p^2}$ (or $\mathbb{F}_p$), returns a square root of α in $\mathbb{F}_{p^2}$ if one exists, or $\perp$ if not. When $p \equiv 3 \pmod 4$—as in CSIDH, and many other isogeny-based cryptosystems—we can use the efficient arithmetic for $\mathbb{F}_{p^2}$ from Appendix A.

Algorithm 5: Modified Sutherland supersingularity test for Montgomery curves over $\mathbb{F}_p$.

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**

Function `IsSupersingular`(A)
  if $p \not\equiv 3 \pmod 4$ **then** // No supersingular curve over $\mathbb{F}_p$ has a Montgomery model
    return False
  if $A = 0$ **then** // j-invariant 1728: always supersingular when $p \equiv 3 \pmod 4$
    return True
  $(a_2, D) \leftarrow (A, A^2 - 4)$
  $\delta \leftarrow$ `SqrtFp2`(D) // D is in $\mathbb{F}_p$, so δ cannot be $\perp$
  if $\delta \in \mathbb{F}_p$ **then** // D is a square in $\mathbb{F}_p$
    return False
  for $0 \le i \le \frac{1}{2}\lfloor\log_2 p\rfloor + 1$ **do**
    $(a_2, D) \leftarrow (a_2 - 3\delta, 8(D - \delta \cdot a_2))$
    $\delta \leftarrow$ `SqrtFp2`(D)
    if $\delta = \perp$ **then** // D is not a square in $\mathbb{F}_{p^2}$
      return False
  return True

Proposition 1. *Given a Montgomery coefficient A in $\mathbb{F}_p$ corresponding to an elliptic curve $\mathcal{E}_A : y^2 = x(x^2+Ax+1)$, Algorithm 5 returns* **True** *if and only if $\mathcal{E}$ is supersingular. It requires $O(\log^2 p)$ $\mathbb{F}_p$-operations (in the worst case, which is when $\mathcal{E}$ is supersingular) and $O(1)$ $\mathbb{F}_p$-elements worth of space.*

Proof. First, recall that every Montgomery model has cardinality divisible by 4, and therefore a Montgomery model can only be supersingular if $p \equiv 3 \pmod 4$ (since otherwise $4 \nmid (p + 1)$). We can immediately dispose of the special case $A = 0$: this curve has j-invariant 0, and is always supersingular when $p \equiv 3 \pmod 4$.

For the general case where $A \neq 0$, let α and $1/\alpha$ be the roots of $x^2 + Ax + 1$ in $\mathbb{F}_{p^2}$. Having a Montgomery model implies that $\mathcal{E}_A$ is on the floor of the $\mathbb{F}_p$-volcano [4, Prop. 8]; there is only one $\mathbb{F}_p$-rational 2-isogeny (its kernel is $(0,0)$), and this isogeny must be ascending. In particular, if $\mathcal{E}_A$ is supersingular then $x^2 + Ax + 1$ can have no roots in $\mathbb{F}_p$; hence, if $A^2 - 4$ is a square in $\mathbb{F}_p$, then $\mathcal{E}_A$ is ordinary and we can return **False** at Line 9.

Otherwise, Lemma 2 shows that we can continue with a non-backtracking walk into the $\mathbb{F}_{p^2}$-part of the 2-isogeny graph by iterating the mapping $(a_2, D = a_2^2 - 4a_4) \mapsto (a_2', D' = (a_2')^2 - 4a_4')$ defined by (4), starting with $(a_2, D) = (A, A^2 - 4)$. The $\mathbb{F}_{p^2}$-rationality of each step depends on whether or not D is a square in $\mathbb{F}_{p^2}$; the choice between each pair of descending isogenies is made arbitrarily, according to which of the two square roots is returned by `SqrtFp2`. Algorithm 5 repeats this process until we either hit the floor of $\mathbb{F}_{p^2}$-rationality and conclude that $\mathcal{E}_A$ is ordinary, or go beyond the maximal path length specified by Lemma 1 and conclude that $\mathcal{E}_A$ is supersingular.

Algorithm 5 is therefore correct. It requires (in the worst case) one square root in $\mathbb{F}_p$ and $\frac{1}{2}\log_2 p + 2$ iterations of a loop consisting of a square root in $\mathbb{F}_{p^2}$ plus a handful of $\mathbb{F}_{p^2}$-operations. Each square root costs $O(\log p)$ $\mathbb{F}_p$-operations using the Tonelli–Shanks algorithm, so we have a total complexity of $O(\log^2 p)$ $\mathbb{F}_p$-operations. □

5 DIVISION POLYNOMIALS AND DOLISKANI'S TEST

Doliskani gives an interesting probabilistic supersingularity test in [12], based on Polynomial Identity Testing and properties of division polynomials. To the best of our knowledge, this algorithm has not yet been used in practice. We will provide some important algorithmic improvements that will ultimately make this the most efficient of our supersingularity tests.

5.1 DIVISION POLYNOMIALS AND SUPERSINGULARITY

Recall that that for $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\} .$$

For $\mathcal{E} : y^2 = x(x^2 + Ax + 1)$, the first few division polynomials are

$$\psi_{\mathcal{E},0} = 0 , \qquad \psi_{\mathcal{E},1} = 1 , \qquad \psi_{\mathcal{E},2} = 2y ,$$

then (for Montgomery models)

$$\psi_{\mathcal{E},3} = 3x^4 + 4Ax^3 + 6x^2 - 1 , \qquad \psi_{\mathcal{E},4} = 4y(x^2 - 1)(x^4 + 2Ax^3 + 6x^2 + 2Ax + 1) .$$

The higher-degree polynomials satisfy the standard recurrences

$$\psi_{\mathcal{E},2m} = (\psi_{\mathcal{E},m-1}^2 \psi_{\mathcal{E},m} \psi_{\mathcal{E},m+2} - \psi_{\mathcal{E},m-2} \psi_{\mathcal{E},m} \psi_{\mathcal{E},m+1}^2)/\psi_{\mathcal{E},2} \qquad \text{for } m \geq 3 , \tag{5}$$

$$\psi_{\mathcal{E},2m+1} = \psi_{\mathcal{E},m}^3 \psi_{\mathcal{E},m+2} - \psi_{\mathcal{E},m-1} \psi_{\mathcal{E},m+1}^3 \qquad \text{for } m \geq 2 . \tag{6}$$

We see that $\psi_{\mathcal{E},m}^2$ is a polynomial in $\mathbb{F}_p[A][x]$ for any $m > 0$, and indeed $\psi_{\mathcal{E},m}$ is in $\mathbb{F}_p[A][x]$ for odd m.

Lemma 3. *The p-th division polynomial satisfies $\psi_{\mathcal{E},p}(x) = \widetilde{\psi}_{\mathcal{E},p}(x)^p$ where $\widetilde{\psi}_{\mathcal{E},p}(x)$ is either a polynomial of degree $(p-1)/2$ if $\mathcal{E}$ is ordinary, or ± 1 if $\mathcal{E}$ is supersingular. In particular,*

$$\psi_{\mathcal{E},p}^2(x) = 1 \iff \mathcal{E} \text{ is supersingular.}$$

Proof. See Theorem 3.1 and Propositions 3.3 and 3.4 of [14] (where the sign in the ± 1 for the supersingular case is made completely explicit, though we only need the fact that $\psi_{\mathcal{E},p}^2(x) = 1$ here). □

Example 1. *Let $\mathcal{E} : y^2 = x(x^2 + Ax + 1)$ be the generic Montgomery model over $\mathbb{F}_7(A)$. We find*

$$\psi_{\mathcal{E},7}(x) = \widetilde{\psi}_{\mathcal{E},7}(x)^7 \qquad \textit{where} \qquad \widetilde{\psi}_{\mathcal{E},7}(x) = A(A^2-1)\big(x^3 - A(A^2-2)x^2 + 2(A^2+1)x\big) - 1 .$$

In particular, if $A \in \{-1, 0, 1\}$ then $\widetilde{\psi}_{\mathcal{E},7} = -1$ and $\mathcal{E}$ is supersingular; otherwise, $\deg \widetilde{\psi}_{\mathcal{E},7} = 3$ *and $\mathcal{E}$ is ordinary.*

Doliskani's test applies basic Polynomial Identity Testing to check the supersingularity criterion of Lemma 3. The algorithm presented in [12] samples a uniformly random u in $\mathbb{F}_{p^2}$, and computes $\widetilde{\psi}_{\mathcal{E},p}(u)^2$ as $\psi_{\mathcal{E},p}(u)^2$. If this yields $\psi_{\mathcal{E},p}(u) = 1$, then the Schwartz–Zippel lemma (see e.g. [19]) tells us that $\psi_{\mathcal{E},p}^2(x) = 1$, and $\mathcal{E}$ is supersingular, with probability $1 - (\deg \widetilde{\psi}_{\mathcal{E},p})/\#\mathbb{F}_{p^2} = 1 - ((p-1)/2)/p^2 \approx 1 - 1/2p$.

5.2 EFFICIENT EVALUATION OF DIVISION POLYNOMIALS

To evaluate $\psi_{\mathcal{E},p}(u)$, Doliskani proposes an algorithm based on the standard recurrences of (5) and (6), which resembles a vectorial addition chain where the vector tracks the values of nine consecutively-indexed division polynomials. This algorithm seems to be folklore, but Cheng gives a detailed analysis in [6].

We can significantly improve the efficiency of division polynomial evaluation by using the link between division polynomials and scalar multiplication. Recall that

$$[m](x, y) = \left(\frac{\phi_{\mathcal{E},m}(x)}{\psi_{\mathcal{E},m}(x)^2}, \frac{\omega_{\mathcal{E},m}(x, y)}{\psi_{\mathcal{E},m}(x)^3}\right) \quad \text{where} \quad \phi_{\mathcal{E},m}(x) := x\psi_{\mathcal{E},m}(x)^2 - \psi_{\mathcal{E},m+1}(x)\psi_{\mathcal{E},m-1}(x) \tag{7}$$

and $\omega_{\mathcal{E},m}(x, y) = (4y)^{-1}\left(\psi_{\mathcal{E},m+2}(x)\psi^2_{\mathcal{E},m-1}(x) - \psi_{\mathcal{E},m-2}(x)\psi^2_{\mathcal{E},m+1}(x)\right)$ (though we will not need ω_m in what follows). Hence, if $(u : v : 1)$ is a point on $\mathcal{E}_A$ and we use the Montgomery ladder to compute the (X, Z)-coordinates of $[p](u : v : 1)$, then (7) tells us that the X and Z coordinates are $\phi_{\mathcal{E},p}(u)$ and $\psi^2_{\mathcal{E},p}(u)$, respectively, *up to a common projective factor*—which we can predict and remove. Proposition 2 makes this explicit for general m.

Proposition 2. *On input A, m, and* $(x, 1)$*, Algorithm 1 (the Montgomery ladder) returns*

$$(X_m, Z_m) = (\phi_{\mathcal{E},m}(x) \cdot f_m(x), \psi^2_{\mathcal{E},m}(x) \cdot f_m(x)) \quad \textit{where} \quad f_m(x) := (4x)^{m(2^{\mathrm{len}(m)} - m)} . \tag{8}$$

Proof. Let P be a point on $\mathcal{E}_A$ with x-coordinate $x \neq 0$. We saw in §2 that $\mathtt{Ladder}(A, m, (x, 1))$ returns $\mathsf{R}_0 = (X_m, Z_m)$ and $\mathsf{R}_1 = (X_{m+1}, Z_{m+1})$ such that $X_m/Z_m = x([m]P)$ and $X_{m+1}/Z_{m+1} = x([m + 1]P)$, so (7) implies

$$\begin{aligned}
\mathsf{R}_0 &= (X_m, Z_m) = \left(\phi_{\mathcal{E},m}(x) \cdot f_m(x), \psi^2_{\mathcal{E},m}(x) \cdot f_m(x)\right) , \\
\mathsf{R}_1 &= (X_{m+1}, Z_{m+1}) = \left(\phi_{\mathcal{E},m+1}(x) \cdot g_m(x), \psi^2_{\mathcal{E},m+1}(x) \cdot g_m(x)\right)
\end{aligned}$$

for some projective factors $f_m(x)$ and $g_m(x)$. We claim that

$$f_m(x) = (4x)^{F_m} \quad \text{and} \quad g_m(x) = (4x)^{G_m} \quad \text{where} \quad \begin{cases} F_m := m(2^{\mathrm{len}(m)} - m) , \\ G_m := (m + 1)(2^{\mathrm{len}(m)} - (m + 1)) . \end{cases} \tag{9}$$

We proceed by induction. First, the base case: on input $(A, 1, (x, 1))$, Algorithm 1 outputs

$$\begin{aligned}
\mathsf{R}_0 &= (4x^2, 4x) = (\phi_{\mathcal{E}_A,1}(x) \cdot 4x, \psi_{\mathcal{E},1}(x) \cdot 4x) , \\
\mathsf{R}_1 &= (x^4 - 2x^2 + 1, 4x^3 + 4Ax^2 + 4x) = (\phi_{\mathcal{E},2}(x) \cdot 1, \psi^2_{\mathcal{E},2}(x) \cdot 1) ,
\end{aligned}$$

which is exactly (9) with $m = 1$. For the inductive step: looking at the differential addition and doubling formulæ in (2) and (3), respectively, we see that the projective factors are defined by mutually recursive relationships:

$$f_{2m}(x) = f^4_m(x) , \qquad f_{2m+1}(x) = g_{2m}(x) = 4xf^2_m(x)g^2_m(x) , \qquad g_{2m+1}(x) = g^4_m(x) . \tag{10}$$

Substituting (9) into (10), it suffices to show that

$$F_{2k} = 4F_k , \quad G_{2k} = 2F_k + 2G_k + 1 = F_{2k+1} , \quad \text{and} \quad G_{2k+1} = 4G_k .$$

Noting that $\mathrm{len}(2k) = \mathrm{len}(k) + 1$, we find that

$$F_{2k} = 2k(2^{\mathrm{len}(2k)} - 2k) = 2k(2 \cdot 2^{\mathrm{len}(k)} - 2k) = 4k(2^{\mathrm{len}(k)} - k) = 4F_k$$

and similarly

$$G_{2k+1} = (2k + 2)(2^{\mathrm{len}(2k)} - (2k + 2)) = 2(k + 1)(2 \cdot 2^{\mathrm{len}(k)} - 2(k + 1)) = 4(k + 1)(2^{\mathrm{len}(k)} - (k + 1)) = 4G_k .$$

Since $F_{2k+1} = G_{2k}$ by definition, it remains to show that $G_{2k} = 2F_k + 2G_k + 1$, and indeed

$$\begin{aligned}
G_{2k} &= (2k + 1)(2^{\mathrm{len}(2k)} - (2k + 1)) \\
&= (2k + 1)(2 \cdot 2^{\mathrm{len}(k)} - 2k) - 2k - 1 \\
&= 2k(2^{\mathrm{len}(k)} - k) + 2(k + 1)(2^{\mathrm{len}(k)} - (k + 1)) + 1 \\
&= 2F_k + 2G_k + 1 ,
\end{aligned}$$

as required. □

When $m = p$, we can use the fact that $\alpha^p = \alpha$ for α in $\mathbb{F}_p$ to replace (8) with the simpler formula

$$f_p(u) = (4\bar{u})^{2^{\mathrm{len}(p)}}/(4u) \qquad \text{where} \qquad \bar{u} = u^p\,. \tag{11}$$

If $p \equiv 3 \pmod 4$ and we realise $\mathbb{F}_{p^2}$ as $\mathbb{F}_p(i)$, then $\overline{a+bi} = a - bi$, so computing $\bar{u} = u^p$ is almost free.

Computing $\psi^2_{\mathcal{E},m}(u)$ using Proposition 2 requires $\approx 19\,\mathrm{len}(m)$ arithmetic operations (each bit of m requires one 18-operation ladder step, plus one squaring in the final exponentiation). This compares very favourably with Cheng's algorithm, which requires $\approx 72\,\mathrm{len}(m)$ operations to compute $\psi_{\mathcal{E},m}(u)$, and more storage (see [6, §4]).

5.3 DOLISKANI'S TEST REVISITED

Doliskani's original test from [12] samples a value u uniformly at random from $\mathbb{F}_{p^2}$, and returns **True** if and only if $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ and $\psi^2_{\mathcal{E},p}(u) = 1$. Algorithm 6 combines Doliskani's test with our efficient algorithm for evaluating division polynomials. Remarkably, we can replace the computation of $\psi^2_{\mathcal{E},p}(u)$, $\psi_{\mathcal{E},p+1}(u)$, and $\psi_{\mathcal{E},p-1}(u)$ with a single application of the Montgomery ladder over $\mathbb{F}_{p^2}$.

Algorithm 6: Doliskani's PIT supersingularity test with fast ladder-based division polynomial evaluation

Input: $A \in \mathbb{F}_p$
Output: **True** or **False**
1 **Function** `IsSupersingular`(A)
2 $\quad u \leftarrow$ `Random`$(\mathbb{F}_{p^2} \setminus \{0\})$
3 $\quad (X_p, Z_p) \leftarrow$ `Ladder`$(A, p, (u, 1))$
4 $\quad$ **if** $X_p \neq u \cdot Z_p$ **then** $\qquad$ `// Order of` $(u : * : 1)$ `does not divide` $p \pm 1$
5 $\qquad$ **return False**
6 $\quad$ **if** $4u \cdot Z_p = (4\bar{u})^{2^{\mathrm{len}(p)}}$ **then** $\qquad$ `// Polynomial Identity Testing:` $Z_p = \psi^2_{\mathcal{E},p}(u) f_p(u)$
7 $\qquad$ **return True**
8 $\quad$ **else**
9 $\qquad$ **return False**

Proposition 3. *Let A in $\mathbb{F}_p$ be the Montgomery coefficient of an elliptic curve $\mathcal{E}_A : y^2 = x(x^2 + Ax + 1)$.*

- *If $\mathcal{E}_A$ is supersingular, then Algorithm 6 returns* **True***.*
- *If $\mathcal{E}_A$ is ordinary, then Algorithm 6 returns* **True** *with probability $1/(2p+2)$, and* **False** *otherwise.*

Algorithm 6 requires one random element of $\mathbb{F}_{p^2}$, $O(\log p)$ $\mathbb{F}_p$-operations and $O(1)$ $\mathbb{F}_p$-elements worth of space.

Proof. As mentioned above, Doliskani's test samples u uniformly at random from $\mathbb{F}_{p^2}$, and returns **True** if and only if $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ and $\psi^2_{\mathcal{E},p}(u) = 1$. We make a small change here, requiring $u \neq 0$ to ensure that the Montgomery ladder returns valid output on input $(u, 1)$.

Let $v \in \mathbb{F}_{p^4}$ be such that $P = (u : v : 1)$ is a point in $\mathcal{E}_A(\mathbb{F}_{p^4})$. If $\mathcal{E}_A/\mathbb{F}_p$ is supersingular, then $\mathcal{E}_A(\mathbb{F}_{p^4}) \cong (\mathbb{Z}/(p^2-1)\mathbb{Z})^2$, so the condition $\psi_{\mathcal{E},p+1}(u)\psi_{\mathcal{E},p-1}(u) = 0$ amounts to checking that P has order dividing $p^2 - 1$. But we can equivalently check that $[p]P = \pm P$, which is what Line 4 does.

It remains to check that $\psi^2_{\mathcal{E},p}(u) = 1$, which holds if and only if $\widetilde{\psi}^2_{\mathcal{E},p}(\bar{u}) = 1$. The probability that $\widetilde{\psi}^2_{\mathcal{E},p}(\bar{u}) = 1$ while $\widetilde{\psi}_{\mathcal{E},p}(x) \neq \pm 1$ (that is, while $\mathcal{E}_A$ is ordinary) is $\deg \widetilde{\psi}_{\mathcal{E},p}/\#(\mathbb{F}_{p^2} \setminus \{0\}) = ((p-1)/2)/(p^2-1) = 1/(2p+2)$ by the Schwartz–Zippel lemma. We compute $\psi^2_{\mathcal{E},p}(u)$ using Proposition 2 and (11). This requires one application of the Montgomery ladder over $\mathbb{F}_{p^2}$ with a scalar of length $\mathrm{len}(p) = \lceil \log_2 p \rceil$, followed by $\mathrm{len}(p)$ squarings in $\mathbb{F}_p$; both cost $O(\log p)$ $\mathbb{F}_p$-operations, storing only $O(1)$ $\mathbb{F}_p$-elements. □

Remark 2. *The projective factor f_p of Proposition 2 depends on the ladder algorithm as presented in Algorithm 1 and the differential addition and doubling formulæ in §2. If the ladder is initialised differently, or replaced with an optimal differential addition chain, or if alternative formulæ are used, then f_p should be redefined accordingly.*

Remark 3. *Doliskani's test is not limited to curves over $\mathbb{F}_p$: indeed, it was designed for curves over $\mathbb{F}_{p^2}$. Our algorithm and implementation makes little use of the fact that the input is defined over $\mathbb{F}_p$ (this slightly simplifies multiplication by curve constants, but this makes only a slight difference to the runtime): it should have essentially the same cost for inputs in $\mathbb{F}_{p^2}$, and the same probability of correctness.*

6 OTHER ELLIPTIC CURVE MODELS

Looking beyond the Montgomery models used in CSIDH, the supersingularity tests described here can all be easily adapted to other elliptic curve models over $\mathbb{F}_p$, such as short Weierstrass curves or (twisted) Edwards curves. Such adaptations may be required for key validation in other isogeny-based systems such as CSURF [3], but they might also be useful for purely number-theoretic calculations.

Elementary tests The random-point tests of Algorithms 2 and 3 work essentially unchanged for other curve models, once the Montgomery ladder has been replaced with a suitable high-speed scalar multiplication routine, and tests like $Z \neq 0$ replaced with comparisons against the neutral element of the group.

Sutherland's test Sutherland's original supersingularity test over $\mathbb{F}_{p^2}$ (Algorithm 4) works for any curve model. Our modified version over $\mathbb{F}_p$ (Algorithm 5) requires extensive changes for alternative curve models, though the underlying algorithm is very similar:

- Lines 2 and 4 are Montgomery-specific, and must be replaced (or omitted) according to the curve model.
- Lines 6 through 14 depend on the Montgomery-specific formulæ from Lemma 2, which should be replaced with an appropriate analogue for the given curve model.
- Lines 7 through 9 use the fact that supersingular Montgomery curves are necessarily on the floor of the $\mathbb{F}_p$-volcano for $\ell = 2$. This is not true for more general curves: for example, the curves in CSURF [3] are supposed to be on the surface. These lines should be replaced with an examination of the neighbourhood of the starting vertex to find a neighbour on the floor, if it exists, before proceeding as usual.

Doliskani's test Our modification of Doliskani's test (Algorithm 6) extends to other projective curve models. The Montgomery ladder must be replaced with a suitable high-speed scalar multiplication, and as we noted in Remark 2, the projective factor must be adapted to fit that scalar multiplication algorithm and the curve arithmetic that it uses.

7 CONCLUSION

This paper improves two supersingularity tests, originally due to Sutherland and Doliskani, and compares them with the state-of-the-art in the context of CSIDH public-key validation.

Our modification of Sutherland's algorithm specialized to prime fields reduces the running time and space; while it does not change the asymptotic complexity, it does improve the constant hidden by the big-O. It performs relatively slowly for valid keys (though it is still quite practical for CSIDH-512 parameters), but it rejects invalid keys much faster than the other tests: it is therefore probably more useful in computational number theory (where we mostly encounter ordinary input) than in CSIDH key validation (where we mostly expect supersingular input from honest parties). In any case, it has the advantages of low memory and definitive proof of supersingularity.

Our modification of Doliskani's algorithm shows a more significant improvement due to our new method for evaluating squared division polynomials. This algorithm achieved a substantial speedup over the alternatives for valid CSIDH-512 keys, while remaining competitive on invalid keys. It uses less memory than the currently-used product-tree algorithm, and is far simpler to implement correctly. We therefore suggest that this algorithm is a better choice for key validation in CSIDH and similar isogeny-based protocols.

Our benchmarks all used the CSIDH-512 parameter set, with a 512-bit prime p. We expect that our algorithms and results will be relevant for larger primes, but more work needs to be done to optimize these cases.

REFERENCES

[1] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. "CTIDH: faster constant-time CSIDH". In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2021.4 (2021), pp. 351–387. DOI: 10.46586/tches.v2021.i4.351-387.

[2] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. "CSI-FiSh: Efficient Isogeny Based Signatures Through Class Group Computations". In: *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I.* Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11921. Lecture Notes in Computer Science. Springer, 2019, pp. 227–247. DOI: 10.1007/978-3-030-34578-5_9.

[3] Wouter Castryck and Thomas Decru. "CSIDH on the Surface". In: *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020, Paris, France, April 15-17, 2020, Proceedings*. Ed. by Jintai Ding and Jean-Pierre Tillich. Vol. 12100. Lecture Notes in Computer Science. Springer, 2020, pp. 111–129. DOI: 10.1007/978-3-030-44223-1_7.

[4] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. "CSIDH: An Efficient Post-Quantum Commutative Group Action". In: *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*. Ed. by Thomas Peyrin and Steven D. Galbraith. Vol. 11274. Lecture Notes in Computer Science. Springer, 2018, pp. 395–427. DOI: 10.1007/978-3-030-03332-3_15.

[5] Jorge Chávez-Saab, Jesús-Javier Chi-Domınguez, Samuel Jaques, and Francisco Rodrıguez-Henrıquez. "The SQALE of CSIDH: Square-root Vélu quantum-resistant isogeny action with low exponents". In: *Journal of Cryptographic Engineering* (2021). DOI: 10.1007/s13389-021-00271-w.

[6] Qi Cheng. "Straight-line programs and torsion points on elliptic curves". In: *Computational Complexity* 12 (2003), pp. 150–161. DOI: 10.1007/s00037-003-0180-0.

[7] Craig Costello and Benjamin Smith. "Montgomery curves and their arithmetic: The case of large characteristic fields". In: *Journal of Cryptographic Engineering* 8.3 (2018), pp. 227–240. DOI: 10.1007/s13389-017-0157-6.

[8] Luca De Feo and Steven D. Galbraith. "SeaSign: Compact Isogeny Signatures from Class Group Actions". In: *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. Lecture Notes in Computer Science. Springer, 2019, pp. 759–789. DOI: 10.1007/978-3-030-17659-4_26.

[9] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. "SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies". In: *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12491. Lecture Notes in Computer Science. Springer, 2020, pp. 64–93. DOI: 10.1007/978-3-030-64837-4_3.

[10] Luca De Feo and Michael Meyer. "Threshold Schemes from Isogeny Assumptions". In: *Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12111. Lecture Notes in Computer Science. Springer, 2020, pp. 187–212. DOI: 10.1007/978-3-030-45388-6_7.

[11] Christina Delfs and Steven D. Galbraith. "Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$". In: *Des. Codes Cryptogr.* 78.2 (2016), pp. 425–440. DOI: 10.1007/s10623-014-0010-1.

[12] Javad Doliskani. "On division polynomial PIT and supersingularity". In: *Appl. Algebra Eng. Commun. Comput.* 29.5 (2018), pp. 393–407. DOI: 10.1007/s00200-018-0349-z.

[13] Mireille Fouquet and François Morain. "Isogeny Volcanoes and the SEA Algorithm". In: *Algorithmic Number Theory. ANTS 2002*. Ed. by Claus Fieker and David R. Kohel. Vol. 2369. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 276–291. ISBN: 978-3-540-45455-7. DOI: 10.1007/3-540-45455-1_23.

[14] Josep González. "On the p-th division polynomial". In: *Journal of Number Theory* 233 (2022), pp. 285–300. DOI: 10.1016/j.jnt.2021.06.011.

[15] David R. Kohel. "Endomorphism rings of elliptic curves over finite fields". http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf. PhD thesis. University of California at Berkeley, 1996.

[16] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. "Compact, Efficient and UC-Secure Isogeny-Based Oblivious Transfer". In: *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12696. Lecture Notes in Computer Science. Springer, 2021, pp. 213–241. DOI: 10.1007/978-3-030-77870-5_8.

[17] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. CRC press, 2018.

[18] Peter L. Montgomery. "Speeding the Pollard and elliptic curve methods of factorization". In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. DOI: 10.1090/S0025-5718-1987-0866113-7.

[19] Jacob T. Schwartz. "Fast probabilistic algorithms for verification of polynomial identities". In: *J. ACM* 27.4 (1980), pp. 701–717. DOI: 10.1145/322217.322225.

[20] Michael Scott. "A note on the calculation of some functions in finite fields: Tricks of the trade". IACR ePrint 2020/1497, https://ia.cr/2020/1497. 2020.

[21] Joseph H. Silverman. *The arithmetic of elliptic curves*. 2nd edition. New York, NY: Springer-Verlag, 2009. ISBN: 0387094938.

[22] Andrew V. Sutherland. "Identifying supersingular elliptic curves". In: *LMS Journal of Computation and Mathematics* 15 (2012), pp. 317–325. DOI: 10.1112/S1461157012001106.

[23] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences, Série A* 273 (July 1971), pp. 238–241.

A FINITE FIELD ARITHMETIC

Square roots in $\mathbb{F}_p$. In our case we have $p \equiv 3 \mod 4$, so the classic Tonelli–Shanks algorithm (see e.g. [17, §3.51]) reduces to computing $t = a^{(p+1)/4}$ for $a \in \mathbb{F}_p$ (which can be done using a precomputed optimal chain of squares and multiplications). If $ta^2 = a$ then t is a square root of a; otherwise, a is not a square in $\mathbb{F}_p$.

Representing $\mathbb{F}_{p^2}$. Since $p \equiv 3 \pmod 4$, we can realise $\mathbb{F}_{p^2}$ as $\mathbb{F}_p(i)$, where $i^2 = -1$. Elements x of $\mathbb{F}_{p^2}$ are encoded as the pair of elements (x_r, x_i) of $\mathbb{F}_p$ (the "real" and "imaginary" parts) such that $x = x_r + x_i \cdot i$.

Addition. Given two elements $a, b \in \mathbb{F}_{p^2}$, we can compute $c = a + b$ at the cost of two additions in $\mathbb{F}_p$ using

$$c_r = a_r + b_r \mod p\,, \qquad c_i = a_i + b_i \mod p\,.$$

Subtraction. Similarly, we can compute $c = a - b$ at the cost of two subtractions in $\mathbb{F}_p$ using

$$c_r = a_r - b_r \mod p\,, \qquad c_i = a_i - b_i \mod p\,.$$

Multiplication. We can compute $c = a \cdot b$ for 4 multiplications, 1 addition, and 1 subtraction in $\mathbb{F}_p$ using

$$c_r = a_r \cdot b_r - a_i \cdot b_i \mod p\,, \qquad c_i = a_i \cdot b_r + a_r \cdot b_i \mod p\,.$$

A Karatsuba-style method computes c with 3 multiplications, 3 additions, and 2 subtractions: first we compute

$$t_0 := a_r \cdot b_r\,, \quad t_1 := a_i \cdot b_i\,, \quad t_2 := a_r + a_i\,, \quad t_3 := b_r + b_i\,, \quad t_4 := t_2 \cdot t_3\,, \quad t_5 := t_0 + t_1\,,$$

and then

$$c_r = t_0 - t_1 \mod p\,, \qquad c_i = t_4 - t_5 \mod p\,.$$

Square roots in $\mathbb{F}_{p^2}$. Algorithm 7 computes square roots in $\mathbb{F}_{p^2}$ following the method of [20] with some minor corrections. The exponentiations in Lines 4 and 11 are done using an optimal precomputed addition chain.

Algorithm 7: Square root in $\mathbb{F}_{p^2} = \mathbb{F}_p(i)$ where $p \equiv 3 \pmod 4$ and $i = \sqrt{-1}$, as in [20].

Input: x in $\mathbb{F}_p(i)$
Output: r in $\mathbb{F}_p(i)$ such that $r^2 = x$, if it exists; otherwise $\perp$

Function `SqrtFp2`(x)
  $a + bi \leftarrow x$ // $a, b \in \mathbb{F}_p$
  $\delta \leftarrow a^2 + b^2$ // δ = norm of x
  $\lambda \leftarrow \delta^{(p-3)/4}$ // λ = progenitor of δ
  $\rho \leftarrow \delta \cdot \lambda$ // ρ is candidate square root of δ
  if $\rho^2 \neq \delta$ **then** // δ not square in $\mathbb{F}_p \implies x$ not square in $\mathbb{F}_{p^2}$
    return $\perp$
  $\gamma \leftarrow (a + \rho)/2$
  if $\gamma = 0$ **then** // Can happen when $b = 0$ and $\rho = -a$
    $\gamma = -\rho$ // Now $\gamma = (a - \rho)/2$
  $\mu \leftarrow \gamma^{(p-3)/4}$ // μ = progenitor of γ
  $\sigma \leftarrow \gamma \cdot \mu$ // σ = candidate square root of γ
  $\gamma^{-1} \leftarrow \sigma \cdot \mu^3$ // True inverse of γ
  $\tau \leftarrow \sigma \cdot \gamma^{-1}$ // τ = candidate square root of γ^{-1}
  $\omega \leftarrow (b/2) \cdot \tau$
  if $\sigma^2 = \gamma$ **then**
    return $\sigma + \omega i$
  else // $-\sigma = \sqrt{-\gamma}$ and $\tau = \sqrt{-\gamma^{-1}}$
    return $\omega - \sigma i$