Faster algorithms for isogeny computations over extensions of finite fields

Shiping Cai[1], Mingjie Chen[2][0009−0003−8274−0685], and Christophe Petit[3,4][0000−0003−3482−6743]

[1] School of Mathematics, Sun Yat-sen University, China shiping.cai@ulb.be
[2] COSIC, KU Leuven, Belgium mjchennn555@gmail.com,
[3] University of Birmingham, UK
[4] Université libre de Bruxelles, Belgium christophe.f.petit@gmail.com

Abstract. Any isogeny between two supersingular elliptic curves can be defined over $\mathbb{F}_{p^2}$, however, this does not imply that computing such isogenies can be done with field operations in $\mathbb{F}_{p^2}$. In fact, the kernel generators of such isogenies are defined over extension fields of $\mathbb{F}_{p^2}$, generically with extension degree linear to the isogeny degree. Most algorithms related to isogeny computations are only efficient when the extension degree is small. This leads to efficient algorithms used in isogeny-based cryptographic constructions, but also limits their parameter choices at the same time. In this paper, we consider three computational subroutines regarding isogenies, focusing on cases with large extension degrees: computing a basis of ℓ-torsion points, computing the kernel polynomial of an isogeny given a kernel generator, and computing the kernel generator of an isogeny given the corresponding quaternion ideal under the Deuring correspondence. We then apply our algorithms to the constructive Deuring correspondence algorithm from [EPSV23] in the case of a generic prime characteristic, achieving around 30% speedup over [EPSV23].

1 Introduction

Isogeny-based cryptography is a key candidate for post-quantum cryptography, developed in response to potential quantum threats to current cryptographic systems. Protocols based on isogenies distinguish themselves from other post-quantum cryptography candidates by their compactness. As suggested by its name, the central objects in isogeny-based cryptography are isogenies between supersingular elliptic curves.

As a well-studied topic in both algorithmic number theory and elliptic curve cryptography, plenty of algorithms exist regarding elliptic curves and isogenies computations such as the classic Vélu's formula [V\'71] and square-root Vélu [BDFLS20], which are both algorithms for computing an isogeny (meaning computing its codomain and evaluation on points) given the domain curve and the kernel. However, the swift advancement in isogeny-based cryptography has led to emerging scenarios that demand more efficient algorithms. For instance, the ideal to isogeny translation is the computational bottleneck for [DFKL+20,DFLLW23,GPS17] but received attention only in recent few years.

The current lack of efficient algorithms is particularly evident in computations over extension fields of $\mathbb{F}_{p^2}$. While it is possible to work exclusively within $\mathbb{F}_{p^2}$ by judiciously selecting parameters for protocol design, this approach can be restrictive, as seen in the case of [DFKL$^+$20], potentially necessitating the use of a larger field characteristic p. Recently, there are a few papers that discuss algorithms in extension fields of larger degrees such as [EPSV23], [BGDS23] and [SEMR23]. In particular, the motivation of [SEMR23] is exactly to relax the constrains on parameter choices for [DFKL$^+$20], and as a result, they achieve better verifying speed without degrading the signing speed as shown by their reference SageMath implementation. In this paper, we build upon these advancements and further explore computational subroutines critical for isogeny-based cryptographic schemes in extension fields.

1.1 Contributions

Let p be a prime > 3, E be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, and ℓ be an odd prime. Let k be the extension degree w.r.t. (p, ℓ) (Definition 2).

In this paper, we look at three computational subroutines:

1. **Torsion basis generation** In Section 3, we provide a new algorithm (Algorithm 1) that finds a basis of $E_0[\ell]$ where E_0 is the curve $y^2 = x^3 + x$ over $\mathbb{F}_{p^2}$ for $p \equiv 3 \bmod 4$ with the extra conditions that $\ell \equiv 1 \bmod 4$ and $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$. This algorithm takes $\widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$ and surpasses other known methods when $k > 5$ according to our experiments.
2. **Kernel polynomial computation** In Section 4, we provide a new algorithm (Algorithm 3) that computes the kernel polynomial of a kernel subgroup H given a minimal polynomial w.r.t. H (see Definition 4). This algorithm mainly follows [EPSV23, Algorithm 4], and it takes $O(k^2)+\widetilde{O}(\ell^3/k^2)+\widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$. Theoretical analysis indicates that the cost of our algorithm outperforms [EPSV23, Algorithm 4] when $k^3 > \ell^2$. The experimental results from running our SageMath implementation suggest that the improvement of our algorithm over [EPSV23, Algorithm 4] is only clear when k is large enough, e.g., $k > 300$.
3. **Ideal kernel points generation** In Section 5, we provide an algorithm (Algorithm 4) that computes the kernel of the corresponding isogeny given a left $\mathcal{O}_0$-ideal where $\mathcal{O}_0$ is a quaternion maximal order isomorphic to $\mathrm{End}(E_0)$. It combines Algorithm 1 with a simple yet effective trick (Lemma 17) that we observe. We incorporate Algorithm 4 to [EPSV23, Algorithm 2]. This speeds up the kernel points generation step by around 40% in the constructive Deuring correspondence algorithm from [EPSV23] and speeds up the full algorithm by around 30%.

We also provide a SageMath [Dev23] implementation to measure the performance of our algorithms in this work. The code is available at https://github.com/wennycai/isoext/tree/master.

2 Preliminaries

In this section, we provide a brief overview of the background necessary for this work.

2.1 Quaternion algebras, supersingular elliptic curves, and isogenies

Quaternion Algebra Let p be a prime and let $\mathcal{B}_{p,\infty}$ denote the unique (up to isomorphism) quaternion algebra ramified precisely at p and ∞. We fix a $\mathbb{Q}$-basis $\langle 1, \mathbf{i}, \mathbf{j}, \mathbf{k} \rangle$ of $\mathcal{B}_{p,\infty}$ that satisfies $\mathbf{i}^2 = -q$, $\mathbf{j}^2 = -p$ and $\mathbf{k} = \mathbf{ij} = -\mathbf{ji}$ for some integer q. A *fractional ideal* I in $B_{p,\infty}$ is a $\mathbb{Z}$-lattice of rank 4. We denote by $n(I)$ the *norm* of I as the largest rational number such that $n(\alpha) \in n(I)\mathbb{Z}$ for any $\alpha \in I$. An order $\mathcal{O}$ is a subring of $B_{p,\infty}$ that is also a fractional ideal. An order is called *maximal* when it is not contained in any other larger order. A fractional ideal is *integral* if it is contained in its *left order* $\mathcal{O}_L(I) = \{\alpha \in \mathcal{B}_{p,\infty} \mid \alpha I \subset I\}$, or equivalently in its *right order* $\mathcal{O}_R(I) = \{\alpha \in \mathcal{B}_{p,\infty} \mid I\alpha \subset I\}$.

Supersingular elliptic curves and their isogenies Let E, E_1, E_2 be elliptic curves defined over a finite field of characteristic p. An isogeny from E_1 to E_2 is a non-constant rational map that is simultaneously a group homomorphism. An isogeny is called as *cyclic* if it is not $[k]\varphi'$ for any integer k and φ' another isogeny from E_1 to E_2. An isogeny from a curve E to itself is an *endomorphism*. The set $\mathrm{End}(E)$ of all endomorphisms of E forms a ring under addition and composition. $\mathrm{End}(E)$ is either an order in an imaginary quadratic field and E is called *ordinary*, or a maximal order in $\mathcal{B}_{p,\infty}$, in which case E is called *supersingular*.

Let $\phi : E_1 \to E_2$ be an isogeny and let its degree to be odd. According to [Gal12, Lemma 25.1.16], ϕ can be written as

$$\phi(x, y) = \left(\frac{A(x)}{\psi^2(x)}, \frac{B(x, y)}{\psi^3(x)} \right),$$

where $\psi(x)$ is a polynomial of degree $(\deg \phi - 1)/2$. We refer to this form as *standard form* of an isogeny ϕ in this paper.

For an supersingular elliptic curve over $\mathbb{F}_q$ where q is a power of p, we denote its *quadratic twist* over $\mathbb{F}_q$ by E^t which is a curve isomorphic to E over a quadratic extension of $\mathbb{F}_q$.

As an important integer that will be used throughout the paper, we give precise definition of *extension degree.*

Lemma 1. *Let E be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, let ℓ be an odd prime and e be a positive integer, the following three descriptions determine the same integer.*

1. *The smallest integer k such that $E[\ell^e] \subseteq E(\mathbb{F}_{p^{2k}})$ or $E^t[\ell^e] \subseteq E^t(\mathbb{F}_{p^{2k}})$ where E^t is the quadratic twist of E over $\mathbb{F}_{p^{2k}}$.*
2. *The degree of the minimal polynomial of $x(P)$ over $\mathbb{F}_{p^2}$ for any order ℓ^e point in $E[\ell^e]$.*

3. Order of p^2 in $(\mathbb{Z}/\ell^e\mathbb{Z})^\times$, i.e., the smallest integer such that $p^{2k} \equiv 1 \bmod \ell^e$.

Proof. The equivalences follow from the definition of a quadratic twist, the structure of $E(\mathbb{F}_{p^{2k}})$ and $E^t(\mathbb{F}_{p^{2k}})$, and the fact that $\#E(\mathbb{F}_{p^{2k}}) \times \#E^t(\mathbb{F}_{p^{2k}}) = (p^{2k}-1)^2$ ([Sch87, Lemma 4.8]). □

Definition 2. *Given p and ℓ^e, we call the integer defined from Lemma 1 the extension degree w.r.t. (p, ℓ^e) and we denote it by k.*

The Deuring correspondence Fix a supersingular elliptic curve E_0, and an order $\mathcal{O}_0 \simeq \mathrm{End}(E_0)$. The curve/order correspondence allows one to associate to each outgoing isogeny $\varphi : E_0 \to E_1$ an integral left $\mathcal{O}_0$-ideal, and every such ideal arises in this way (see [Koh96] for instance). Through this correspondence, the ring $\mathrm{End}(E_1)$ is isomorphic to the right order of this ideal. This isogeny/ideal correspondence is defined in [Wat69], and in the separable case, it is explicitly given as follows.

Definition 3. *Given I an integral left $\mathcal{O}_0$-ideal coprime to p, we define the I-torsion $E_0[I] = \{P \in E_0(\overline{\mathbb{F}}_{p^2}) | \alpha(P) = 0$ for all $\alpha \in I\}$. To I, we associate the separable isogeny φ_I of kernel $E_0[I]$. Conversely given a separable isogeny φ, the corresponding ideal is defined as $I_\varphi = \{\alpha \in \mathcal{O}_0 \mid \alpha(P) = 0$ for all $P \in \ker(\varphi)\}$.*

We summarize properties of the Deuring correspondence in Table 1, borrowed from [DFKL$^+$20].

Table 1. The Deuring correspondence, a summary [DFKL$^+$20].

Supersingular j-invariants over $\mathbb{F}_{p^2}$ $j(E)$ (up to Galois conjugacy)	Maximal orders in $\mathcal{B}_{p,\infty}$ $\mathcal{O} \cong \mathrm{End}(E)$ (up to isomorphism)
(E_1, φ) with $\varphi : E \to E_1$	I_φ integral left $\mathcal{O}$-ideal and right $\mathcal{O}_1$-ideal
$\theta \in \mathrm{End}(E_0)$	Principal ideal $\mathcal{O}\theta$
$\deg(\varphi)$	$n(I_\varphi)$

2.2 Kernel polynomial and minimal polynomial of a kernel subgroup

Let $E/\mathbb{F}_{p^2}$ be a supersingular elliptic curve, the *kernel polynomial* defining a finite subgroup $H \leq E$, or equivalently an isogeny with kernel H, is the unique monic squarefree polynomial h_H whose set of roots is precisely the set of x-coordinates of nonzero points in H. A related concept is the *minimal polynomial of H*, whose definition is introduced in [EPSV23] and we recall here.

Definition 4. *[EPSV23, Definition 15] Let E be an elliptic curve over a field K and $f \in K[X]$ a monic squarefree polynomial. The subgroup defined by f is the subgroup H of E generated by the set of points $\{P \in E\backslash\{0\} | f(x(P)) = 0\}$. In this situation, we say that f is a defining polynomial for H, and if f is furthermore irreducible, we refer to f as a minimal polynomial of H.*

Let ℓ be an odd prime and $H \leq E$ be a cyclic subgroup of order ℓ defined over $\mathbb{F}_{p^2}$. H gives rise to a unique cyclic isogeny $\varphi : E \to E'$ such that $\ker(\varphi) = H$ up to post-composition, and φ is defined over $\mathbb{F}_{p^2}$. Note that as discussed in [EPSV23, Section 2.2], in fact any isogeny between two supersingular elliptic curves can be defined over $\mathbb{F}_{p^2}$, by working with isomorphism representatives on which the p^2-power Frobenius is a scalar.

We denote the kernel polynomial of H by h_H. $h_H(x) \in \mathbb{F}_{p^2}[x]$ as H is defined over $\mathbb{F}_{p^2}$, and h_H is of degree $\frac{\ell-1}{2}$ by its definition. Let $H = \langle P \rangle$, and let $f(x)$ denote the minimal polynomial of $x(P)$ over $\mathbb{F}_{p^2}$, denote its degree by k, clearly $f(x) \mid h_H(x)$. Let π denote the p^2-power Frobenius map, $x(\pi(P)) = x(P)^{p^2}$ is another root of $f(x)$, and therefore $\pi(P)$ is another generator of H and there exists $\lambda \in (\mathbb{Z}/\ell\mathbb{Z})^\times$ such that $\pi(P) = [\lambda]P$.

Clearly, there is a free and transitive action of the group $(\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$ on the set $X := H - 0_E/\{\pm 1\}$ by scalar multiplication. The subgroup $\langle \lambda \rangle/\{\pm 1\}$ partitions X into $\frac{\ell-1}{2k}$ orbits, each orbit is of length k. Let a be a primitive root in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, then

$$h_H(x) = \prod_{i=0}^{\frac{\ell-1}{2k}-1} \left(\prod_{j=0}^{k-1} (x - x([a^i \lambda^j]P))\right). \tag{1}$$

This follows from the fact that $\{a^i\}_{i=0}^{\frac{\ell-1}{2k}-1}$ is a set of coset representatives of $\langle \lambda \rangle/\{\pm 1\}$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$, and $\{\lambda^j\}_{j=0}^{k-1}$ is a set of representatives of $\langle \lambda \rangle/\{\pm 1\}$. For each fixed i, the inner layer product $\prod_{j=0}^{k-1}(x - x([a^i\lambda^j]P))$ is the minimal polynomial of $x([a^i]P)$ over $\mathbb{F}_{p^2}$ of degree k.

2.3 Torsion basis generation algorithms over extension fields

In this section, we revisit the following problem of computing a torsion basis of supersingular elliptic curves.

Problem 5. Let E be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, ℓ be a prime and e be a positive integer, compute a $\mathbb{Z}/\ell^e\mathbb{Z}$-basis of $E[\ell^e]$.

Torsion basis generation in isogeny-based cryptography is a fundamental step that is involved in the efficiency of lots of constructions. In most isogeny-based schemes, one prefers choosing a good characteristic p to make sure the extension degree k w.r.t. (p, ℓ^e) equals to 1. This choice leaves solving Problem 5 for general p, ℓ^e less studied. In what follows, we summarize two recent algorithms that considers solving Problem 5 in extension fields. We note that in both works, the torsion basis generation algorithms are not their main focuses but rather a tool developed for further computational tasks.

Method 1 The authors of [EPSV23] use a straightforward method to compute the torsion basis, which is by sampling random elements in $E(\mathbb{F}_{p^{2k}})$ and multiplying the points by a cofactor $\frac{(p^k \pm (-1)^k)^2}{\ell^e}$ until a $\mathbb{Z}/\ell^e\mathbb{Z}$-linearly independent pair P, Q is found. This method costs $O(k \log p)$ operations in $\mathbb{F}_{p^{2k}}$ [EPSV23, Section 3.3].

Method 2 This method is introduced in [BGDS23, Appendix A] and we briefly recall the main ideas here. Let π be the p^2-power Frobenius endomorphism on E and $\Phi_k(x)$ denote the k-th cyclotomic polynomial. Since $\Phi_k(x) | x^k - 1$, there exists an endomorphism θ_k on E such that $(\pi^k - \mathrm{id}_E) = \Phi_k(\pi) \circ \theta_k$. For any point $P \in E(\mathbb{F}_{p^{2k}})$, $\theta_k(P) \in \ker(\Phi_k(x))$, then sampling random points P and computing $[\frac{\# \ker(\Phi_k(\pi))}{\ell^e}]\theta_k(P)$ would give rise to a basis of $E[\ell^e]$ similar to the idea of the previous method. This method costs $O(\varphi(k) \log p)$ operations in $\mathbb{F}_{p^{2k}}$ [BGDS23, Appendix A], where $\varphi(k)$ represents the value of the Euler totient function.

2.4 Constructive Deuring correspondence

The constructive Deuring correspondence problem is the following:

Problem 6. Given a maximal order $\mathcal{O}$ in $\mathcal{B}_{p,\infty}$, compute a supersingular elliptic curve $E/\mathbb{F}_{p^2}$ such that $\mathrm{End}(E) \cong \mathcal{O}$.

A natural strategy to tackle this, which is also used in [EPSV23], goes as follows:

Step 0: Fix a base curve $E_0/\mathbb{F}_{p^2}$ with a known effective endomorphism ring $\mathcal{O}_0$.
Step 1 (KLPT): Construct an ideal I connecting $\mathcal{O}_0$ and $\mathcal{O}$ of suitable norm.
Step 2(IdealToIsogeny): Compute the isogeny corresponding to I as $\varphi_I : E_0 \to E$.

[EPSV23] provides the most efficient algorithm for computing this correspondence in the literature.

3 Torsion basis generation

Let p be a prime such that $p \equiv 3 \bmod 4$. Let E_0 be a supersingular elliptic curve over $\mathbb{F}_p$ defined by the equation $y^2 = x^3 + x$. There exists an automorphism τ on E_0 given by the map $\tau : (x, y) \mapsto (-x, \sqrt{-1}y)$ where $\sqrt{-1}$ is the square root of -1 in $\mathbb{F}_{p^2}$. In this section, we propose a new method that finds a basis of $E_0[\ell]$ for an odd prime ℓ when ℓ satisfies the additional conditions that:

- $\ell \equiv 1 \bmod 4$;
- $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, this is equivalent to $\mathrm{ord}(p^2) = \frac{\ell-1}{2}$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ given the previous condition, i.e., the extension degree is $\frac{\ell-1}{2}$.

The main idea of our method is to extract the minimal polynomials of $x(P), x(Q)$ where P, Q form a basis of $E[\ell]$ from the denominator of the x-coordinate of a degree ℓ endomorphism θ on E and its dual $\hat{\theta}$. Since the extension degree $k = \frac{\ell-1}{2}$, the x-coordinates of P, Q are defined over $\mathbb{F}_{p^{\ell-1}}$. However, our method allows us to only work in the base field $\mathbb{F}_{p^2}$ to obtain the minimal polynomials. This significantly reduces the cost when the extension degree is large.

3.1 New algorithm for computing $E_0[\ell]$

Lemma 7. *Let ℓ be an odd prime such that $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$. Let $\theta = \left(\frac{\psi_x(x)}{\phi_x(x)}, \frac{\psi_y(x)}{\phi_y(x)}y\right)$ be an endomorphism of degree ℓ defined over $\mathbb{F}_{p^2}$ in its standard form (see definition in Section 2.1). Let R be any generator of $\ker\theta$. Then $\phi_x(x) = f^2(x)$ where $f(x)$ is the minimal polynomial of $x(R)$.*

Proof. Since $\deg\theta = \ell$ and ℓ is odd, we have $\phi_x(x) \in \mathbb{F}_{p^2}[x]$ is of degree $\ell - 1$ and is a square. Since $x(P)$ is a root of $\phi_x(x)$, then $f(x) \mid \phi_x(x)$. Note that $\mathrm{ord}(p) = \ell - 1$ implies that $f(x)$ is of degree $\frac{\ell-1}{2}$ (see Lemma 1), then it follows immediately that $\phi_x(x) = f^2(x)$. □

Algorithm 1 $\mathsf{TorsionBasisE_0}(\ell)$

```
Input: A prime $\ell \equiv 1 \bmod 4$ such that $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$.
Output: Minimal polynomials of $x$-coordinates of a basis $P, Q$ of $E_0[\ell]$.
1: Compute $a, b$ such that $a^2 + b^2 = \ell$ by Cornacchia's algorithm [Coh93, Section 1.5.2].
2: Compute the $x$-coordinates of the standard form of $\theta = [a] + \tau[b]$ and $\hat{\theta} = [a] - \tau[b]$,
   which we denote by $\frac{A(x)}{F(x)}$ and $\frac{B(x)}{\tilde{F}(x)}$ respectively.
3: Compute the square roots of $F(x)$ and $\tilde{F}(x)$ as $f_P(x)$ and $f_Q(x)$.
4: Return $f_P(x), f_Q(x)$.
```

Lemma 8. *Let p, E_0 and τ be as introduced in the beginning of Section 3. Let ℓ be an odd prime with $\ell \equiv 1 \bmod 4$ such that $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$. Algorithm 1 returns the minimal polynomials the x-coordinates of a basis of $E_0[\ell]$, and its takes $\widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$.*

Proof. The correctness follows from Lemma 7 and the fact that a generator of $\ker(\theta)$ and a generator of $\ker(\hat{\theta})$ are $\mathbb{Z}/\ell\mathbb{Z}$-linearly independent. Indeed, let $\ker(\theta) = \langle R\rangle$, since $\hat{\theta} = 2a - \theta$, $\ker(\hat{\theta}) = \langle R\rangle$ if and only if $\ell \mid 2a$ which is impossible by our choice of a.

As for complexity, the bit-complexity for Step 1 is $O(\log^2 \ell)$ [FS16, Section 3.1]. Step 2 takes $\widetilde{O}(\ell)$ operations in $\mathbb{F}_p$ [ACL+23, Lemma 2.7]. At Step 3, since $F(x) = f_P^2(x)$ from Lemma 7 and $f_P(x)$ is irreducible over $\mathbb{F}_{p^2}[x]$, we have $\gcd(F(x), F'(x)) = f_P(x)$, where $F'(x)$ is the derivative of $F(x)$. Hence, Step 3 requires $\widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$ using FFT-based polynomial arithmetic. □

Generalization of the torsion basis computation algorithm (Algorithm 1). At first glance, it may seem like our method is very restrictive as it is for only one curve E_0 and the prime ℓ needs to satisfy extra conditions. In fact, the curve E_0 plays a major role in isogeny-based cryptography. Assuming that $p \bmod \ell$ is random in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, then the condition $\mathrm{ord}(p) = \ell - 1$ holds with probability $\frac{\varphi(\ell-1)}{\ell-1}$ where φ is the Euler's totient function. As ℓ considered in isogeny-based cryptography is significantly smaller than p, we compute the ratio $\frac{\varphi(\ell-1)}{\ell-1}$ for all primes $\ell < 5000$ such that $\ell \equiv 1 \bmod 4$, and they have $\frac{1}{5}$ as a lower bound. This means that for a random pair (p, ℓ) with $p \equiv 3 \bmod 4$ and $\ell \equiv 1 \bmod 4$ and $\ell < 5000$, there is at least a chance of $\frac{1}{5}$ that our algorithm can be applied.

Our method can be generalized to other curves E as long as E has a known endomorphism τ such that ℓ splits in $\mathbb{Z}[\tau]$. We can also remove the condition that $\operatorname{ord}(p^2) = \frac{\ell-1}{2}$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ and even extend to computing a basis for the ℓ^e-torsion. The main idea still applies but the algorithm now requires factoring $F(x)$ and $\tilde{F}(x)$ as the minimal polynomials of the x-coordinates of a torsion basis is only a factor of their square roots.

3.2 Experiments and performance

We implement Algorithm 1 in SageMath, and compare with the results running the SageMath implementations of the two methods introduced in Section 2.3. The comparison between the three methods are given in Table 2.

We choose $\ell \in \{5, 13, 17, 41, 97, 193\}$, and for each ℓ, we randomly generate a 256-bit p that satisfies the conditions that $p \equiv 3 \bmod 4$ and $\operatorname{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$. Note that in our case when $\ell \equiv 1 \bmod 4$, we have that $E_0[\ell] \subseteq E_0(\mathbb{F}_{p^{2(\ell-1)}})$ while $E_0^t[\ell] \subseteq E_0^t(\mathbb{F}_{p^{\ell-1}})$. Therefore, it is more efficient for Method 1 to work with the twist curve E_0^t and then sends back the torsion basis to E_0 via their isomorphism. However, this trick does not apply to Method 2 as Method 2 works with a curve on which the p^2-power Frobenius is defined.

Note that Algorithm 1 outputs minimal polynomials of $x(P)$ and $x(Q)$ while the other two methods give the x, y coordinates for both points. For a fair comparison, we also recover P and Q from $f_P(x)$ and $f_Q(x)$. Note that $\mathbb{F}_{p^{\ell-1}}$ can be constructed as an extension of $\mathbb{F}_{p^2}$ with modulus equals to $f_P(x)$ or $f_Q(x)$, which means the x-coordinate of torsion basis is the generator β of $\mathbb{F}_{p^{\ell-1}}$. The y-coordinate of torsion basis can be recovered by computing a square root on the twisted curve.

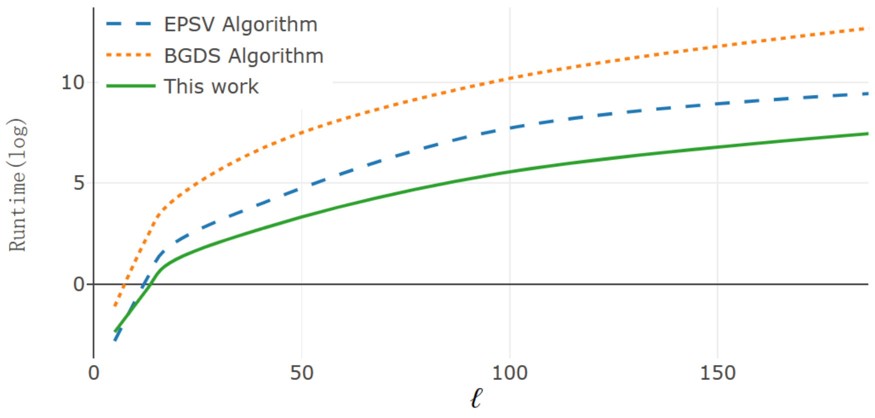
Table 2. Timings of torsion basis generation for different method. Since method 1 is a probabilistic algorithm, we run 10 times for each method and take the average as the final runtime. All of these benchmarks are running in SageMath 10.0 on a laptop with an Intel Core i7-12700H processor.

Method	$\ell = 5$	$\ell = 13$	$\ell = 17$	$\ell = 41$	$\ell = 97$	$\ell = 193$
Method 1 [EPSV23]	$0.14s$	$1.26s$	$3.17s$	$16.50s$	$196.88s$	$737.17s$
Method 2 [BGDS23]	$0.46s$	$5.30s$	$13.46s$	$109.73s$	$1080.62s$	$7369.61s$
This work	$0.19s$	$0.86s$	$1.82s$	$6.86s$	$44.11s$	$190.90s$

According to Table 2, we give a logarithmic runtime picture, where the runtime of torsion basis generation is represented on a logarithmic scale, as shown in Figure 1. As ℓ increases, the growth rate of runtime for the three algorithms slightly decelerates. Overall, Algorithm 1 significantly outperforms both Method

1 and Method 2. Therefore, for large primes $\ell \equiv 1 \pmod 4$ with $\mathrm{ord}(p) = \ell - 1$ in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, using Algorithm 1 for generating a torsion basis of $E_0[\ell]$ is much more efficient.

Fig. 1. We plot the following graphs with data points provided in Table 2. The x-axis represents the values of ℓ. The y-axis represents the logarithm of the runtime (in seconds) from Table 2. This figure is mainly to illustrate the growth trend of the runtime of the three algorithms as ℓ increases.



4 Computing kernel polynomials

Given a kernel generator $P \in E$ of prime order ℓ that lives in an extension field $\mathbb{F}_{p^{2k}}$ over $\mathbb{F}_{p^2}$, the time complexity of computing ϕ defined by the kernel subgroup $\langle P \rangle$ using Velu's formula is $\widetilde{O}(\ell)$ operations over $\mathbb{F}_{p^{2k}}$. An alternative method is to use Kohel's formulas. Given the kernel polynomial of the isogeny ϕ, it takes $\widetilde{O}(\ell)$ operations over $\mathbb{F}_{p^2}$ to compute ϕ. [EPSV23, Algorithm 4] proposes a new method to compute the kernel polynomial of ϕ given P, which takes $O(k^2) + O(\ell k) + \widetilde{O}(\ell)$ operations in the field $\mathbb{F}_{p^2}$. This leads to an algorithm that computes the isogeny ϕ given P which can be faster than Vélu's formulas in some cases. Moreover, when it comes to evaluating a point $P' \in \mathbb{F}_{p^{2k'}}$, Vélu's formulas perform this evaluation in the composition of the two fields $\mathbb{F}_{p^{2k}}$ and $\mathbb{F}_{p^{2k'}}$ whereas Kohel's formulas perform this evaluation in $\mathbb{F}_{p^{2k'}}$. Therefore, in the application scenarios when evaluations of points defined over different field extensions are needed, it can be preferable to use Kohel's formula as well.

In this section, we further improve on [EPSV23, Algorithm 4] that computes the kernel polynomial of φ given P, precisely, the x-coordinate of P. Their main idea is to compute all the irreducible factors of the kernel polynomial and then multiply them together, and each irreducible factor is computed via Shoup's algorithm given a root of the irreducible factor. The main idea of our new algorithm is a new method that computes the irreducible factors that outperforms Shoup's algorithm for some parameter choices.

4.1 Kernel polynomial computation algorithm

Let us recall that from Section 2.2, $\pi(P) = [\lambda]P$ with $\lambda \in (\mathbb{Z}/\ell\mathbb{Z})^\times$. The subgroup $\langle\lambda\rangle/\{\pm 1\} \subseteq (\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$ is the unique subgroup of order k in $(\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$ and we denote it by $\mathcal{S}$. Our new algorithm starts with finding a pair of integers (a, b) that is *good* in the sense of the following definition.

Definition 9. *We say a pair of integers (a, b) is (ℓ, k)-good if it satisfies the following:*

(1) modulo ℓ, a, b are primitive roots in $(\mathbb{Z}/\ell\mathbb{Z})^\times$,
(2) viewed as elements in $(\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$, b and a are in the same coset with respect to the subgroup $\mathcal{S}$,
(3) $\gcd(a, b) = 1$.

Any integer a gives rise to an endomorphism on a supersingular elliptic curve E which is simply the multiplication-by-a map often denoted by $[a]$. We introduce a new notation $[a]_x(x)$ which is the x-coordinate of $[a]$ when written as rational maps.

Let us write $[a]_x(x) = \frac{a_1(x)}{a_2(x)}$ with $a_1(x), a_2(x) \in \mathbb{F}_{p^2}[x]$ be coprime polynomials with degrees bounded above by $O(a^2)$, and $[b]_x(x) = \frac{b_1(x)}{b_2(x)}$ with $b_1(x), b_2(x) \in \mathbb{F}_p[x]$ be coprime polynomials with degrees bounded above by $O(b^2)$. Let $f(x) = \sum_{i=0}^k c_i x^i$ with $c_i \in \mathbb{F}_{p^2}$ be the minimal polynomial of $x(P)$ over $\mathbb{F}_{p^2}$. Then

$$f([a]_x(x)) = \frac{\sum_{i=0}^k c_i a_2^{k-i}(x) a_1^i(x)}{a_2^k(x)} =: \frac{d_{fa}(x)}{n_{fa}(x)},$$

$$f([b]_x(x)) = \frac{\sum_{i=0}^k c_i b_2^{k-i}(x) b_1^i(x)}{b_2^k(x)} =: \frac{d_{fb}(x)}{n_{fb}(x)}.$$

Lemma 10. *Let a, b be two integers satisfying the conditions* (2) *and* (3) *in Definition 9, then* $\gcd(d_{fa}(x), d_{fb}(x))$ *is the minimal polynomial of* $x([a^{-1}]P)$ *where here a^{-1} is the inverse of a modulo ℓ.*

Proof. Since $[a]([a^{-1}]P) = P$, we have that $[a]_x(x([a^{-1}]P)) = x(P)$ and it is a root of $f(x)$. Therefore, $x([a^{-1}]P)$ is a root of $f([a]_x(x))$, and it means that $d_{fa}(x([a^{-1}]P)) = 0$. Similarly, we can show that $d_{fb}(x([b^{-1}]P)) = 0$. Since a, b are in the same coset of $(\mathbb{Z}/\ell\mathbb{Z})^\times/\{\pm 1\}$ modulo $\langle\lambda\rangle/\{\pm 1\}$, $x([a^{-1}]P)$ and $x([b^{-1}]P)$ have the same minimal polynomial, let us denote it by $f_1(x)$. Then this implies that $f_1(x) \mid \gcd(d_{fa}(x), d_{fb}(x))$.

We now show that $d_{fa}(x)$ and $d_{fb}(x)$ have no other common roots besides those shared with $f_1(x)$. Consider the two sets $[a^{-1}](\mathcal{S} \cdot P)$ (the orbit of P of the

$\mathcal{S}$ action multiplied with $[a^{-1}]$) and $[b^{-1}](\mathcal{S}\cdot P)$, they are equal by the conditions on a, b. The roots of $d_{fa}(x)$ are x-coordinates of $[a^{-1}](\mathcal{S}\cdot P)\tilde{+}E[a]$ (where the addition $\tilde{+}$ is performed over all possible pairwise combinations by summing each element from the first set with each element from the second set), and the roots of $d_{fb}(x)$ are x-coordinates of $[b^{-1}](\mathcal{S}\cdot P)\tilde{+}E[b]$. Since $\gcd(a, b) = 1$, one can see that there are no common elements in these two sets except for $[a^{-1}](\mathcal{S}\cdot P)$. This shows that $f_1(x) = \gcd(d_{fa}(x), d_{fb}(x))$. □

Heuristic 11. *We can find desired $a, b \in [-B, B] - \{0\}$ with B a small multiple of $\frac{\ell-1}{4k}$.*

We provide a justification of the Heuristic above.

- There must be two integers in $[-B, B] - \{0\}$ belong to the same coset whenever $B > \frac{\ell-1}{4k}$ since then $2B > \frac{\ell-1}{2k}$ and $\frac{\ell-1}{2k}$ is the number of cosets.
- The chance that the other two conditions are satisfied for a random pair in $[-B, B] - \{0\}$ is reasonably high, we expect to obtain one such pair by adding a small constant multiple to $\frac{\ell-1}{4k}$.

Remark 12. Asymptotically, we expect to find the desired $a, b \in [-B, B] - \{0\}$ when B is a small multiple of $((\ell-1)/k)^{1/2}$ by the birthday paradox. This can be used to improve the complexity of Algorithm 2 and then Algorithm 3. But we do not use this bound since we are dealing with ℓ and k where ℓ/k is not very large.

Algorithm 2 KernelPolynomialFromMinipoly(E,f,ℓ)

```
Input: An elliptic curve $E/\mathbb{F}_{p^2}$, a prime $\ell$ and a minimal polynomial $f$ of $x(P)$ of
    degree $k$ where $P$ is a point of order $\ell$.
Output: The kernel polynomial $h$ w.r.t. the kernel subgroup $\langle P \rangle$.
1: Set $m \leftarrow (\ell-1)/2k$ and $f_0 \leftarrow f$.
2: Find smallest pair $(a, b)$ (smallest in terms of $\max\{|a|, |b|\}$) that is $(\ell, k)$-good. Let
    $B = \max\{|a|, |b|\}$).
3: For $i = 1$ to $m-1$ do
4:     Compute $[a]_x(x) = \frac{a_1(x)}{a_2(x)}$ and $[b]_x(x) = \frac{b_1(x)}{b_2(x)}$.
5:     Construct polynomials $d_{fa}(x) \leftarrow a_2^k(x) f_{i-1}([a]_x(x))$, $d_{fb}(x) \leftarrow b_2^k(x) f_{i-1}([b]_x(x))$.
6:     Set $f_i(x) \leftarrow \gcd(d_{fa}(x), d_{fb}(x))$.
7: end For
8: Compute $h \leftarrow \prod_{i=0}^{m-1} f_i$ using a product tree.
9: Return $h$.
```

Lemma 13. *Assuming Heuristic 11, Algorithm 2 is correct and it takes $\widetilde{O}(\frac{\ell^3}{k^2}) + \widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$.*

Algorithm 3 KernelPolynomialFromIrrationalX(E, ζ, ℓ)

```
Input: Elliptic curve $E/\mathbb{F}_{p^2}$, extension $\mathbb{F}_{p^{2k}}/\mathbb{F}_{p^2}$, $x$-coordinate $\zeta \in \mathbb{F}_{p^{2k}}$ of an order-$\ell$
    point $P \in E$ lying in an eigenspace of the $p^2$-power Frobenius on $E$.
Output: The kernel polynomial $h_{\langle P \rangle} \in \mathbb{F}_{p^2}[x]$ defining the subgroup of $E$ generated
    by $P$.
1: Find the minimal polynomial $f \in \mathbb{F}_{p^2}[x]$ of $\zeta$ over $\mathbb{F}_{p^2}$ using Shoup's algorithm.
2: Return KernelPolynomialFromMinipoly$(E, f, \ell)$.
```

Proof. The correctness of Algorithm 2 follows from Lemma 10 and Eq. (1). The smallest (a, b) pair that is (ℓ, k) good can be found by an exhaustive search, i.e., by enumerating all (a, b) pairs in each coset such that $|a|, |b| \leq B$ and check whether they are (ℓ, k)-good. The cost of finding (a, b) is $\widetilde{O}(\log^3(\ell)B^2)$ bit operations. The cost of Step 4 - 6 is $\widetilde{O}(kB^2)$ using FFT as explained in Remark 14, which is $\widetilde{O}(\frac{\ell^2}{k})$ according to Heuristic 11. Therefore the loop takes $\widetilde{O}(\frac{\ell^3}{k^2})$ operations in $\mathbb{F}_{p^2}$. Finally, the product tree in the last step can be computed in $\widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$, using FFT based arithmetic, same as [EPSV23, Lemma 16]. $\square$

Remark 14. We explain how to compute $d_{fa}(x)$, $d_{fb}(x)$ and $f_i(x)$ here in more detail. If $2^{\lfloor \log_2^{k \cdot a^2+1} \rfloor} | p^2 - 1$, then there always exist $2^{\lfloor \log_2^{k \cdot a^2+1} \rfloor}$-th roots of unity over $\mathbb{F}_{p^2}$, we denote the set of these units to be μ_a. For $i = 1, \ldots, m-1$, we compute each $d_{fa}(x)$ in Algorithm 2 by first computing $Y_a := \{[a]_x(\eta), \eta \in \mu_a\}$, we then compute the evaluation of $f_{i-1}(x)$ at each element in Y_a in $\widetilde{O}(k)$ operations over $\mathbb{F}_{p^2}$. Taking these evaluation values, the inverse FFT algorithm can be used to recover the coefficients of $d_{fa}(x)$ in $\widetilde{O}(kB^2)$ operations in $\mathbb{F}_{p^2}$. The construction of $d_{fb}(x)$ is similar. Then the final gcd computation can be done in $\widetilde{O}(kB^2)$ operations in $\mathbb{F}_{p^2}$ with FFT.

Lemma 15. *Algorithm 3 is correct and takes $O(k^2) + \widetilde{O}(\frac{\ell^3}{k^2}) + \widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$.*

Proof. The complexity of Algorithm 3 is $O(k^2)$ for running Shoup's algorithm plus the complexity of Algorithm 2.

Remark 16. Let us recall here that the corresponding algorithm [EPSV23, Algorithm 4] takes $O(k^2) + O(\ell k) + \widetilde{O}(\ell)$ operations in $\mathbb{F}_{p^2}$. Therefore, the difference is only in the middle term and we expect to outperform [EPSV23, Algorithm 4] when $\frac{\ell^3}{k^2} < \ell k$, i.e., when $k^3 > \ell^2$. We test this observation with our experiments in next section.

4.2 Experiments and performance

We explain our strategy for generating the parameters k, ℓ, p. We first fix the extension degree k, then we find a prime ℓ such that $\frac{\ell-1}{2k} > 1$. We then find a pair (a, b) that is (ℓ, k)-good (Definition 9). Finally we look for primes of the form $2^r f \pm 1$ (for $p \equiv 1 \bmod 4$ and $p \equiv 3 \bmod 4$ respectively) with $r = \lfloor \log_2^{k\cdot\max(a,b)^2+1} \rfloor$ by randomly generating f and testing whether $2^r f \pm 1$ is a prime.

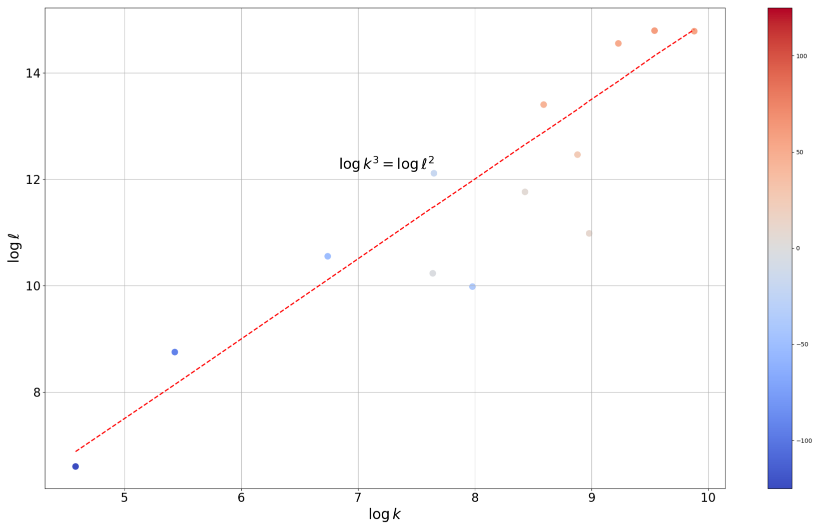
In the case of $p \equiv 1 \bmod 4$, the 2^r-th root of unity is an element in $\mathbb{F}_p$. However, since we consider the elliptic curve over $\mathbb{F}_{p^2}$, computing $[a]_x(x)$ and $[b]_x(x)$ still involves operations on $\mathbb{F}_{p^2}$ and there is not much difference in the case of $p \equiv 1 \bmod 4$ and $p \equiv 3 \bmod 4$. For convenience, we only discuss the case of $p \equiv 1 \bmod 4$ in this work. We use the above strategy to find 13 parameters for our experiments, with k ranging from 24 to 940. Besides, for these ℓ and k we can always find the smallest $a, b \in \{\pm 2, \pm 3\}$. Table 3 provides comparisons of computing the kernel polynomial for a given x-coordinate of an order-ℓ point.

From Remark 16, we expect that our algorithm can achieve better performance if $k^3 > \ell^2$. However, from Table 3, experiments suggests that our algorithm is faster than the previous method in [EPSV23] if $k > 300$ regardless of $k^3 > \ell^2$ or not. This is potentially caused by the fact that we have an extra log factor in $\widetilde{O}(\frac{\ell^3}{k^2})$ that was not taken into consideration in the comparison in Remark 16, therefore we perform worse when k is relatively small. On the other hand, the (a, b) pair we encounter in practice are much smaller than the upper bound we gave in the complexity analysis of Algorithm 2, this explains why we outperform [EPSV23] when k is large enough in practice.

Table 3. Timings of computing the kernel polynomial from the x-coordinate of generator for ℓ-isogeny, running in SageMath 10.1 on a computer with an Intel(R) Core(TM) i9-12900K processor. For comparison, we also record the runtime of this procedure in [EPSV23]. Both algorithms run in a same environment.

(ℓ, k)	[EPSV23]	Algorithm 3	Speed Up
$(97, 24)$	$0.04s$	$0.09s$	-125.00%
$(431, 43)$	$0.29s$	$0.56s$	-93.10%
$(1499, 107)$	$1.67s$	$2.51s$	-50.30%
$(1201, 200)$	$2.96s$	$3.04s$	-2.7%
$(4423, 201)$	$10.03s$	$11.90s$	-18.64%
$(1009, 252)$	$3.11s$	$4.31s$	-38.59%
$(3461, 346)$	$12.05s$	$11.41s$	5.31%
$(10781, 385)$	$47.51s$	$26.50s$	44.22%
$(5641, 470)$	$31.96s$	$23.90s$	25.22%
$(2017, 504)$	$11.38s$	$10.29s$	9.58%
$(24001, 600)$	$146.88s$	$71.42s$	51.38%
$(28387, 747)$	$196.60s$	$78.25s$	60.20%
$(28201, 940)$	$306.89s$	$136.20s$	55.62%

Fig. 2. Heatmap of timing comparison about computing kernel polynomials in Table 3. On the right side of this log-log picture is a colorbar from −100 (resp. blue) to 100 (resp. red), which indicates the percentage of speed up. The dotted red line represents the case when $k^3 = \ell^2$, i.e. $2 \log \ell = 3 \log k$. Hence, dots above the line correspond to the case when $k^3 > \ell^2$, and those below correspond to the case when $k^3 < \ell^2$. The color of the dots in the figure represents the speedup percentage.



5 Application to the constructive Deuring correspondence

In this section, we consider applications of our new algorithms to the constructive Deuring correspondence algorithm in [EPSV23]. The relevant part is Step 2 in the general strategy as defined in Section 2.4, i.e., the **IdealToIsogeny** step. Even though both of our two algorithms TorsionBasis$E_0(\ell)$(Algorithm 1) and KernelPolynomialFromIrrationalX(E, ζ, ℓ)(Algorithm 3) can be used in theory when the parameter requirements are met, we only incorporate our Algorithm 1 to the algorithm in [EPSV23] as the effect of the SageMath implementation of Algorithm 3 becomes clear only when k is bigger than 300, and this never happens for the extension degrees considered in [EPSV23].

Specifically, we use our Algorithm 1 in IdealToKernelGens ([EPSV23, Algorithm 2]). The cost of this algorithm consists of two main parts, one is computing a basis of $E_0[\ell^e]$ which costs $(k \log p)$ operations in $\mathbb{F}_{p^{2k}}$, the other one is evaluating a basis of $\mathcal{O}_0$ on the torsion basis points which costs $O(\log p)$ operations in $\mathbb{F}_{p^{2k}}$. We replace their method for generating a torsion basis algorithm for E_0 with ours whenever parameters p, ℓ satisfy our conditions. Besides this, we also observe one trick that could speed up their computation which is supported by the following lemma.

Lemma 17. *Let E be a supersingular elliptic curve, ℓ be a prime, and e be a positive integer. Let $P \in E[\ell^e]$ be of order ℓ^e, let $\gamma \in \mathrm{End}(E)$ be an endomorphism whose degree is ℓ^e, then $\gamma(P)$ is of order ℓ^e with probability $\frac{\ell}{\ell+1}$.*

Proof. There exists an explicit isomorphism of $E[\ell^e]$ and $\mathbb{Z}/\ell^e\mathbb{Z} \times \mathbb{Z}/\ell^e\mathbb{Z}$ such that under this isomorphism $\gamma = L$ where L is the map

$$\begin{aligned} L : \mathbb{Z}/\ell^e\mathbb{Z} \times \mathbb{Z}/\ell^e\mathbb{Z} &\to \mathbb{Z}/\ell^e\mathbb{Z} \\ (m, n) \quad &\mapsto m. \end{aligned}$$

A point $(m, n) \in \mathbb{Z}/\ell^e\mathbb{Z} \times \mathbb{Z}/\ell^e\mathbb{Z}$ is of order ℓ^e if and only if at least one of m and n is of order ℓ^e, and therefore that are $2 \times (\ell^e - \ell^{e-1})\ell^e - (\ell^e - \ell^{e-1})^2$ such elements. $L(m, n)$ is of order ℓ^e if and only m is of order ℓ^e and there are $(\ell^e - \ell^{e-1})\ell^e$ such choices. Therefore, the desired probability is

$$\frac{(\ell^e - \ell^{e-1})\ell^e}{2 \times (\ell^e - \ell^{e-1})\ell^e - (\ell^e - \ell^{e-1})^2} = \frac{\ell}{\ell + 1}.$$

□

In view of this lemma, it is of high probability that knowing one element of order ℓ^e in $E_0[\ell^e]$ suffices to find the kernel generator (see step 9 of [EPSV23, Algorithm 2]). Therefore, it is unnecessary to generate the full torsion basis as in step 8 of [EPSV23, Algorithm 2]. We introduce Algorithm 4 as a variant of [EPSV23, Algorithm 2] to include our Algorithm 1 and this new observation, Algorithm 4 deviates from [EPSV23, Algorithm 2] from step 8.

The asymptotic runtime of Algorithm 4 is the same as that of [EPSV23, Algorithm 2], but it performs better in practice. The performance depends heavily on the prime factors of $N(J) = \prod_{i=1}^{r} \ell_i^{e_i}$, so it is hard to give a simple estimate of how faster the new algorithm is than the old one, in particularly so for the basis generation step. If we only consider the trick of computing one less torsion basis, the speedup over [EPSV23, Algorithm 2] is $\sum_{i=1}^{r} \frac{2\ell_i}{\ell_i+1}$ times.

Implementations To measure the performance of our improvements, we compare the runtime of the constructive Deuring correspondence algorithm in [EPSV23] with our version replacing [EPSV23, Algorithm 2] by Algorithm 4.

We first run the comparison on the 3 parameters mentioned in [EPSV23]. For each parameter, we record the primes p and the norms of the ideals J connecting E_0[5] to E in Appendix A. The results are summarized in Table 4, and the table indicates that our work can significantly reduce the cost of the constructive Deuring correspondence computation, achieving approximately a 1.3x acceleration. However, we note that the requirements on ℓ are met only around 3 times in the whole iterations for parameter p_2 and 1 time for p_{3923}. Both of them need around 70 times iterations for kernel points generation in total. As a consequence, for both primes, Algorithm 1 only brings around 1% speed up.

[5] Note that when $p \equiv 1 \bmod 4$, we take E_0 to be the curve defined as in [EPSV23, Section 3.1] and our torsion basis generation algorithm can be adapted to it as well.

Algorithm 4 IdealToKerGens(J, E_0)

```
Input: left $\mathcal{O}_0$-ideal $J$ of norm $N = \prod_{i=1}^{r} \ell_i^{e_i}$, curve $E_0$ with effective endomorphism
    ring $\mathrm{End}(E_0) \cong \mathcal{O}_0$.
Output: $\{G_1, \dots, G_r\}$, a generating set of $\ker \varphi_J$ with $\mathrm{ord}(G_i) = \ell^{e_i}$.
 1: Compute $\alpha \in \mathrm{End}(E_0)$ such that $J = \mathcal{O}_0\alpha + \mathcal{O}_0 N$ under the isomorphism
    $\mathrm{End}(E_0) \cong \mathcal{O}_0$.
 2: let $(\phi_1, \dots, \phi_4)$ be a basis of $\mathrm{End}(E_0) \cong \mathcal{O}_0$ consisting of efficiently evaluatable
    endomorphisms.
 3: Write $\bar{\alpha}$ as a fraction of the form $(c_1\phi_1 + \ldots + c_4\phi_4)/t$, where $c_1$, $c_2$, $c_3$, $c_4 \in \mathbb{Z}$ and
    $t \in \mathbb{Z}_{\geq 1}$.
 4: For $i \in \{1, \dots, r\}$ do
 5:   Set $v_i = \nu_{\ell_i}(t)$ to be the $\ell_i$-adic valuation of $t$.
 6:   Let $c_j^{(i)} = c_j(t/\ell^{v_i})^{-1} \bmod \ell_i^{e_i+v_i}$ for $j \in \{1, \dots, 4\}$.
 7:   Define $\gamma_i = c_1^{(i)}\phi_1 + \cdots + c_4^{i}\phi_4$.
 8:   If $p \equiv 3 \bmod 4$ and $\ell_i \equiv 1 \bmod 4$ and $e_i + v_i = 1$ then
 9:     $f(x), f'(x) \leftarrow$ TorsionBasis$(\ell_i)$.
10:     Compute $\ell_i$-torsion point $P$ from $f(x)$.
11:     Compute $G_i \leftarrow \gamma_i(P)$.
12:     If $\ell^{e_i-1}G_i = \infty_{E_0}$ then
13:       Compute $\ell_i$-torsion point $Q$ from $f'(x)$.
14:       Compute $G_i \leftarrow \gamma_i(Q)$.
15:     end If
16:   else
17:     Find $\ell_i$-torsion point $P \in E_0$.
18:     Compute $G_i \leftarrow \gamma_i(P)$.
19:     If $\ell^{e_i-1}G_i = \infty_{E_0}$ then
20:       Find $\ell_i$-torsion point $Q \in E_0$.
21:       Compute $G_i \leftarrow \gamma_i(Q)$.
22:     end If
23:   end If
24: end For
25: Return $\{G_1, \dots, G_r\}$
```

Table 4. Timings of the IdealToKerGens algorithms (referred to as *kernel* in the table) and the full constructive Deuring correspondence computations (referred to as *Deuring* in the table) for different primes, using SageMath 10.0 on a laptop with an Intel Core i7-12700H processor. Since the IdealToKerGens algorithms are probabilistic, we execute 10 times benchmarks with random maximal orders and take the average as the final result for each characteristic.

Primes	*Kernel*			*Deuring*		
	[EPSV23, Alg 2]	Algorithm 4	Speed Up	[EPSV23]	This work	Speed Up
p_{3923}	$274.97s$	$167.05s$	39.25%	$520.07s$	$404.60s$	22.20%
p_1	$174.07s$	$107.83s$	38.05%	$352.42s$	$266.92s$	24.26%
p_2	$477.32s$	$299.57s$	37.24%	$689.76s$	$512.63s$	25.68%

In order to better measure the performance of the new torsion basis generation algorithm, we randomly generate three 256-bit primes and adjust cost model proposed in [EPSV23, Section 4.2] to allow more ℓ in the norm of the ideal (which is an input of the **KLPT** step, see Section 2.4) that satisfies the conditions of Algorithm 1. Specially, we consider all primes $\ell < 150$ and add it to the norm if it satisfies that $\ell \equiv 1 \bmod 4$ and that the order of p in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ is $\ell - 1$. The runtime of constructive Deuring correspondence computation for p_3, p_4, p_5 is provided in Table 5. According to the results, after adjusting cost model to use new algorithm for torsion basis generation more frequently, one can get a significant speed up during the computation of the constructive Deuring correspondence.

Table 5. Timings of the IdealToKerGens algorithms (referred to as *kernel* in the table) and the full constructive Deuring correspondence computations (referred to as *Deuring* in the table) for p_3, p_4 and p_5, using SageMath 10.0 on a laptop with an Intel Core i7-12700H processor. Since we want to measure the speedup brought by Algorithm 1 and the trick implied by Lemma 17, we add two invariants to our work. One is IdealToKerGens without using Algorithm 1 (referred to as *trick only*), the other one is IdealToKerGens with the trick and Algorithm 1, which is Algorithm 4. Since the kernel point generation algorithm is probabilistic, we execute 10 times benchmarks with random maximal orders and take the average as the final result.

		p_3		p_4		p_5	
		Kernel	*Deuring*	*Kernel*	*Deuring*	*Kernel*	*Deuring*
[EPSV23]		2217.06s	3205.94s	1424.43s	2007.57s	1371.94s	2318.64s
This work	*trick only*	1801.41s	2614.69s	1086.72s	1608.23s	928.96s	1823.63s
	Algorithm 4	1144.33s	1957.88s	832.24s	1353.75s	787.53s	1678.88s
Speed Up	*trick only*	18.75%	18.44%	23.71%	19.89%	32.29%	21.35%
	Algorithm 4	48.39%	38.93%	41.57%	32.57%	42.60%	27.59%

Acknowledgements We thank Lorenz Panny for helpful discussions which sparked the first algorithm. We thank Kaizhan Lin and the anonymous reviewers for helpful suggestions. Mingjie Chen and Christophe Petit are partly supported by EPSRC through grant number EP/V011324/1.

References

ACL$^+$23. Sarah Arpin, Mingjie Chen, Kristin Lauter, Renate Scheidler, Katherine Stange, and Ha Tran. Orienteering with one endomorphism. *La Matematica*, 2, 06 2023.

BDFLS20. Daniel J Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. In Steven Galbraith, editor, *Progress in ANTS XIV: Proceedings of the fourteenth algorithmic number theory symposium*, pages 39–55, Auckland, 2020. Mathematical Sciences Publishers.

BGDS23. Gustavo Banegas, Valerie Gilchrist, Anaëlle Le Dévéhat, and Benjamin Smith. Fast and frobenius: Rational isogeny evaluation over finite fields. In Abdelrahaman Aly and Mehdi Tibouchi, editors, *Progress in Cryptology – LATINCRYPT 2023*, pages 129–148, Cham, 2023. Springer Nature Switzerland.

Coh93. Henri Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 1993.

Dev23. The Sage Developers. Sagemath, the sage mathematics software system (version 10). https://www.sagemath.org/, 2023.

DFKL$^+$20. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 64–93, Cham, 2020. Springer International Publishing.

DFLLW23. Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New Algorithms for the Deuring Correspondence: Towards Practical and Secure SQISign Signatures. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 659–690, Cham, 2023. Springer Nature Switzerland.

EPSV23. Jonathan Komada Eriksen, Lorenz Panny, Jana Sotáková, and Mattia Veroni. Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. Cryptology ePrint Archive, Paper 2023/106, 2023. Accepted in LuCaNT 2023.

FS16. Francesc Fité and Andrew V. Sutherland. Sato-Tate groups of $y^2 = x^8 + c$ and $y^2 = x^7 - cx$. In *Frobenius distributions: Lang-Trotter and Sato-Tate conjectures*, volume 663 of *Contemp. Math.*, pages 103–126. Amer. Math. Soc., Providence, RI, 2016.

Gal12. Steven D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012.

GPS17. Steven D Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*, pages 3–33. Springer, 2017.

Koh96. David Russell Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.

Sch87. René Schoof. Nonsingular plane cubic curves over finite fields. *Journal of Combinatorial Theory, Series A*, 46(2):183–211, 1987.

SEMR23. Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Krijn Reijnders. Aprèssqi: Extra fast verification for sqisign using extension-field signing. Cryptology ePrint Archive, Paper 2023/1559, 2023. https://eprint.iacr.org/2023/1559.

V́71. Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes Rendus de l'Académie des Sciences de Paris, A*, 273:238–241, 1971.

Wat69. William C Waterhouse. Abelian varieties over finite fields. In *Annales scientifiques de l'École normale supérieure*, volume 2, pages 521–560, 1969.

A Experimental parameters for constructive deuring correspondence

The primes used in Section 5 are given as follows.

$p_{3923} = 23759399264157352358673788613307970528646815114090876784643387662192449945599$
$p_1 = 11956566944641502957704189594909498993478297403838643406058180334130656750161$
$p_2 = 37670568336551536389503919665937491111216122470333837677213877442445311999999$
$p_3 = 88881583528687251695085351202716893361162950661419911309645115899579883741851$
$p_4 = 82460884298062985073154572827668830392815810118105762484859341606469999173287$
$p_5 = 95008112981120315885997172640930988762706307039235815007466698894954688225259$

For each prime, we also give the norm of the equivalent left ideal J. Our optimization focus on kernel point generations, so we provide the top three costly torsion groups during this procedure which was also mentioned in [EPSV23, Table 2].

- For prime p_{3923}, the top three costly torsion groups to work are $E_0[3^{68}] \in E_0(\mathbb{F}_{q^{27}})$, $E_0[109] \subseteq E_0(\mathbb{F}_{q^{27}})$ and $E_0[4733] \subseteq E_0(\mathbb{F}_{q^{26}})$.

$$\begin{aligned} N(J) = 2^{65} &\cdot 3^{66} \cdot 5^4 \cdot 7^2 \cdot 11^2 \cdot 13^2 \cdot 17^2 \cdot 19^2 \cdot 23 \cdot 29^2 \cdot 31 \cdot 37^2 \cdot 41 \cdot 43 \cdot 47 \cdot 53 \\ &\cdot 61 \cdot 67 \cdot 73 \cdot 79 \cdot 97 \cdot 101 \cdot 109 \cdot 127 \cdot 131 \cdot 139 \cdot 151 \cdot 157 \cdot 197 \cdot 239 \cdot 241 \\ &\cdot 263 \cdot 271 \cdot 281 \cdot 283 \cdot 307 \cdot 331 \cdot 397 \cdot 461 \cdot 521 \cdot 563 \cdot 599 \cdot 607 \cdot 619 \cdot 743 \\ &\cdot 827 \cdot 941 \cdot 1153 \cdot 1301 \cdot 2179 \cdot 2357 \cdot 2393 \cdot 3061 \cdot 3361 \cdot 3907 \cdot 3923 \cdot 4733 \\ &\cdot 8273 \cdot 9199 \cdot 9661 \cdot 10069 \cdot 10753 \cdot 11719 \cdot 12517 \cdot 17033 \cdot 26489 \cdot 58897 \\ &\cdot 62731 \cdot 107641. \end{aligned}$$

- For prime p_1, the top three costly torsion groups are $E_0[461] \subseteq E_0(\mathbb{F}_{q^{23}})$, $E_0[691] \subseteq E_0(\mathbb{F}_{q^{23}})$ and $E_0[13789] \subseteq E_0(\mathbb{F}_{q^{18}})$.

$$\begin{aligned} N(J) = 2^{5} &\cdot 3^{5} \cdot 5^2 \cdot 7^2 \cdot 11 \cdot 13^{19} \cdot 17 \cdot 19^2 \cdot 29^{18} \cdot 31 \cdot 37 \cdot 41 \cdot 43^{18} \cdot 47 \cdot 53 \cdot 61 \\ &\cdot 79 \cdot 89 \cdot 97 \cdot 101 \cdot 103 \cdot 113 \cdot 131 \cdot 137 \cdot 151 \cdot 157 \cdot 181 \cdot 193 \cdot 199 \cdot 239 \\ &\cdot 277 \cdot 281 \cdot 331 \cdot 401 \cdot 419 \cdot 421 \cdot 443 \cdot 457 \cdot 461 \cdot 541 \cdot 601 \cdot 617 \cdot 691 \cdot 739 \\ &\cdot 811 \cdot 919 \cdot 1621 \cdot 2473 \cdot 2741 \cdot 3299 \cdot 3373 \cdot 4049 \cdot 4933 \cdot 6823 \cdot 8609 \\ &\cdot 11953 \cdot 13789 \cdot 13921 \cdot 15467 \cdot 15679 \cdot 25969 \cdot 33161 \cdot 41681 \cdot 91837. \end{aligned}$$

- For prime p_2, the top three costly torsion groups are $E_0[859] \subseteq E_0(\mathbb{F}_{q^{33}})$, $E_0[1321] \subseteq E_0(\mathbb{F}_{q^{33}})$ and $E_0[409] \subseteq E_0(\mathbb{F}_{q^{34}})$.

$$\begin{aligned} N(J) = 2^{42} &\cdot 3^{5} \cdot 5^8 \cdot 7^2 \cdot 11^4 \cdot 13 \cdot 17 \cdot 19^4 \cdot 23 \cdot 29 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 47^3 \cdot 53 \\ &\cdot 59 \cdot 61 \cdot 67^6 \cdot 73 \cdot 79 \cdot 97 \cdot 101^3 \cdot 103 \cdot 113^3 \cdot 137^3 \cdot 139 \cdot 157 \cdot 181^2 \cdot 197 \\ &\cdot 223 \cdot 239 \cdot 241 \cdot 271 \cdot 277^3 \cdot 281 \cdot 307^3 \cdot 311 \cdot 349 \cdot 397 \cdot 409 \cdot 421^3 \cdot 449 \\ &\cdot 547 \cdot 691 \cdot 829 \cdot 859 \cdot 907 \cdot 919 \cdot 1013 \cdot 1103 \cdot 1171 \cdot 1321 \cdot 1597 \cdot 2341 \\ &\cdot 2647 \cdot 2777 \cdot 3271 \cdot 3739 \cdot 4513 \cdot 5419 \cdot 6091 \cdot 9007 \cdot 10267 \cdot 11981 \\ &\cdot 20641 \cdot 26083 \cdot 32957 \cdot 52627. \end{aligned}$$

- For prime p_3, before adjusting cost model, the top three costly torsion groups are $E_0[11161] \subseteq E_0(\mathbb{F}_{q^{45}})$, $E_0[2351] \subseteq E_0(\mathbb{F}_{q^{47}})$ and $E_0[1223] \subseteq E_0(\mathbb{F}_{q^{47}})$.

$$\begin{aligned} N(J) = {} & 2^4 \cdot 3^5 \cdot 5^4 \cdot 11 \cdot 13^2 \cdot 17^2 \cdot 19 \cdot 23 \cdot 29 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 47 \cdot 53 \cdot 59 \cdot 61 \\ & \cdot 67 \cdot 71 \cdot 73 \cdot 79 \cdot 83 \cdot 89 \cdot 97 \cdot 101 \cdot 103 \cdot 109 \cdot 113 \cdot 137 \cdot 149 \cdot 157 \cdot 173 \cdot \\ & 181 \cdot 191 \cdot 193 \cdot 197 \cdot 199 \cdot 239 \cdot 241 \cdot 251 \cdot 257 \cdot 271 \cdot 311 \cdot 353 \cdot 397 \cdot 431 \\ & \cdot 463 \cdot 521 \cdot 601 \cdot 631 \cdot 677 \cdot 769 \cdot 953 \cdot 1063 \cdot 1117 \cdot 1223 \cdot 1301 \cdot 1453 \cdot \\ & 1621 \cdot 1783 \cdot 1801 \cdot 2351 \cdot 2521 \cdot 2857 \cdot 4673 \cdot 5581 \cdot 6043 \cdot 7937 \cdot 8087 \cdot \\ & 9103 \cdot 9829 \cdot 10729 \cdot 11161 \cdot 12161 \cdot 13441 \cdot 17209 \cdot 17807 \cdot 29017 \cdot 46901 \\ & \cdot 47269 \cdot 47441 \cdot 77969 \cdot 85021 \cdot 89839 \cdot 180503. \end{aligned}$$

- For prime p_4, before adjusting cost model, the top three costly torsion groups are $E_0[1723] \subseteq E_0(\mathbb{F}_{q^{41}})$, $E_0[1559] \subseteq E_0(\mathbb{F}_{q^{41}})$ and $E_0[3361] \subseteq E_0(\mathbb{F}_{q^{40}})$.

$$\begin{aligned} N(J) = {} & 2^5 \cdot 3^4 \cdot 5^2 \cdot 7^2 \cdot 13^2 \cdot 17 \cdot 19 \cdot 23 \cdot 29 \cdot 31 \cdot 37^2 \cdot 41 \cdot 43 \cdot 47 \cdot 53 \cdot 59 \cdot 61 \\ & \cdot 67 \cdot 71 \cdot 73 \cdot 79 \cdot 83 \cdot 89 \cdot 97 \cdot 101 \cdot 109 \cdot 113 \cdot 127 \cdot 139 \cdot 157 \cdot 163 \cdot 181 \cdot \\ & 193 \cdot 229 \cdot 233 \cdot 239 \cdot 241 \cdot 277 \cdot 311 \cdot 379 \cdot 397 \cdot 401 \cdot 409 \cdot 463 \cdot 677 \cdot 769 \\ & \cdot 859 \cdot 877 \cdot 881 \cdot 937 \cdot 1217 \cdot 1301 \cdot 1321 \cdot 1423 \cdot 1489 \cdot 1559 \cdot 1597 \cdot 1723 \\ & \cdot 1873 \cdot 1973 \cdot 2377 \cdot 2887 \cdot 3361 \cdot 3461 \cdot 3499 \cdot 3571 \cdot 3697 \cdot 3769 \cdot 3877 \cdot \\ & 4231 \cdot 4993 \cdot 5023 \cdot 5437 \cdot 5669 \cdot 5881 \cdot 6211 \cdot 6449 \cdot 6781 \cdot 8317 \cdot 8677 \cdot \\ & 10501 \cdot 12757 \cdot 15227 \cdot 19441 \cdot 19793 \cdot 29389 \cdot 64577. \end{aligned}$$

- For prime p_5, the top three costly torsion groups are $E_0[83] \subseteq E_0(\mathbb{F}_{q^{41}})$, $E_0[2029] \subseteq E_0(\mathbb{F}_{q^{39}})$ and $E_0[859] \subseteq E_0(\mathbb{F}_{q^{39}})$.

$$\begin{aligned} N(J) = {} & 2^4 \cdot 3^4 \cdot 5^3 \cdot 7^3 \cdot 11^2 \cdot 13 \cdot 17 \cdot 19 \cdot 23 \cdot 29 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 47 \cdot 53 \cdot 59 \\ & \cdot 61 \cdot 67 \cdot 71 \cdot 73 \cdot 79 \cdot 83 \cdot 89 \cdot 97 \cdot 101 \cdot 103 \cdot 109 \cdot 113 \cdot 127 \cdot 131 \cdot 137 \cdot \\ & 151 \cdot 157 \cdot 193 \cdot 199 \cdot 223 \cdot 229 \cdot 233 \cdot 239 \cdot 257 \cdot 349 \cdot 409 \cdot 541 \cdot 601 \cdot 631 \\ & \cdot 661 \cdot 701 \cdot 727 \cdot 751 \cdot 859 \cdot 1009 \cdot 1213 \cdot 1471 \cdot 1489 \cdot 1601 \cdot 1657 \cdot 2029 \\ & \cdot 2341 \cdot 2441 \cdot 2609 \cdot 2801 \cdot 3361 \cdot 3559 \cdot 4159 \cdot 4831 \cdot 5077 \cdot 7193 \cdot 9649 \cdot \\ & 9929 \cdot 13441 \cdot 13907 \cdot 13913 \cdot 14293 \cdot 21937 \cdot 22573 \cdot 27961 \cdot 38851 \cdot 52667 \\ & \cdot 59621 \cdot 69193 \cdot 162499 \cdot 170047 \cdot 208141 \cdot 288493. \end{aligned}$$