UNIVERSITÉ DE VERSAILLES SAINT-QUENTIN-EN-YVELINES

Laboratoire de Mathématiques de Versailles LMV – UMR 8100

HABILITATION À DIRIGER DES RECHERCHES

EN INFORMATIQUE

Présentée par

LUCA DE FEO

EXPLORING ISOGENY GRAPHS

Soutenue le 14 décembre 2018

Devant le jury composé de

Andreas Enge	Inria Bordeaux Sud-Ouest	(Rapporteur)
Florian Hess	Carl von Ossietzky Universität Oldenburg	(Rapporteur)
David Kohel	Aix-Marseille Université	(Rapporteur)
Paulo Barreto	University of Washington Tacoma	(Examinateur)
Jean-Marc Couveignes	Université de Bordeaux	(Examinateur)
Annick Valibouze	Université Pierre et Marie Curie	(Examinatrice)
Louis Goubin	Université Paris Saclay – UVSQ	(Tuteur)

Original version: December 14, 2018.

This document was typeset using LaTeX, TikZ, biber, the class Memoir and many LaTeX packages. The source code is available for download at https://github.com/defeo/hdr/.

Preface

About ten years removed from my PhD, the time has come to pause and reflect on my research path. But, even admitting I have come to a point where I can look back at my work and make sense of it, how to convey this in a coherent manuscript that still has some value to a reader?

I don't know if I found the answer, at any rate I could not get a clear one by asking around. I guess this is part of the exercise. I chose to write three chapters that survey topics I particularly endear, in the hope that they motivate readers to get into similar areas of research. You will judge if I have succeeded.

As a last thing before moving to the main course, I would like to leave a message in a bottle for future HDR candidates: everyone knows what an HDR is not, but no one seems to know what it is. You shall follow the advice of previous candidates, of course, but in the end you will decide what you make of it, so try and make what is most valuable to you and your community.

Akcnowledgments. I would like to start by thanking the person who is responsible for getting me into all of this. He has been a mentor, a coworker, a friend, a paternal figure, and, above all, he was the one crazy enough to believe, back in 2007, that isogeny based cryptography may be a serious thing one day. I am obviously naming François Morain. Who knows whether isogeny based cryptography would have been ready on time for NIST's post-quantum competition without his vision.

The next person, is the one responsible for keeping me from falling completely into this. She is my best friend, an amazing chef, a talented artist, a charismatic teacher, and a loving presence. Without her, I could not have taken the load of stress and fatigue that comes with preparing this thesis. I am talking of Rachel Deyts.

I want to specially thank my research students: C. Hugounenq, S. Besnier, L. Brieulle, É. Rousseau, R. Larrieu, J. Kieffer and M. Veroni. In the end, this manuscript is about them, and the wonderful moments we have spent working together. I am also grateful to all my other students for everything they have given to me, but I can't possibly fit all their names in here.

My coauthors, É. Schost, D. Jao, J. Doliskani, J. Plût, J.-P. Flori, M. Kohlhase, D. Müller, M. Pfeiffer, N. Thiéry, V. Vasilyev, T. Wiesing, B. Smith, S. Galbraith, R. Azarderakhsh, B. Koziel, M. Campagna, B. LaMacchia, C. Costello, P. Longa, M. Naehrig, B. Hess, J. Renes, A. Jalali, V. Soukharev and D. Urbanik, also deserve a special thanks. Research is no fun alone in an office. Even if we are not technically coauthors, I have spent hours working on things that I value as much as pure research with E. Bray, J. Demeyer and V. Delecroix, and I am grateful for that.

A special thank to Louis Goubin, who is the ideal boss, and whose advice for preparing this manuscript and the defense has been priceless.

I am honored and humbled that A. Enge, F. Hess and D. Kohel have accepted to review this manuscript, and that P. Barreto, J.-M. Couveignes and A. Valibouze have agreed to serve on the jury. They are among my role models, and it means a lot to me that they have looked with interest to my work.

I am immensely grateful for the working and leisure time I have spent with my coworkers at UVSQ, at Inria Saclay, at Télécom Paristech and at Paris Sud: C. Boura, C. Chaigneau, I. Chilotti, N. Gama, M. Krir, A. Mathieu-Mahias, G. Moreno-Socias, J. Patarin, N. Perrin, M. Quisquater, V. Sécherre, F. Vial-Prado, D. Augot, E. Barelli, A. Couvreur, V. Ducet, J. Lavauzelle, F. Lévy, B. Meyer, D. Madore, J. Pieltant, M.

Rambaud, H. Randriam, F. Hivert, S. Lelièvre, B. Pilorget, V. Pons, and I am sure I am forgetting students who have come and gone.

My life would have been much harder without the invaluable help of our assistants, C. Le Quéré, I. Moudenner, C. Ducoin, and F. Chevalier.

I want to thank my family for always having been supportive of my life choices, and for being always ready to come all the way from Italy to support me in these moments. My friends, for cheering me up and giving me moments of pleasure away from work.

And finally, thanks to you who are reading this. Your interest in these pages honors me deeply.

Contents

Introduction

If you are reading this on a computer screen, chances are that you got to it by browsing the web. You probably went to a search engine, or maybe received a link via email, or social media. You followed the link, landing on some scientific repository, and downloaded the pdf.

Two to three hops, each triggering dozens of connections to search servers, CDNs for static assets, targeted advertisement services, and trackers of all sorts. All over HTTPS.

In the lapse of a few seconds, your CPU has chosen some standardized elliptic curves, drawn dozens of random integers, multiplied some default generators by them, and sent around projective points over the wire. Maybe it is even the year 2050, and, as we have all moved to Siberia to escape the effects of global warming, your Megacorp branded browser is now offering perfect forward secrecy through ephemeral supersingular isogeny key exchange.

Yes. *Supersingular isogeny key exchange*. Indeed, this may sound straight out of a William Gibson novel, but it actually is a real thing. And you do not even need to wait for Netherlands to be under water to use it: Microsoft has released a fork of OpenVPN containing it.[1]

The goal of this document is to make those four words sound less otherworldly, at least for those who have been around asymmetric cryptography in recent years. It is not a course on arithmetic geometry, nor a complete review of isogeny-based cryptography, not even a monography. It is more like a promenade, a stroll around topics related to *isogeny graphs* that are dearest to my heart. A journey through unexplored lands made possible by science and technological progress. Like mathematical *capitaines Nemos* we shall start our journey on an isolated island, an *elliptic curve* surfacing out of an uncharted sea.

We shall start our exploration by diving in the sea. We will discover nearby underwater elliptic curves, linked to our island by *isogenies*. We will enter our *Nautilus*, and, equipped with an *algebraic bathymeter*, we will dive to the sea floor to explore the slopes of an underwater *volcano*.

Next, we shall climb to the top of the island to gain a vantage point and observe the seascape around us. We will build observation *towers* to look as far as the remotest islets, discovering that our tiny island is only a minuscule part of an immense archipelago: an *isogeny graph* crisscrossed by isogenies of any *degree*.

Finally, we shall set sail to explore the archipelago, charting the isogeny routes that link the elliptic curves. The theories of *complex multiplication* and of *quaternion algebras* will work as a compass, indicating the direction to the next island. However, despite our technological prowess, like seamen in a starless night, we will miss a fundamental tool: a telemeter to keep track of the distance between elliptic curves. A blessing and a curse, the lack of a telemeter will let us hide *secrets* in the isogeny graph, confident that any pirate seeking them will be condemned to wander aimlessly through the archipelago for centuries to come.

Breaking out of the metaphors, Chapter I deals with isogenies of elliptic curves and algorithmic problems related to them. We will introduce the Explicit isogeny problem, first studied by Elkies, and present some algorithms to solve it. We will then focus on elliptic curves over finite fields, and on a specific algorithm for the explicit isogeny problem due to Couveignes. To understand it better, we will study the *Frobenius*

[1] https://github.com/Microsoft/PQCrypto-VPN.

endomorphism, and see how it determines the types of isogenies around an elliptic curve; by looking at its action on the *Tate module*, we will gain a global view on the *isogeny graph*. Then, an *effective version of Tate's isogeny theorem* will provide us with an effective way to *probe the depths* of the isogeny graph. Armed with this tool, we will end the chapter with a generalization of Couveignes' algorithm, currently the algorithm with the best complexity for solving the explicit isogeny problem.

The effective version of Tate's theorem is only as efficient as the algorithms at our disposal to compute in the algebraic closure of a finite field. In Chapter II we shall study algorithms to represent and compute with finite extensions of a finite field. We will review algorithms to compute irreducible polynomials, then move to two radically different paradigms to represent the algebraic closure of finite fields. One, based on *special families of irreducible polynomials*, will extend some algorithms for irreducible polynomials by adding to them more features: *compatibility*, *incrementality* and *uniqueness*. The other one, based on *lattices of arbitrary irreducible polynomials*, will be founded on an algorithm for computing isomorphisms of finite fields; we shall thus review all known algorithms for this problem, and see how they are related to algorithms for irreducible polynomials.

The goal of this chapter is not only to be a review in computational complexity, but also to explore the practical implementation aspects of the algorithms. All along the exposition, we will refer to the available implementations in the most popular computer algebra systems and libraries (Magma, SageMath, PARI/GP, Nemo, Flint, NTL, ...), and highlight the implementation challenges and possible ways forward.

Finally, in Chapter III we will come to the much anticipated isogeny-based cryptography. This novel type of cryptography, pioneered by Couveignes in the nineties, is built on the algebraic structure of large isogeny graphs. We will see how the theory of *complex multiplication* and that of *quaternion algebras* determine the structure of these graphs, and how they prove their *expansion* properties. We will then focus, only, on key exchange protocols based on isogenies graphs; we will review three proposals: Couveignes' original one based on *ordinary graphs*, a recent twist on it, named CSIDH, based on *supersingular* $\mathbb{F}_p$*-graphs*, and another one based on generic *supersingular graphs* named SIDH. We will conclude the chapter by discussing security of isogeny-based primitives. The main selling point for isogeny-based algorithms is their supposed resistance to quantum attacks, we shall thus review the known quantum algorithms for breaking them, and discuss the impact on security parameters.

The whole document is meant as an introduction to a research area, thus it purposely ignores some important topics and skips technical details. Each chapter is accompanied by one or two research articles, previously appeared in peer reviewed journals and proceedings, for the reader interested in gaining more insights. These are, for Chapter I,

> Luca De Feo, Cyril Hugounenq, Jérôme Plût and Éric Schost. "Explicit isogenies in quadratic time in any characteristic". *LMS Journal of Computation and Mathematics* [De +16].

Introducing the generalization of Couveignes' algorithm. For Chapter II,

> Luca De Feo, Javad Doliskani and Éric Schost. "Fast Arithmetic for the Algebraic Closure of Finite Fields". *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation (ISSAC 2014)* [DDS14].

On realizing the algebraic closure of a finite field, and

> Ludovic Brieulle, Luca De Feo, Javad Doliskani, Jean-Pierre Flori and Éric Schost. “Computing Isomorphisms and Embeddings of Finite Fields”. *Mathematics of Computation* [Bri+18].

On the complexity and the practical performance of isomorphism algorithms for finite fields. For Chapter III,

> Luca De Feo, David Jao and Jérôme Plût. “Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies”. *Journal of Mathematical Cryptology* [DJP14].

introducing the key exchange protocol SIDH, and

> Luca De Feo, Jean Kieffer and Benjamin Smith. “Towards practical key exchange from ordinary isogeny graphs”. *Proceedings of AsiaCrypt 2018* [DKS18].

on improvements to the Couveignes–Rostovtsev–Stolbunov key exchange protocol.

Finally, this document also wants to serve as a snapshot of the current state of the research in the various areas it touches. For this reason, each chapter is terminated by a section called “Perspectives”, pointing to interesting related research topics, both easy and hard.

I hope that you will appreciate the topics I have selected, enjoy the flow of the presentation, and forgive me for omissions and approximations.

I Bathymetry

What is an isogeny, anyway? Despite the imposing sounding name, an isogeny is a pretty simple concept: a morphism between two elliptic curves, preserving their algebraic structure (both as a group and as a variety).

Isogenies of abelian varieties have been studied since the beginning of the 19th century, by the likes of Abel, Jacobi, Weierstrass, Riemann, Picard, etc. According to Wikipedia[1], the name "isogeny" was introduced in the 20th century by André Weil, to avoid confusion with "isomorphism". After the *schematic* revolution in algebraic geometry, major contributions to the theory of abelian varieties and isogenies were made by Cartier, Serre, Tate, and, obviously, Grothendieck. With the development of computer algebra, people grew increasingly interested in effective methods, with important contributions being made by Schoof, Atkin, Elkies, Satoh, Kedlaya, and many others. In recent years, isogenies have found various applications in cryptology, sparking a remarkable wave of results on their algorithmic properties.

In this chapter, we shall take the algorithmic point of view, and discuss algorithms to compute and classify isogenies of elliptic curves. The focal point of interest will be the *Frobenius endomorphism* of an elliptic curve defined over a finite field. We will first learn how it governs the properties of isogenies "in the neighborhood" of an elliptic curve. We will then define *isogeny graphs*—graphs whose vertices are elliptic curves and whose edges are isogenies—, and, not unlike a biologist, set on a mission to classify them. Isogeny graphs inherit from an *infinite tree* structure, but over a finite field they must "fold" in order to fit into a finite space. Thanks to a celebrated theorem of Tate, we will discover that the Frobenius endomorphism, much like the DNA of a living creature, determines the "folding" of the isogeny graph.

Even knowing the structure of an isogeny graph, it is not always easy to navigate it. An *effective version* of Tate's theorem will provide us with a tool to "probe the depth" of a curve inside an isogeny graph. Armed with our brand new tool, we will conclude the chapter by presenting the algorithm with the best known complexity to compute an isogeny between two given elliptic curves.

I.1 Isogenies

An isogeny is a non-constant algebraic map between elliptic curves, preserving the point at infinity. An isogeny is also a surjective group morphism of elliptic curves. It turns out these definitions are equivalent, but, before getting these pages drenched in more properties and theorems, let's have a look at an example.

The map ϕ from the elliptic curve $y^2 = x^3 + x$ to $y^2 = x^3 - 4x$ defined by

$$\begin{aligned} \phi(x,y) &= \left(\frac{x^2+1}{x}, y\frac{x^2-1}{x^2} \right), \\ \phi(0,0) &= \phi(\mathscr{O}) = \mathscr{O} \end{aligned} \tag{I.1}$$

is an isogeny. As an algebraic map it has degree 2, which implies that it is a two-to-one map, as it can be inferred from the polynomial degrees.

What does an isogeny "look like"? Drawing the above one in $\mathbb{R}^2$ would look rather messy, but an isogeny defined over the rationals is still an isogeny if we reduce modulo a prime p. Figure I.1 plots the action of the isogeny (I.1) on the image of the curves

[1] See https://en.wikipedia.org/wiki/Isogeny. No source is given, though.

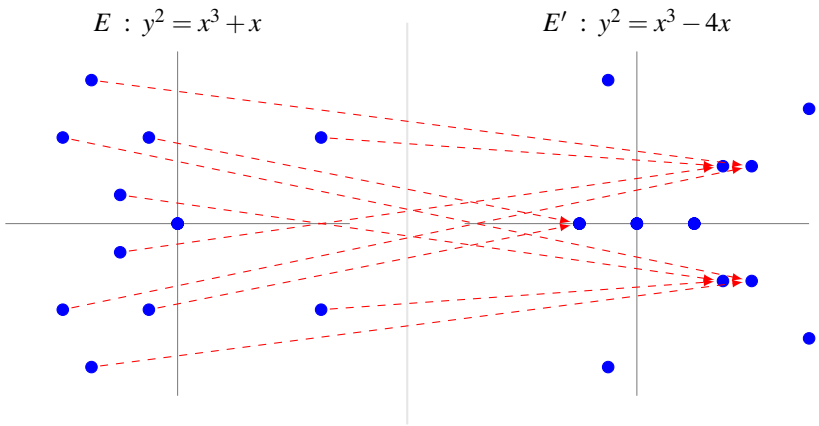


Figure I.1: The isogeny $(x,y) \mapsto \left((x^2+1)/x,\ y(x^2-1)/x^2\right)$, as a map between curves defined over $\mathbb{F}_{11}$.

in $\mathbb{F}_{11}$. A red arrow indicates that a point of the left curve is sent onto a point on the right curve; the action on the point in $(0,0)$, going to the point at infinity, is not shown. We observe a symmetry with respect to the x-axis, a consequence of the fact that ϕ is a group morphism; and, by looking closer, we may also notice that collinear points are sent to collinear points, also a necessity for a group morphism.

Something strikes us, though: the map looks by no means surjective! This is because, when we think of isogenies, we think of them as geometric objects, acting on the extension of the curves to the algebraic closure. This is not dissimilar from the way power-by-n maps act on the multiplicative group $k^\times$ of a field k: the map $x \mapsto x^2$, for example, is a two-to-one (algebraic) group morphism on $\mathbb{F}_{11}^\times$, and those elements that have no preimage, the non-squares, will have exactly two square roots in $\mathbb{F}_{11^2}$, and so on. In much the same way, in an algebraic closure $\bar{\mathbb{F}}_{11}$ of $\mathbb{F}_{11}$, the isogeny ϕ becomes surjective and every point gains exactly two antecedents. This analogy is more profound that it may seem, and shall bear its fruits in Chapter II.

For elliptic curves defined over a field of characteristic $p > 0$, there is another kind of isogeny. Let $E : y^2 = x^3 + ax + b$ be an elliptic curve, let q be a power of p, and let $E^{(q)} : y^2 = x^3 + a^q x + b^q$. The isogeny $\pi_q : E \to E^{(q)}$ defined by

$$\begin{aligned} \pi_q(x,y) &= (x^q, y^q), \\ \pi_q(\mathcal{O}) &= \mathcal{O} \end{aligned} \tag{I.2}$$

is a *purely inseparable* isogeny of degree q. We call π_q a *Frobenius isogeny*. Despite being of degree q, Frobenius isogenies have trivial kernel, and are one-to-one over finite fields (and other perfect fields).

Any isogeny can be decomposed as a product of a Frobenius isogeny and a *separable* isogeny:

$$E \xrightarrow{\pi_q} E^{(q)} \xrightarrow{\phi_s} E' \qquad E \xrightarrow{\phi} E'$$

Computing this decomposition is also easy given rational functions for ϕ: simply factor out the powers of p from the polynomials. For these reasons we shall be mostly concerned with separable isogenies and their computations.

The most unique property of separable isogenies is that they are entirely determined by their kernel.

Proposition I.1. *Let E be an elliptic curve defined over an algebraically closed field, and let G be a finite subgroup of E. There is a curve E', and a separable isogeny ϕ, such that* $\ker\phi = G$ *and* $\phi : E \to E'$. *Furthermore, E' and ϕ are unique up to composition with an isomorphism $E' \simeq E''$.*

Said otherwise, for any finite subgroup $G \subset E$, we have an exact sequence of algebraic groups

$$0 \longrightarrow G \longrightarrow E \xrightarrow{\phi} E' \longrightarrow 0.$$

Uniqueness up to isomorphisms justifies the notation E/G for the isomorphism class of the image curve E'. Now, it would be useful if we could find a way to define a canonical representative inside E/G. It turns out there is a pretty natural way to define one.

Definition I.2 (Normalized isogeny)**.** Let E, E' be two elliptic curves, ω_E, ω'_E their *invariant differential*, $\phi : E \to E'$ a separable isogeny and $\phi^* : \Omega_{E'} \to \Omega_E$ its *pullback*. We say that ϕ is *normalized* if its pullback preserves the invariant differentials, i.e., $\phi^*(\omega_{E'}) = \omega_E$.

Since ϕ is separable, ϕ^* is an isomorphism of vector spaces of dimension 1. Said otherwise, if ϕ is not normalized, then it is only "off" by a (non-zero) constant $\phi^*(\omega_{E'}) = c\omega_E$, and we can easily normalize ϕ by a change of variables. This also shows that, for fixed E and $\ker\phi$, the normalized isogeny is unique, and justifies abusing the notation E/G to mean the image of the normalized isogeny with kernel G.[2]

Conversely, since any non-constant morphism of elliptic curves necessarily has finite kernel, we have a canonical bijection between the finite subgroups of a curve E and the normalized isogenies with domain E. This correspondence is rich in consequences: it is an easy exercise to prove the following useful facts.

Corollary I.3.

1. *Any isogeny of elliptic curves can be decomposed as a product of prime degree isogenies.*
2. *Let E be defined over an algebraically closed field k, let ℓ be a prime different from the characteristic of k, then there are exactly $\ell + 1$ normalized isogenies of degree ℓ with domain E.*

Slightly more work is required to prove the following, fundamental, theorem (the difficulty comes essentially from the inseparable part, see [Sil92, p. III.6.1] for a detailed proof).

Theorem I.4 (Dual isogeny theorem)**.** *Let $\phi : E \to E'$ be an isogeny of degree m. There is a unique isogeny $\hat{\phi} : E' \to E$ such that*

$$\hat{\phi} \circ \phi = [m]_E, \quad \phi \circ \hat{\phi} = [m]_{E'}.$$

$\hat{\phi}$ is called the dual isogeny of ϕ*; it has the following properties:*

[2] Note that this convention is not universal in the literature, as there are other useful choices for a canonical representative of E/G.

1. $\hat{\phi}$ *has degree* m*;*
2. $\hat{\phi}$ *is defined over* k *if and only if* ϕ *is;*
3. $\widehat{\psi\circ\phi} = \hat{\phi}\circ\hat{\psi}$ *for any isogeny* $\psi : E' \to E''$*;*
4. $\widehat{\psi+\phi} = \hat{\psi}+\hat{\phi}$ *for any isogeny* $\psi : E \to E'$*;*
5. $\deg\phi = \deg\hat{\phi}$*;*
6. $\hat{\hat{\phi}} = \phi$.

Note that, since $[m]^*(\omega) = m\omega$, if ϕ is normalized so that $\phi^*\omega' = \omega$, $\hat{\phi}$ almost never is. The computational counterpart to the kernel-isogeny correspondence is given by Vélu's much celebrated formulas.

Proposition I.5 (Vélu [Vél71])**.** *Let* $E : y^2 = x^3 + ax + b$ *be an elliptic curve defined over a field* k*, and let* $G \subset E(\bar{k})$ *be a finite subgroup. The normalized separable isogeny* $\phi : E \to E/G$*, of kernel* G*, can be written as*

$$\phi(P) = \left(x(P) + \sum_{Q\in G\setminus\{\mathcal{O}\}} x(P+Q) - x(Q), y(P) + \sum_{Q\in G\setminus\{\mathcal{O}\}} y(P+Q) - y(Q) \right);$$

and the curve E/G *has equation* $y^2 = x^3 + a'x + b'$*, where*

$$a' = a - 5\sum_{Q\in G\setminus\{\mathcal{O}\}} (3x(Q)^2 + a),$$
$$b' = b - 7\sum_{Q\in G\setminus\{\mathcal{O}\}} (5x(Q)^3 + 3ax(Q) + b).$$

But how "good" are Vélu's formulas from a computational perspective? "Pretty good", is the message we want to convey, but, in order to understand the question, we need to discuss *rationality*. Let E be defined over a field k with algebraic closure $\bar{k}$. We say that an isogeny $\phi : E \to E'$ is *defined over* k, or *k-rational*, if ϕ is invariant under the action of the Galois group of $\bar{k}/k$. This is equivalent to ϕ being defined by rational maps with coefficients in k, and implies[3] that $\ker\phi$ is stable under $\mathrm{Gal}(\bar{k}/k)$. It implies that E' is defined over k, but the converse is not true.

In the rest of this work, when we say "an isogeny", we really mean "an isogeny defined over the base field", unless specified otherwise. What are the input and output sizes of Vélu's formulas, if we restrict to k-rational isogenies?

The output is a pair of rational fractions, and, letting $\ell = \#G$, it is not too difficult to see that they have $O(\ell)$ coefficients. The input is the kernel G, and, since it is finite, it must be generated by at most two elements. However G is only stable under $\mathrm{Gal}(\bar{k}/k)$, implying that its elements are defined in an (abelian) extension of degree in $O(\ell)$. Thus, in general, G is represented by $O(\ell)$ coefficients of k.

Now, if we apply mindlessly Vélu's formulas, we need at least $O(\ell^2)$ coefficients in k to write down all the elements of G. A better approach is to represent G by a polynomial with coefficients in k that vanishes on all $P \in G$, for example

$$h_G(x) = \prod_{P\in G\setminus\{\mathcal{O}\}} (x - x(P)).$$

[3] The converse is only true up to *twist*.

The polynomial h_G is called the *kernel polynomial*[4] of G, and has coefficients in k if and only if ϕ is k-rational; it can be computed from the generators of G in only $\tilde{O}(\ell)$ operations over k. Following Kohel [Koh96], we can rewrite Vélu's formulas in terms of h_G, then, their evaluation can also be accomplished in $\tilde{O}(\ell)$ operations.

In conclusion, we see that Vélu's formulas make the kernel/isogeny correspondence explicit, using a quasi-optimal number of operations in general. This will be crucial when we will study isogeny-based cryptosystems in Chapter III, however, we will encounter there some examples where the cost of evaluating an isogeny is exponentially lower than that of Vélu's formulas.

I.2 The explicit isogeny and other problems

When it comes to computations, Vélu's formulas are only part of the story: how do we find a rational kernel G in the first place? Elkies, while working on point counting [Elk92; Elk98], famously baptized this the *explicit isogeny problem*.

Problem I.6 (Explicit isogeny problem)**.** Let E be an elliptic curve, and let ℓ be an integer. Find, if it exists, an isogeny of degree ℓ with domain E.

A slightly modified version of the same problem is often found in the literature.

Problem I.7. Let E and E' be two elliptic curves, and ℓ an integer. Decide whether there exists an isogeny $\phi : E \to E'$ of degree ℓ, and compute its kernel.

The many variants of the explicit isogeny problem have kept the research community busy for more than twenty years, and still do today. Let's have a closer look at it.

Elkies' algorithm. For a start, it should be noticed that both variants are by no means "hard". Indeed, we have explicit formulas for adding points on a curve E, from which we can deduce an explicit formula for multiplying points on E by any scalar $\ell \in \mathbb{Z}$. Said otherwise, we have an explicit formula for the multiplication-by-ℓ isogeny, and, by reading its denominators, we can deduce its kernel polynomial h_ℓ.[5]

Let us assume for simplicity that ℓ is prime and different from the characteristic, then we know there are at most $\ell+1$ normalized isogenies of degree ℓ from E. Factoring h_ℓ over the field of definition k of E lets us compute all possible kernel polynomials of order ℓ, and thus all possible isogenies. At most $\ell+1$ applications of Vélu's formula will then give the answer to either of the two variants of the explicit isogeny problem. Since $E[\ell] \simeq (\mathbb{Z}/\ell\mathbb{Z})^2$ over the algebraic closure, h_ℓ has degree ℓ^2-1, thus this algorithm costs no more than factoring a degree $O(\ell^2)$ polynomial in $k[x]$.

We see that the matter is not solving the explicit isogeny problem. The matter is solving it fast!

An interesting case, and the one Elkies was originally interested in, is when the curves are defined over a finite field. Let us relax the problem a bit, and see what can be told about it. Decisional versions, first: two elliptic curves are said to be *isogenous* if there exists an isogeny connecting them (this is an equivalence relation, thanks to the dual isogeny theorem).

[4] Other works prefer defining the kernel polynomial as the square root of h_G, however this adds some complications when $\#G$ is even.

[5] Up to a constant, h_ℓ is the square of ψ_ℓ, the *ℓ-th division polynomial*. See [BSS99, p. III.4] for explicit formulas.

Problem I.8. Let E, E' be two elliptic curves defined over a finite field $\mathbb{F}_q$, decide whether they are isogenous.

Tate [Tat66, Th. 1(c)] famously showed[6] that E and E' are isogenous over $\mathbb{F}_q$ if and only if $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$. Schoof's point counting algorithm [Sch85; Sch95] completely settles the problem by computing the orders of E and E' in polynomial time in $\log q$. However, when we add a degree constraint on the isogeny, the problem immediately becomes harder, even for finite fields.

Problem I.9. Let E, E' be two elliptic curves, and ℓ be an integer. Decide whether they are ℓ-isogenous.

The *modular polynomial* helps solve this problem. Assuming ℓ is prime, the ℓ-th modular polynomial, denoted by $\Phi_\ell(x,y)$, is a bivariate polynomial with coefficients in $\mathbb{Z}$, symmetric in x and y, of degree $\ell+1$ in each variable, with the following property: two elliptic curves E, E' are ℓ-isogenous if and only if $\Phi_\ell(j(E), j(E')) = \Phi_\ell(j(E'), j(E)) = 0$. We stress that the definition of Φ_ℓ is independent of the base field. Given that Φ_ℓ has $O(\ell^2)$ coefficients (and rather large ones), using it to decide the explicit isogeny problem is asymptotically only slightly better than factoring the division polynomial; it is however usually considerably more efficient in practice, especially when tables of modular polynomials are precomputed, as is the case in computer algebra systems such as Pari [PARI], Magma [BCP97], or SageMath [Sage].

The modular polynomial can also be used to produce all isogenous elliptic curves, up to isomorphism, to a given curve: simply plug $j(E)$ in Φ_ℓ, then factor $\Phi_\ell(j(E), y)$ to find the isogenous j-invariants. Elkies used this approach to reduce the explicit isogeny problem to Problem I.7, but he managed to extract even more information from Φ_ℓ: he showed how to obtain a *normalized equation* for the image curve.

Theorem I.10 ([Elk92; Sch95; Elk98])**.** *Let $E : y^2 = x^3 + ax + b$ be an elliptic curve, let j be its j-invariant and let j' be such that $\Phi_\ell(j, j') = 0$. Assume that $(\partial\Phi_\ell/\partial y)(j, j') \neq 0$, and define*

$$\begin{aligned} \lambda &= \frac{-18}{\ell}\frac{b}{a}\frac{\frac{\partial\Phi_\ell}{\partial x}(j,j')}{\frac{\partial\Phi_\ell}{\partial y}(j,j')}j, \\ a' &= -\frac{1}{48}\frac{\lambda^2}{j'(j'-1728)}\frac{1}{\ell^4}, \\ b' &= -\frac{1}{864}\frac{\lambda^3}{(j')^2(j'-1728)}\frac{1}{\ell^6}. \end{aligned} \tag{I.3}$$

Then there is a normalized isogeny of degree ℓ from E to $E' : y^2 = x^3 + a'x + b'$.

Elkies' theorem prompts us to define a weaker variant of the explicit isogeny problem.

Problem I.11 (Inverse Vélu problem[7])**.** Let E, E' be elliptic curves such that there exists a normalized isogeny $\phi : E \to E'$ of degree ℓ. Compute the kernel of ϕ.

[6] Tate, citing Mumford, also points out that, for the case of elliptic curves, this is an easy consequence of the much celebrated work of Deuring [Deu41].

[7] The name is ours and not attested in the literature.

Unsurprisingly, Elkies also gave the solution to this problem in [Elk92; Elk98]. He observed that the rational fractions defining ϕ are related by a differential equation, involving only the coefficients of E and E'. Solving the differential equation gives the rational fractions, and thus the kernel. This gives a method to solve the inverse Vélu problem in $O(\ell^2)$ operations over the base field, or even $\tilde{O}(\ell)$ using computer algebra techniques as suggested by Bostan, Morain, Salvy and Schost [Bos+08].

We have, essentially, sketched the computation involved in the Schoof-Elkies-Atkin (SEA) point counting algorithm [Sch95], for those that are called *Elkies primes* (more on these later). However, the last part of Elkies' algorithm, the solution to the inverse Vélu problem, only works when the characteristic is 0 or *large enough*. While this is good enough for counting points of elliptic curves defined over a prime field $\mathbb{F}_p$, it fails, for example, over binary fields.

Couveignes' algorithm. After Elkies, others set out to solve the explicit isogeny problem in small characteristic. While Elkies' method is grounded in complex analysis, and thus naturally works in characteristic 0, Couveignes [Cou94; Cou96] and Lercier [Ler96] introduced "more algebraic" methods, that only work over finite fields.

The one that shall interest us here is Couveignes' second method: a strikingly simple idea to solve Problem I.7 directly. It is based on the observation that any isogeny $\phi : E \to E'$ must preserve Sylow subgroups:

$$\phi(E[r^k]) \subseteq E'[r^k] \quad \text{for any prime } r \text{ and } k \geq 0, \tag{I.4}$$

with equality if r does not divide $\deg \phi$. If $E/\mathbb{F}_{p^n}$ is an ordinary curve, $E[p^k] \simeq \mathbb{Z}/p^k\mathbb{Z}$ has a particularly simple structure. The idea is to compute $E[p^k]$ and $E'[p^k]$ for k large enough (precisely, $p^k \sim 4\deg\phi$), make a guess for the exact image of one group into the other, and *interpolate* the isogeny. If the guess was right, the computed isogeny can be verified through Vélu's formulas; if not a new guess is made. Given that the p^k-torsion groups are cyclic, at most $\varphi(p^k)$ different guesses must be made.

Despite its simplicity, Couveignes' algorithm requires some heavy computer algebra artillery to achieve a decent complexity, but with some effort it can be made to run in $\tilde{O}(\ell^2 p^3)$ operations [Cou00; DS09; De 11]. However, the polynomial dependency in p is a serious handicap, quickly making the algorithm unusable as the characteristic grows. Couveignes' other algorithm is affected by the same problem, whereas Lercier's algorithm only works when $p = 2$.

With the introduction of p-adic alternatives to Schoof's point counting algorithm [Sat00; Ked01; Ked04; Lau04; Har14], interest in solutions to the explicit isogeny problem limited to such small characteristic started to fade. Later, Lercier and Sirvent [LS08b] explained how to extend Elkies' algorithm to finite fields of any characteristic by lifting the explicit isogeny problem to a p-adic field. Their algorithm only has a logarithmic dependency in the characteristic, and gracefully degrades to Elkies' algorithm when p becomes large enough. Said otherwise, Lercier and Sirvent effectively rendered all previous algorithms obsolete!

Incidentally, this coincides with the beginning of my career in research, one that started off by desperately trying to beat the cycles out of an algorithm that would be made obsolete before the end of my first year as a PhD student.[8]

Nevertheless, Couveignes' algorithm is still a great source of inspiration, with many ramifications that we shall explore in the rest of this work. By the end of this chapter

[8] I can only imagine FM's cold sweats when Lercier and Sirvent published their algorithm. I did not understand at the time. I do now.

it will be clear that its algebraic nature, deeply related to Tate's isogeny theorem, has more to offer than what may appear at first glance.

I.3 The neighborhood

From now on, $\mathbb{F}_q$ will be a finite field of characteristic p, and all elliptic curves and isogenies will be defined over it, unless stated otherwise.

We want to explore the "neighborhood" of $E/\mathbb{F}_q$, i.e., given a prime ℓ, how many ℓ-isogenous curves to E are there? What properties do they have?

Fortunately, we have a Swiss-army-knife to answer these questions. The *Frobenius endomorphism* is the map

$$\begin{aligned} \pi : E &\longrightarrow E, \\ (x,y) &\longmapsto (x^q, y^q). \end{aligned}$$

Hasse's well known theorem states that π, as an element of the endomorphism ring $\mathrm{End}(E)$, satisfies a quadratic equation with integer coefficients $\pi^2 + q = t\pi$, where t is called the *trace* of π. Hasse also proved that $\Delta_\pi = t^2 - 4q \leq 0$, with equality happening only if E is supersingular. Δ_π is called the *discriminant of* π.

An isogeny $\phi : E \to E/G$ is $\mathbb{F}_q$-rational if and only if $\pi(G) = G$, which suggests looking at the restriction of π to $E[\ell]$. Assume $\ell \neq p$, then $E[\ell]$ is a group of rank 2 and π acts on it like an element of $\mathrm{GL}_2(\mathbb{F}_\ell)$, up to conjugation. Clearly, the order of π in $\mathrm{GL}_2(\mathbb{F}_\ell)$ is the degree of the smallest extension of $\mathbb{F}_q$ where all ℓ-isogenies of E are defined. But we can tell even more by diagonalizing the matrix: π must have between 0 and 2 eigenvalues, and the corresponding eigenvectors define kernels of rational isogenies. We thus are in one of the following four cases[9]:

(0) π is not diagonalizable in $\mathbb{F}_\ell$, then E has no ℓ-isogenies.

(1.1) π has one eigenvalue of (geometric) multiplicity one, i.e., it is conjugate to a non-diagonal matrix $\left(\begin{smallmatrix} \lambda & * \\ 0 & \lambda \end{smallmatrix}\right)$; then E has one ℓ-isogeny.

(1.2) π has one eigenvalue of multiplicity two, i.e., it acts like a scalar matrix $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \lambda \end{smallmatrix}\right)$; then E has $\ell + 1$ isogenies of degree ℓ.

(2) π has two distinct eigenvalues, i.e., it is conjugate to a diagonal matrix $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \mu \end{smallmatrix}\right)$ with $\lambda \neq \mu$; then E has two ℓ-isogenies.

Naturally, the number of eigenvalues of π depends on the factorization of the polynomial $x^2 - tx + q$ over $\mathbb{F}_\ell$, or equivalently on whether Δ_π is a square modulo ℓ.

Each of the four cases also corresponds to a different factorization pattern of the modular polynomial. The following proposition is at the heart of Atkin's improvement to Schoof's point counting algorithm.

Proposition I.12 (Atkin [Atk91; Atk92])**.** *Let $E/\mathbb{F}_q$ be a curve with $j(E) \neq 0, 1728$. Let ℓ be a prime different from the characteristic, and let Φ_ℓ be the ℓ-th modular polynomial. The number of distinct $\mathbb{F}_q$-rational normalized ℓ-isogenies of E is equal to the number of linear factors of $\Phi_\ell(j(E), y)$ over $\mathbb{F}_q$; furthermore, the factorization degree pattern of $\Phi_\ell(j(E), y)$ falls into one of these four categories:*

(0) $r, \ldots, r$ for some r dividing $\ell + 1$;

[9] In the point counting literature, Case (0) is known as the *Atkin case*, and Case (2) as the *Elkies case*.

(1.1) $1, \ell$*;*

(1.2) $1, \ldots, 1$*;*

(2) $1, 1, r, \ldots, r$ *for some* r *dividing* $\ell - 1$.

For ordinary elliptic curves, Kohel [Koh96] showed that this classification can be further refined by introducing a notion of "depth" of an elliptic curve. Let $K = \mathbb{Q}(\pi)$ be an imaginary quadratic number field where we identify the Frobenius π to one root of $x^2 - tx + q$. Let $\mathcal{O}_K$ be the ring of integers of K then $\mathrm{End}(E)$ is isomorphic to an order $\mathcal{O}$

$$\mathbb{Z}[\pi] \subseteq \mathcal{O} \subseteq \mathcal{O}_K.$$

We have already seen that two elliptic curves are isogenous over $\mathbb{F}_q$ if and only if they have the same number of points; equivalently, they are isogenous if and only if $\mathbb{Q}(\pi_E) \simeq \mathbb{Q}(\pi_{E'})$. Kohel refined Tate's theorem as follows.

Proposition I.13 (Kohel [Koh96, Prop. 21])**.** *Let* E, E' *be elliptic curves defined over a finite field, and let* $\mathcal{O}, \mathcal{O}'$ *be their respective endomorphism ring. Suppose that there exists an isogeny* $\phi : E \to E'$ *of prime degree* ℓ*, then* $\mathcal{O}$ *contains* $\mathcal{O}'$ *or* $\mathcal{O}'$ *contains* $\mathcal{O}$*, and the index of one in the other divides* ℓ*.*

For a fixed prime ℓ, Kohel defines a curve E to be *at the surface* if $v_\ell([\mathcal{O}_K : \mathrm{End}(E)]) = 0$, where v_ℓ is the ℓ-adic valuation. E is said to be *at depth* d if $v_\ell([\mathcal{O}_K : \mathrm{End}(E)]) = d$; the maximal depth being $d_{\max} = v_\ell([\mathcal{O}_K : \mathbb{Z}[\pi]])$, curves at depth $d_{\max}$ are said to be *at the floor (of rationality)*, and $d_{\max}$ is called the *height* of the graph of E. Kohel calls then an ℓ-isogeny *horizontal* if it goes to a curve at the same depth, *descending* if it goes to a curve at greater depth, *ascending* if it goes to a curve at lesser depth.

But how many horizontal and vertical ℓ-isogenies does a given curve have? Typically this question is answered by the theory of complex multiplication, but we shall use another strategy that better serves our purpose. So far, the Frobenius endomorphism has only given us a "local" view on the neighboring curves. We need to "elevate" our point of view and look further away, in order to gain a global view on the whole isogeny class.

I.4 How isogeny graphs fold

An *isogeny graph* is a connected graph whose vertices are elliptic curves up to isomorphism, and whose edges are isogenies under some restrictions. In this chapter we are only interested in graphs of ℓ-isogenies, for some fixed prime ℓ; other types of isogeny graphs will appear in Chapter III. Because of the dual isogeny theorem, these isogeny graphs are undirected; technically we should be more properly speaking of directed multi-graphs, since multiple edges and self-loops are possible, but these cases are rare enough that we can deal with them explicitly.

As a first example, let us start with elliptic curves over the complex numbers. We know every such curve has exactly $\ell + 1$ isogenies, thus every vertex in the isogeny graph has out degree $\ell + 1$. If we let $E/\mathbb{C}$ be a curve *without complex multiplication*, i.e., such that $\mathrm{End}(E) = \mathbb{Z}$, then its connected component cannot have loops, otherwise that would provide a non-trivial endomorphism of E. Hence, the isogeny graph of E is an infinite $(\ell + 1)$-tree, as pictured in Figure I.2.

To study the structure of these graphs we introduce a tool, a sort of "lighthouse" planted at the origin, lighting the graph as it extends away towards infinity. Here is

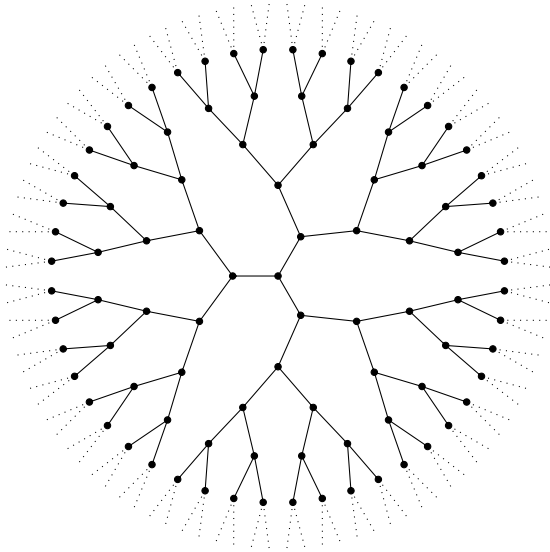


Figure I.2: Infinite 2-isogeny graph of elliptic curves without complex multiplication.

an intuition: if we put E at the origin of the graph,[10] its neighbors are determined by the cyclic subgroups of $E[\ell]$; if we "climb" to $E[\ell^n]$, we will be able to "see" as far as the ball of radius n around E. To make sense of the whole graph, it thus feels natural to climb "infinitely high", i.e., to ascend to the *Tate module* $T_\ell(E)$.

The *Tate module* $T_\ell(E)$ is the *projective limit*

$$T_\ell(E) = \varprojlim E[\ell^n]$$

given by the natural projections

$$E[\ell^n] \xrightarrow{[\ell]} E[\ell^{n-1}].$$

Since the $E[\ell^n]$ are $\mathbb{Z}/\ell^n\mathbb{Z}$-modules, $T_\ell(E)$ has a $\mathbb{Z}_\ell$-module structure, where $\mathbb{Z}_\ell$ denotes the ℓ-adic integers. Any isogeny $\phi : E \to E'$ induces a morphism $\phi_\ell : T_\ell(E) \to T_\ell(E')$ on the Tate modules, and we may prove that *no information is lost* in the process (see [Sil92, III, Th 7.4]).

Theorem I.14. *Let E, E' be elliptic curves defined over a field k, and let ℓ be a prime different from the characteristic of k. The canonical map*

$$\mathrm{Hom}(E, E') \otimes \mathbb{Z}_\ell \longrightarrow \mathrm{Hom}(T_\ell(E), T_\ell(E'))$$

is injective.

We can thus associate elements of $\mathrm{GL}_2(\mathbb{Q}_\ell)$ to the isogeny graph rooted in E as follows. Fix a basis of $T_\ell(E)$, there are $\ell+1$ isogenies of degree ℓ from E to other curves E', determined by their respective kernels; up to a change of basis of $T_\ell(E')$ the matrix ϕ_ℓ (acting on the left) associated to $\phi : E \to E'$ is one of

$$\begin{pmatrix} \ell & 0 \\ 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & a \\ 0 & \ell \end{pmatrix} \text{ for } 0 \leq a < \ell. \tag{I.5}$$

[10] An infinite tree has no well defined origin, but we may arbitrarily choose one.

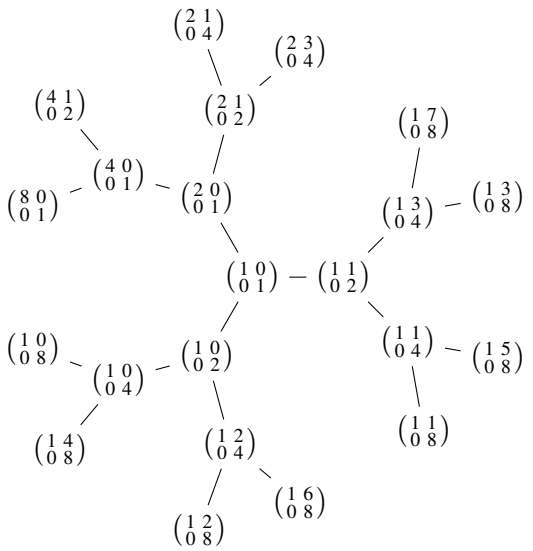


Figure I.3: Dyadic Serre tree, representing isogenies of degree 2^n on $T_2(E)$.

By composing these elementary matrices, we obtain the matrix of any isogeny of degree ℓ^n; then, quotienting by the center of $\mathrm{GL}_2(\mathbb{Q}_\ell)$, we factor out endomorphisms of E.

We thus define an infinite tree on $\mathrm{PGL}_2(\mathbb{Q}_\ell)/\mathrm{PGL}_2(\mathbb{Z}_\ell)$, isomorphic to the graph of E, by associating the identity matrix to the origin, and the matrices ϕ_ℓ to the paths $\phi : E \to E'$, as shown in Figure I.3. The *tree of* $\mathrm{PGL}_2(\mathbb{Q}_\ell)$ was already studied by Serre [Ser77, p. II],[11] and is at the heart of various constructions of *expander graphs* [LPS88; Lub94; Cos+18], a topic that we shall encounter again in Chapter III.[12]

Despite the nice drawings, these graphs are, algebraically, "boring": the choice of an origin is arbitrary, and they look the same from every vertex. Things get more interesting if we go back to finite fields. Any curve $E/\mathbb{F}_q$ can be seen as the reduction modulo p of some curve $E/\bar{\mathbb{Q}}$; thus it must inherit the connectivity structure of the isogeny graph of $E/\bar{\mathbb{Q}}$. However, there is only a finite number of curves defined over $\mathbb{F}_q$, and not all isogenies will be $\mathbb{F}_q$-rational. Thus, the infinite tree of $\mathrm{PGL}_2(\mathbb{Q}_\ell)$ must somehow "fold" to fit inside $\mathbb{F}_q$.

For example, if $E/\mathbb{F}_q$ is a supersingular curve, we shall see in Chapter III that its isogeny graph "folds" to a finite $(\ell+1)$-regular graph containing all supersingular curves, up to $\bar{\mathbb{F}}_q$-isomorphisms.

For the case of ordinary curves, we have already discussed the notion of "depth", we thus know that, as we travel along a path of descending isogenies, there is an algebraic invariant that tells us how far we are from the surface. Said otherwise, unlike the graph of $E/\mathbb{C}$ without complex multiplication, that of $E/\mathbb{F}_q$ has one (or more) recognizable origins.

Is it possible to read on $T_\ell(E)$ the depth of E? We again turn to the Frobenius endomorphism π for help. Tate's isogeny theorem makes a stronger statement than Theorem I.14, by restricting to morphisms that are invariant under the action of π: it

[11] Note that Serre uses a different convention for the elementary matrices in Eq. (I.5): he has them act on the right, instead of on the left, and thus obtains $\left(\begin{smallmatrix}1&0\\0&\ell\end{smallmatrix}\right), \left(\begin{smallmatrix}\ell&a\\0&1\end{smallmatrix}\right), \dots$ as a basis.

[12] I am grateful to J. Plût for explaining this to me, and for providing the Ti*k*Z code for Figure I.3

tells us that, for finite fields, studying Galois-invariant morphisms of $T_\ell(E)$ is the same as studying rational isogenies of E.

Theorem I.15 (Tate [Tat66]). *Let $\mathbb{F}_q$ be a finite field of characteristic p, and let $\ell \neq p$ be a prime. Let E, E' be elliptic curves defined over $\mathbb{F}_q$, the canonical map*

$$\mathrm{Hom}_{\mathbb{F}_q}(E,E') \otimes \mathbb{Z}_\ell \longrightarrow \mathrm{Hom}_{\mathrm{Gal}(\bar{\mathbb{F}}_q/\mathbb{F}_q)}(T_\ell(E), T_\ell(E'))$$

is an isomorphism of $\mathbb{Z}_\ell$-modules.

Tate's theorem has many important consequences. Among those, we have already mentioned that E and E' are isogenous if and only if $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$. Furthermore, the action of π on $T_\ell(E)$ provides a 2-dimensional representation of $\mathrm{Gal}(\bar{\mathbb{F}}_q/\mathbb{F}_q)$, and Tate's theorem states that E and E' are isogenous over $\mathbb{F}_q$ if and only if $T_\ell(E)$ and $T_\ell(E')$ are isomorphic as $\mathbb{Q}_\ell$-representations. By explicitly computing this representation we obtain an *effective version of Tate's theorem*; one that lets us, in Kohel's words, "probe the depths".

We again let $K = \mathbb{Q}(\pi)$ be an imaginary quadratic number field where we identify the Frobenius π to one root of $x^2 - tx + q$; we let $\Delta_\pi = t^2 - 4q$ be the *discriminant* of $\mathbb{Z}[\pi]$, and Δ_K the *fundamental discriminant* of $\mathcal{O}_K$. In particular, $[\mathcal{O}_K : \mathbb{Z}[\pi]] = \sqrt{\Delta_\pi/\Delta_K}$.

Proposition I.16 (Miret *et al.* [Mir+08], Hugounenq [Hug17]). *Let $E/\mathbb{F}_q$ be an ordinary elliptic curve with Frobenius endomorphism π, and let $h = v_\ell(\sqrt{\Delta_\pi/\Delta_K})$. There exists a unique $e \in \{0,h\}$ such that $\pi|T_\ell(E)$ is conjugate, over $\mathbb{Z}_\ell$, to a matrix $M = \left(\begin{smallmatrix} a & \ell^e \\ c & d \end{smallmatrix}\right)$ with $ad \wedge \ell = 1$, $a = d \pmod{\ell^h}$, and $v_\ell(\Delta_\pi) \leq v_\ell(c) + e$. In particular, $M = \left(\begin{smallmatrix} a & \ell^e \\ 0 & a \end{smallmatrix}\right) \pmod{\ell^h}$.*

Moreover, $h = v_\ell([\mathcal{O}_K : \mathbb{Z}[\pi]])$ is the height of the graph of E; if E lies at the surface, then $e = h$, otherwise $h - e$ is the depth of E.

In the case where $\pi^2 - t\pi + q$ splits in $\mathbb{Z}_\ell$, i.e., when $\left(\frac{\Delta_K}{\ell}\right) = 1$, we have a more precise statement.

Proposition I.17 (D., Hugounenq, Plût, Schost [De +16]). *Let $E/\mathbb{F}_q$ be an ordinary elliptic curve with Frobenius endomorphism π. Assume that the characteristic polynomial of π has two distinct roots λ, μ in $\mathbb{Z}_\ell$, and let $h = v_\ell(\lambda - \mu) = v_\ell(\sqrt{\Delta_\pi/\Delta_K})$. Then there exists a unique $e \in \{0,h\}$ such that $\pi|T_\ell(E)$ is conjugate, over $\mathbb{Z}_\ell$, to the matrix $\left(\begin{smallmatrix} \lambda & \ell^e \\ 0 & \mu \end{smallmatrix}\right)$.*

We thus have an effective *bathymeter* to navigate the isogeny graph: it is indeed sufficient to compute $\pi|T_\ell(E)$ up to precision ℓ^h, i.e., $\pi|E[\ell^h]$, in order to determine the depth of E. This generalizes previous partial results of Miret *et al.* [Mir+06; Mir+08] and Ionica and Joux [IJ13].

But what about horizontal isogenies? Can we construct indefinitely long walks entirely made of them? The effective version of Tate's theorem also gives us an effective way to characterize horizontal isogenies. Indeed, if $\phi : E \to E'$ is an $\mathbb{F}_q$-rational isogeny, ϕ_ℓ its restriction to $T_\ell(E)$, and we let π, π' be the Frobenius endomorphisms of E, E', then $\pi' = \phi_\ell \pi \phi_\ell^{-1}$ (where we have tensored by $\mathbb{Q}_\ell$ to make sense of ϕ_ℓ^{-1}).

We have already conveniently arranged all isogenies of degree ℓ^n in the graph of Figure I.3, thus, if we are given a matrix for $\pi|T_\ell(E)$, all we have to do to compute $\pi|T_\ell(E')$ is to conjugate by the corresponding matrix ϕ_ℓ. For example, assume that the characteristic polynomial of π has two distinct roots, so that we are in the setting of Proposition I.17. If π diagonalizes as $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \mu \end{smallmatrix}\right)$, the two isogenies $\left(\begin{smallmatrix} 1 & 0 \\ 0 & \ell \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} \ell & 0 \\ 0 & 1 \end{smallmatrix}\right)$ do not change the matrix of π, thus they are both horizontal, whereas all other isogenies are

			Isogeny types		
			$\rightarrow$	$\uparrow$	$\downarrow$
$v_\ell(\Delta_\pi/\Delta_K)=0$	$\ell \nmid [\mathcal{O}_K:\mathcal{O}]$	$\ell \nmid [\mathcal{O}:\mathbb{Z}[\pi]]$	$1+\left(\frac{\Delta_K}{\ell}\right)$		
$v_\ell(\Delta_\pi/\Delta_K)>1$	$\ell \nmid [\mathcal{O}_K:\mathcal{O}]$	$\ell \mid [\mathcal{O}:\mathbb{Z}[\pi]]$	$1+\left(\frac{\Delta_K}{\ell}\right)$		$\ell-\left(\frac{\Delta_K}{\ell}\right)$
	$\ell \mid [\mathcal{O}_K:\mathcal{O}]$	$\ell \mid [\mathcal{O}:\mathbb{Z}[\pi]]$		1	ℓ
	$\ell \mid [\mathcal{O}_K:\mathcal{O}]$	$\ell \nmid [\mathcal{O}:\mathbb{Z}[\pi]]$		1	

Table I.1: Number and types of ℓ-isogenies, according to splitting type of the characteristic polynomial of π.

descending. On the other hand, if π can only be put in the form $\left(\begin{smallmatrix}\lambda & \ell^e\\ 0 & \mu\end{smallmatrix}\right)$ with $e<h$, we see that the isogeny $\left(\begin{smallmatrix}\ell & 0\\ 0 & 1\end{smallmatrix}\right)$ is ascending, whereas all others are descending. Finally, if π is of the form $\left(\begin{smallmatrix}\lambda & 1\\ 0 & \mu\end{smallmatrix}\right)$, then we have one ascending isogeny as before, however no descending isogeny can be rational.

By applying the same reasoning to $\left(\frac{\Delta_K}{\ell}\right)=-1,0$, we can prove a complete classification of rational isogenies. This is summarized in Table I.1.

Theorem I.18 (Kohel [Koh96])**.** *Let $E/\mathbb{F}_q$ be an ordinary elliptic curve, π its Frobenius endomorphism, and Δ_K the fundamental discriminant of $\mathbb{Q}(\pi)$.*

1. *If E is not at the floor, there are $\ell+1$ isogenies of degree ℓ from E, in total.*
2. *If E is at the floor, there are no descending ℓ-isogenies from E.*
3. *If E is at the surface, then there are $\left(\frac{\Delta_K}{\ell}\right)+1$ horizontal ℓ-isogenies from E (and no ascending ℓ-isogenies).*
4. *If E is not at the surface, there are no horizontal ℓ-isogenies from E, and one ascending ℓ-isogeny.*

This theorem shows that, away from the surface, isogeny graphs just look like ℓ-regular complete trees of bounded height, with ℓ descending isogenies at every level except the floor. However, the surface has a more varied structure:

(0) If $\left(\frac{\Delta_K}{\ell}\right)=-1$, there are no horizontal isogenies: the isogeny graph is just a complete tree of degree $\ell+1$ (in the graph theoretic sense) at each level but the last. We call this the *Atkin case*, as it is an extension of the Atkin case in the SEA point counting algorithm.

(1) If $\left(\frac{\Delta_K}{\ell}\right)=0$, there is exactly one horizontal isogeny $\phi: E\to E'$ at the surface. Since E' also has one horizontal isogeny, it necessarily is $\hat{\phi}$, so the surface only contains two elliptic curves, each the root of a complete tree. We call this the *ramified case*.

(2) The case $\left(\frac{\Delta_K}{\ell}\right)=1$ is arguably the most interesting one. Each curve at the surface has exactly two horizontal isogenies, thus the subgraph made by curves on the surface is two-regular and finite, i.e., a cycle. The eigenvalue λ (resp. μ)

of π defines an eigenspace, that projects onto a cyclic subgroup of $E[\ell^n]$, which is the kernel of an ℓ^n-isogeny represented by the matrix $\left(\begin{smallmatrix}\ell^n & 0\\ 0 & 1\end{smallmatrix}\right)$ (resp. $\left(\begin{smallmatrix}1 & 0\\ 0 & \ell^n\end{smallmatrix}\right)$). Hence, λ and μ define two opposite *directions* on the cycle, independent of the starting point, and dual to one another.

Below each curve of the surface there are $\ell - 1$ curves, each the root of a complete tree. We call this the *Elkies case*, again by extension of point counting.

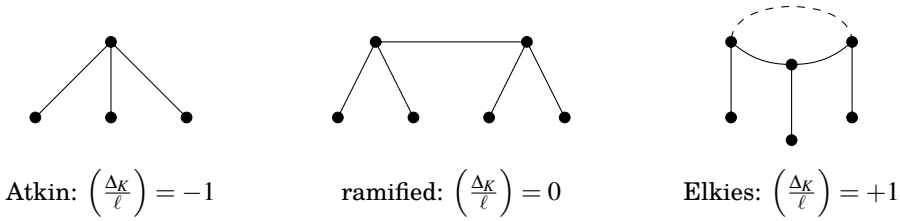


Figure I.4: The three shapes of volcanoes of 2-isogenies of height 1.

The three cases are summarized in Figure I.4. Tate's theorem only allows us to tell as much; to know more on the number and sizes of isogeny graphs, we shall need the theory of *complex multiplication*, however we delay this to Chapter III, where it will be used to construct "cryptographic" isogeny graphs.

The shapes of the graphs, in particular the Elkies case, suggest a geological metaphor: Fouquet and Morain [FM02] famously called them *isogeny volcanoes*. Adhering to this metaphor, from now on we shall call *crater* the cycle at the surface of an Elkies volcano, but we shall refrain from using this name for the surface of other types of volcanoes.[13] Of course, to reconcile Kohel's maritime metaphors with Fouquet and Morain's, we shall assume that isogeny volcanoes are underwater, with the crater just touching the sea surface.

I.5 Explicit isogenies in quadratic time

Armed with our new knowledge on isogeny volcanoes, we can now come back to the explicit isogeny problem.

Recall Couveignes' algorithm: it *interpolates* an isogeny $\phi : E \to E'$ of degree r over the p^k-torsion subgroups, for k large enough. Its main disadvantage is the polynomial dependency in p, the characteristic of the base field; in practice, Couveignes' algorithm is hardly practical for $p > 3$.

To get rid of the bad dependency in p, the obvious idea is to replace $E[p^k]$ with $E[\ell^k]$ for some small prime ℓ coprime to r, say $\ell = 2$. However, a naive algorithm based on this would have a much worse complexity than Couveignes' original algorithm. Indeed $E[p^k]$ is cyclic, thus there are only $\varphi(p^k)$ possible morphisms $E[p^k] \to E'[p^k]$ to test; if each test takes $p^{k+O(1)}$ operations, the whole algorithm takes $p^{2k+O(1)} = r^2 p^{O(1)}$. On the other hand, $E[\ell^k]$ is of rank 2, thus $\mathrm{Hom}(E[\ell^k], E'[\ell^k])$ is isomorphic to $\mathrm{GL}_2(\mathbb{Z}/\ell^k\mathbb{Z})$ and has size $O(\ell^{4k})$; if each interpolation test takes $\ell^{2k+O(1)}$ operations, the whole algorithm takes $\ell^{6k+O(1)} = r^3 \ell^{O(1)}$.

[13] The literature, including my own works [De +16], is inconsistent on the use of the word "crater" for non-Elkies volcanoes.

But we are not interested in *any* isogeny: we are explicitly looking for a *rational* isogeny, thus we can use all that we have learned so far. Indeed, we are just applying Tate's theorem: trying to identify, among all matrices in $\mathrm{Hom}(T_\ell(E), T_\ell(E'))$ (truncated to precision ℓ^k), the one that corresponds to the isogeny ϕ. Since ℓ does not divide $\deg\phi$, the curves E and E' have the same depth in their respective volcanoes (which may or may not be distinct); and since ϕ is rational, its matrix must commute with π. Thus, even though $\mathrm{Hom}(T_\ell(E), T_\ell(E'))$ has dimension 4 as a $\mathbb{Z}_\ell$-module, we can focus on the, potentially smaller, submodule of matrices that leave π stable.

Concretely, assume that the characteristic polynomial of π has two distinct roots, and suppose that E is on the crater. Then we can find bases for $E[\ell^k]$ and $E'[\ell^k]$ such that the respective Frobenius endomorphisms act like $\left(\begin{smallmatrix}\lambda & 0\\ 0 & \mu\end{smallmatrix}\right)$ on each. Since ϕ is rational, it must map the eigenspace of λ in $E[\ell^k]$ into the eigenspace of λ in $E'[\ell^k]$, and similarly for μ. Said otherwise, ϕ must be represented by a diagonal matrix, thus the search space is reduced to a dimension 2 submodule, that is $O(\ell^{2k})$ different possibilities to try, for an overall complexity of only $r^2\ell^{O(1)}$ operations.

What we just described is the gist of the algorithm presented in "Explicit isogenies in quadratic time in any characteristic" written with C. Hugounenq, J. Plût and É. Schost [De +16], and included in the appendix to this document. Although I must admit that the title cheats in two ways:

- The algorithm solves Problem I.7 in quadratic time, i.e., not exactly the "explicit isogeny problem" as we have stated it, and thus does not improve the complexity of the SEA point counting algorithm;

- The algorithm only achieves quadratic complexity for *almost all* prime powers q and *almost all* pairs of isogenous curves E, E' defined over $\mathbb{F}_q$.

In our defense, artificial as Problem I.7 may seem, ours is the only algorithm that achieves quadratic complexity in the isogeny degree, beating even Lercier and Sirvent's algorithm. Although its impact is purely theoretical, the techniques employed are of independent interest and may find useful applications in other contexts.

Concerning the second problem, the difficulty comes from the fact that our techniques only work when the characteristic polynomial of π splits over $\mathbb{Z}_\ell$, i.e., when ℓ is an Elkies prime for E. However, it may happen that no small prime is Elkies for E, and indeed curves such that none of the first $O(\log q)$ primes is Elkies do exist, although they are "rare".[14]

Before we close this chapter, let us summarize the steps of our "ℓ-adic Couveignes' algorithm". Note that, to run the algorithm, we need an Elkies prime ℓ for E. It would be easy to find one if we knew the order of $E(\mathbb{F}_q)$, but this would be cheating, since one of the goals of Couveignes' algorithm is to help count the points of E. Instead we show that the number of roots of π in $\mathbb{Z}_\ell$ can be "discovered" as we proceed in the steps below. For simplicity, we are also going to assume that E and E' are on the craters of the respective volcanoes; note that we can always reduce to this situation using Proposition I.17.

1. For a given prime ℓ, construct torsion bases $E[\ell^k]$ and $E'[\ell^k]$, where k is chosen so that $\ell^{2k} > 4r$.

[14] Interestingly, we will look at the opposite problem in Chapter III: construct curves such that a lot of small primes are Elkies.

2. Perform a change of basis so that π acts on $E[\ell^k] = \langle P, Q\rangle$ like a diagonal matrix $\left(\begin{smallmatrix}\lambda & 0\\ 0 & \mu\end{smallmatrix}\right)$, with $\lambda \neq \mu$; do the same for $E'[\ell^k] = \langle P', Q'\rangle$. If this is not possible, either ℓ is not an Elkies prime, or we have computed $T_\ell(E)$ to too low a precision (i.e., we need to choose a larger k). In either case, we can decide to change prime ℓ and start again, or to increase k up to an acceptable bound.

3. For each diagonal matrix M in $\mathrm{GL}_2(\mathbb{Z}/\ell^k\mathbb{Z})$, interpolate the isogeny that maps $(P,Q)^t$ to $M(P',Q')^t$. If this results in a rational isogeny of degree r, return it and stop.

Pretty simple, huh? Well, now it is time to look at what we swept under the rug. So far we have spoken of "constructing $E[\ell^k]$" as if this was an easy thing to do. However the attentive reader will have noticed that $E[\ell^k]$ may be not (entirely) contained in $E(\mathbb{F}_q)$, and indeed it will almost never be in the context of our algorithm. Thus, we first need to compute the smallest field extension of $\mathbb{F}_q$ where $E[\ell^k]$ is going to be defined. We "ascend" level by level: first computing $E[\ell]$, then $E[\ell^2]$, and so on until we reach $E[\ell^k]$. Each step will require factoring some polynomials, in general of degree ℓ, leading to the construction of a *tower of extensions* on top of $\mathbb{F}_q$. Performing computations in towers of field extensions in optimal time is a delicate task, requiring a great deal of computer algebra techniques that we shall explore in the next chapter.

I.6 Perspectives

Point counting. The *raison d'être* of the explicit isogeny problem lies in improving Schoof's point counting algorithm. In this chapter, we have not succeeded in improving its complexity, and, to be completely honest, we have not even tried: any solution to the explicit isogeny problem wanting to improve upon the Elkies-Lercier-Sirvent algorithm needs to get rid of the modular polynomial first. Indeed, even assuming an optimal algorithm[15] to compute Φ_ℓ, simply storing its coefficients requires $O(\ell^3)$ bits, that become $O(\ell^2 \log p)$ after reducing modulo p.

A possible way around would be an algorithm to compute $\Phi_\ell \mod p$ directly, without computing its integer coefficients first; however the best algorithm for this [BLS12], a multi-modular approach exploiting the structure of isogeny volcanoes, only achieves quasi-optimal storage, but still requires $\tilde{O}(\ell^3)$ binary operations. Even better, one could compute $\Phi_\ell(j(E), y)$ directly, as Sutherland does [Sut13b], however even for this problem we only have an algorithm with quasi-optimal storage, but the same cubic complexity.

The same problem is felt, even more strongly, for curves of higher genus. Indeed, even for the case of genus two curves, modular polynomials are so unwieldy [Gau00; Dup06; BL09; Mil15; MR17] that they do not allow improving upon the basic Schoof-Pila algorithm [Pil90].

To the present day, the only known techniques to enumerate isogenous curves of a fixed degree are based on factoring the division polynomial or the modular polynomial. The techniques of this chapter do not seem to help. I am personally rather pessimistic on the possibility of improving the SEA algorithm using Tate's isogeny theorem alone, however it is certainly interesting to try to combine it with other ideas, in the hope of getting a breakthrough in point counting, especially for higher genus curves.

[15] Quasi-optimal algorithms for computing modular polynomials do exist, see [Eng09; BLS12]

Couveignes' algorithm. Moving to Problem I.7 and to the "ℓ-adic Couveignes' algorithm" presented in the previous section, even though we know that it does not improve the asymptotic complexity of point counting, it would still be interesting to know for which parameters it improves SEA in practice, and by how much.

Related to this, a goal that looks realistic would be to lift the restriction to Elkies primes in the algorithm; hopefully having it run in quadratic time for any elliptic curve. To be more precise, our paper uses [SS14] to show that one can find an Elkies prime $\ell = O(\log q)$ for almost all finite fields $\mathbb{F}_q$ and curves $E/\mathbb{F}_q$. In his PhD thesis [Hug17], C. Hugounenq partially solves this problem by giving a quadratic time algorithm when $\left(\frac{\Delta_\pi}{\ell}\right) = -1$, i.e. when ℓ is an Atkin prime and the corresponding volcano has height 0. This allows him to prove the existence of a quadratic algorithm for any elliptic curve, however it does not improve upon the bound $\ell = O(\log q)$. A quadratic algorithm working with any ℓ would considerably improve the complexity in $\log q$, and would also be much more practical and easy to implement.

It is tempting to look for a variant of Couveignes' algorithm with sub-quadratic complexity, possibly even quasi-linear. The techniques developed in this chapter do not seem capable of breaking the quadratic barrier, and this looks somehow intrinsic to Tate's theorem. My guess is that, if it was possible to obtain a sub-quadratic variant of Couveignes' algorithm, bad things would start happening for the cryptosystems presented in Chapter III, at least those based on complex multiplication.

Computing isogenies of supersingular curves would be another obvious extension of Couveignes' algorithm. Couveignes' original algorithm simply does not apply to supersingular curves, because p^k-torsion groups are trivial. Our ℓ-adic algorithm is easily adapted to supersingular curves defined over $\mathbb{F}_p$, because, as we shall see, their Frobenius endomorphism behaves similarly to that of ordinary curves; however it does not achieve the desired complexity for general supersingular curves. This is deeply related to the differences between the CSIDH and SIDH protocols presented in Chapter III, and their security.

Computing endomorphism rings. Kohel's original motivation for defining depth and direction was to compute the endomorphism ring of an ordinary curve $E/\mathbb{F}_q$, a problem strictly harder than point counting. Indeed, knowing $\#E$ determines π, which in turn determines $\mathbb{Q}(\pi)$; the only thing that is left to know, then, is the depth of E in each of the ℓ-volcanoes, for ℓ^2 dividing Δ_π. The problem with this is that ℓ is potentially as large as $O(\sqrt{q})$, and thus any algorithm computing ℓ-isogenies is bound to have exponential complexity. An alternative approach using isogenies of smooth degree, due to Bisson and Sutherland [BS11], achieves sub-exponential complexity.

The effective versions of Tate's theorem give an alternative way to determine the depth of an elliptic curve, one that potentially has polynomial complexity in $\log q$. However, the methods proposed in this chapter to compute $\pi | T_\ell(E)$ involve computing the ℓ-torsion, and thus have polynomial complexity in ℓ.

I find it unlikely that the techniques of this chapter could improve significantly the computation of endomorphism rings, but let's be optimistic and imagine a sci-fi scenario. In the same way that Schoof's algorithm computes the trace of π modulo many small primes to find its value in $\mathbb{Z}$, one may hope that there is some sort of "global" description of $\pi | \prod'_\ell T_\ell(E)$ that can be reconstructed from $\pi | T_{\ell'}(E)$ for many small primes ℓ'. If this description could be computed in polynomial time, then we would have a polynomial time algorithm for the endomorphism ring.

Besides science-fiction, the way isogeny volcanoes interact for different primes has received very little attention so far, the only work I am aware of being [Moo12]. I believe that some interesting algorithmic ideas could derive from studying how the knowledge of $\pi|T_\ell(E)$ affects the computation of $\pi|T_{\ell'}(E)$.

II Altimetry

In the previous chapter we saw how to determine the structure of an isogeny volcano by looking at the way the Frobenius endomorphism acts on the Tate module. To effectively perform the computation, we need to approximate the Tate module by projecting it onto a torsion group of order, say, ℓ^k. Even this finite group, however, may not be defined over the base field. It is then natural to construct *towers of extensions fields*, over which increasingly larger torsion groups are defined.

But why stop at towers? This chapter is devoted to techniques to "ascend" in *lattices of extension fields*, possibly up to the full algebraic closure $\bar{\mathbb{F}}_p$. The structure of $\bar{\mathbb{F}}_p$ is simple enough that we will not need more than the basic theory of cyclotomic extensions. Instead, we will concentrate our efforts on looking for asymptotically optimal algorithms, discovering on our path a rich palette of algorithmic ideas.

After learning about representations of finite fields and algorithms to compute irreducible polynomials, we will explore two radically different paradigms to represent the algebraic closure of $\mathbb{F}_p$, both being currently used in computer algebra systems. On one side, we will have lattices of finite fields represented by families of *special* polynomials, the most well known example being the family of Conway polynomials, introduced in the GAP system [GAP], and then adopted by Magma [BCP97] and SageMath [Sage]. On the other side, we will have lattices of *arbitrarily represented* finite fields, such as those used by Magma. The fundamental tool for these will be an *isomorphism algorithm*, we shall thus learn about the two main existing families: the first one, due to Lenstra [Len91] and Allombert [All02a], based on the theory of Kummer extensions; the second one, due to Pinch [Pin92] and Rains [Rai96], based on Gaussian periods.

II.1 Computing irreducible polynomials

From now on, $\mathbb{F}_p$ is a finite field of prime order. Much of what we are going to present is easily generalized to non-prime fields, however we will stick to prime fields for simplicity, and refer to the appendix the reader interested in the general case.

Since this chapter is chiefly about complexity, we need to agree on a unit of measurement. The field $\mathbb{F}_p$ is typically represented as the ring of integers modulo p, using $\log p$ bits per element. Addition, subtraction and multiplication modulo p can all be performed in $\tilde{O}(\log p)$ binary operations using asymptotically fast integer multiplication and Euclidean division, while field inversion can be computed at the same asymptotic cost using a fast extended Euclidean algorithm (see [GG99] for a detailed account). *Zech logarithms* are another commonly used representation for finite fields of small size: elements of $\mathbb{F}_p$ are represented as powers of a generator, making it relatively cheap to multiply and invert elements, whereas additions are computed by a lookup in a table with $O(p)$ entries.

Even though in practice any representation has noticeably different costs for the various arithmetic operations, it will be convenient to abstract from the actual implementation of $\mathbb{F}_p$ and measure complexities in the *algebraic model*, i.e., counting every field operation as $O(1)$. Given a complexity in the algebraic model, a relatively accurate estimate of the binary complexity can be obtained by multiplying by $\log p$ (and ignoring polynomial terms in $\operatorname{loglog} p$).

The universally employed way to represent a field extension $\mathbb{F}_{p^n}$ is as a quotient of the polynomial ring $\mathbb{F}_p[X]$ by a monic irreducible polynomial $f(X)$ of degree n. In

this representation, much like in the modular integers case, all arithmetic operations can be performed in $\tilde{O}(n)$ elementary algebraic operations using fast polynomial multiplication, Euclidean division, and extended Euclidean algorithm (see again [GG99]). Most popular software libraries for number theory, e.g., Flint [Har10], NTL [NTL], PARI/GP [PARI], Magma [BCP97], use this representation[1], and a considerable amount of effort has been spent in optimizing it. Hence this representation, that we shall call *univariate*, is the best choice both from an asymptotical and a practical point of view.

Of course, to employ this representation, we need an algorithm to compute irreducible polynomials of arbitrary degree. Three different approaches are known.

The first one, and the simplest, consists in taking random monic polynomials until an irreducible one is found. The density of irreducible polynomials of a given degree n is $\sim 1/n$, thus this approach will lead to an irreducible polynomial in $O(n)$ tries on average. Testing irreducibility of random polynomials can be done in $\tilde{O}(n \log p)$ operations on average, using Ben-Or's algorithm [Ben81; GP97], thus in total this approach has a quasi-quadratic dependency on n. This is the most commonly implemented method, available in Magma, SageMath, etc.

The second method is due to Adleman and Lenstra [AL86], and implemented, as far as I know, only in PARI/GP[2]. It is based on the properties of cyclotomic polynomials, and is similar in spirit to Rains' algorithm presented in Section II.4. Adleman and Lenstra show that their algorithm takes deterministic polynomial time, under the generalized Riemann hypothesis, albeit with a quite large exponent. However, the algorithm is quite efficient in practice, and the average case complexity of the variant implemented in PARI/GP is similar to that of factoring a degree n polynomial over $\mathbb{F}_p$.[3]

The third method is due to Shoup [Sho90; Sho93; Sho94b], later extended by Couveignes and Lercier [CL13; DDS13]. It uses a variety of algorithms, that we shall discuss in Section II.3. Using the best available routines, it can compute an irreducible polynomial in $\tilde{O}(n(\log p)^5)$ operations on average, but trade-offs are available if the cost in $\log p$ is deemed too high. We are not aware of any computer algebra software implementing this method, probably owing to the relative novelty of the method, and to its intricacies.

We note that it is an open problem to give an unconditionally deterministic algorithm to compute irreducible polynomials of arbitrary degree. The closest to this is Shoup's first algorithm [Sho90]: it consists of reduction from the problem of finding irreducible polynomials to that of polynomial factoring, and can be made fully deterministic using Berlekamp's deterministic factoring algorithm [Ber70]; however Berlekamp's algorithm has an exponential dependency in $\log p$.

II.2 From one extension to the algebraic closure

In many contexts, such as when manipulating geometrical objects, it is natural to work in many extensions of $\mathbb{F}_p$ at once. We may push this to the limit: the algebraic closure $\bar{\mathbb{F}}_p$ is the (infinite) reunion of all the finite extensions of $\mathbb{F}_p$; it is thus sufficient to represent all finite extensions of $\mathbb{F}_p$ in a *compatible* way in order to represent $\bar{\mathbb{F}}_p$.

[1] Givaro [Giv] is one notable exception, employing Zech logarithms. All of the mentioned libraries, with the exception of Magma, are used by SageMath [Sage].

[2] And thus also available in SageMath.

[3] We are not aware of any published formal analysis of the PARI/GP variant, however we believe that the average-case complexity is (heuristically) dominated by the cost of factoring a cyclotomic polynomial of degree $O(n \log n)$. See also Section II.4

A natural choice to represent a collection of extensions of $\mathbb{F}_p$ is as a *towers of extensions*: for example, $\mathbb{F}_{p^2}$ may be represented as $\mathbb{F}_p[X_1]/f_1(X_1)$, then $\mathbb{F}_{p^6}$ as $\mathbb{F}_{p^2}[X_2]/f_2(X_2)$, and so on. In general, the polynomial f_i will have coefficients over the previous field. "Flattening" the tower, we may rewrite the system of extension fields as a quotient

$$\mathbb{F}_p[X_1,\ldots,X_k]/\left|\begin{array}{l} X_k^{n_k}-\tilde{f}_k(X_1,\ldots,X_k),\\ \quad\vdots\\ X_1^{n_1}-\tilde{f}_1(X_1),\end{array}\right. \tag{II.1}$$

where the i-th field in the tower is identified with the subring generated by $X_1,\ldots,X_i$. The polynomial ideal in Eq. (II.1) is a special case of a (zero-dimensional) *triangular set*, and an extensive literature is devoted to computing modulo them, both in dimension 0 [Leb15; PS13b], and in higher dimension [ALM99]. Performing arithmetic operations modulo triangular sets incurs an intrinsic penalty, exponential in the number k of variables, that an ordinary univariate representation does not [CKY89; LMS07; Hoe04].

To recover the quasi-optimal performance of the univariate representation, we may seek a *change of order*[4] algorithm to rewrite the quotient as

$$\mathbb{F}_p[X_1,\ldots,X_k]/\left|\begin{array}{l} X_k^{e}-g_n(X_k),\\ \quad\vdots\\ X_2-g_2(X_k),\\ X_1-g_1(X_k).\end{array}\right.$$

This representation goes under various names, such as *rational univariate representation* [Rou99] and *geometric resolution* [GLS01]. Using this representation, we can efficiently perform arithmetic in the top level of the tower, and we can still identify the i-th intermediate fields as being generated by X_i, or equivalently by the polynomial expression $g_i(X_k)$.

However, our special instance enjoys many special properties that general triangular sets do not, and we wish to exploit them. On the other hand, the tower of extensions paradigm is not adapted to all situations: for example, a field $\mathbb{F}_{p^{mn}}$ with $m\wedge n=1$ can be seen both as an extension of $\mathbb{F}_{p^n}$ and as one of $\mathbb{F}_{p^m}$.

Ideally, we would like to have a *data structure* to represent *arbitrary collections* of extensions of $\mathbb{F}_p$, in such a way that any extension is represented in optimal space (i.e., $O(n)$ coefficients for an element of an extension of degree n), and that arithmetic operations are performed in quasi-optimal time (i.e., $\tilde{O}(n)$ operations). To this end, we now name several useful properties that we are going to seek.

Compatibility: For any pair of extensions $\mathbb{F}_p\subset k\subset K$, there is an algorithm that takes an element $x\in k$ and outputs its representation as an element of K. Reciprocally, there is an algorithm that tests whether an element $y\in K$ belongs to k, and in that case outputs its representation as an element of the smaller field.

Incrementality: The data associated with an extension (e.g., its irreducible polynomial, change-of-basis matrices, ...) must be computable efficiently and *incrementally*, i.e., adding a new field extensions to the collection does not require recomputing data for all extensions already represented.

[4] The name "change of order" comes from the theory of Gröbner bases, it is indeed equivalent to a change of order from lexicographic to inverse lexicographic.

Uniqueness: Any extension is determined by an irreducible polynomial whose definition only depends on the characteristic p and the degree of the extension.

Note that both incrementality and uniqueness are optional, however the former is necessary to represent the algebraic closure effectively, and the latter provides a *standard* way to represent it. More advanced features, such as computing normal bases, evaluating Frobenius morphisms, etc., are also (terribly) interesting, but they are out of the scope of this document.

The reader may be surprised to learn that no such representation is known! The difficulty is not a theoretical one: besides the problem of finding irreducible polynomials, any other question is amenable to linear algebra, as Lenstra showed [Len91]. Instead, the difficulty is to satisfy all requirements in an efficient, possibly quasi-optimal, manner.

Various solutions have been deployed in practice in computer algebra systems such as Magma and SageMath, however none of these is especially efficient. In the next sections we shall explore the various available constructions, and possible research avenues.

II.3 Special families of irreducible polynomials

Among all irreducible polynomials, which ones are best suited to represent a collection of finite extensions of $\mathbb{F}_p$, or potentially the collection of *all* finite extensions of $\mathbb{F}_p$? This fascinating question, investigated by many, has no single answer: indeed, depending on what is meant by "best", different solutions are possible.

Conway polynomials. One of the most famous constructions is that of *Conway polynomials*. The main feature of Conway polynomials is *norm compatibility*: the norm map $\mathbb{F}_{q^n} \to \mathbb{F}_{q^m}$ is a surjection from the roots of the n-th Conway polynomial to the roots of the m-th Conway polynomial, whenever m divides n.

Norm compatibility is easy to achieve for a fixed collection $\mathscr{F}$ of finite extensions of $\mathbb{F}_p$: let $K/\mathbb{F}_p$ be the smallest finite field containing all fields in $\mathscr{F}$, let η be a primitive element of K, i.e., a generator of the multiplicative group $K^\times$, then the Conway polynomial of a field $k \subset K$ is defined as the minimal polynomial of $N_{K/k}(\eta)$, where $N_{K/k}$ is the norm map. However, Conway polynomials have two other goals: incrementality and uniqueness. This leads to the following definition.

Definition II.1 (Conway polynomial)**.** Let p be a prime and $n > 1$ an integer. The *Conway polynomial* $C_{p,n}$ is the *lexicographically smallest* monic irreducible polynomial of degree n satisfying the following conditions:

- *Primitivity:* $C_{p,n}$ is primitive (i.e., its roots generate the multiplicative group $\mathbb{F}_{p^n}^\times$);
- *Norm compatibility:* If m divides n, then $C_{p,m}\left(X^{\frac{p^n-1}{p^m-1}}\right) = 0 \mod C_{p,n}$.

The "lexicographically smallest" condition is required to ensure uniqueness; it is typically defined by writing $f \in \mathbb{F}_p[X]$ as

$$f = \sum_{i=0}^{n} (-1)^{n-i} f_i x^i, \qquad \text{with } 0 \le f_i < p,$$

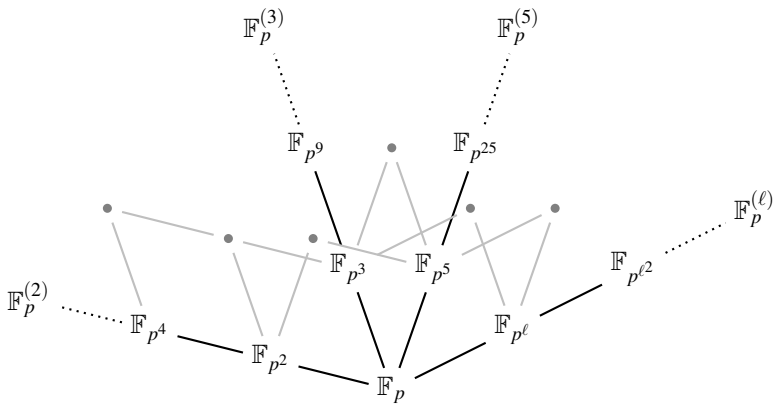


Figure II.1: Lattice of extensions of $\mathbb{F}_p$.

and taking the lexicographic order on the words $f_n \dots f_0$.

Conway polynomials were defined by Parker[5], who named them in honor of John Conway and his famous book "On Numbers and Games" [Con00]; their existence was shown by Nickel [Nic88]. They were first adopted by the computer algebra system GAP [GAP] as a default representation for finite fields. They are typically computed by exhaustive search over all irreducible polynomials, or by a slightly better algorithm due to Heath and Loehr [HL99]. Given the huge computational cost involved in finding them, they are usually precomputed; tables of Conway polynomials are available in any major computer algebra system.[6]

We note that Conway polynomials are not especially good to represent embeddings: given an element of $\mathbb{F}_{p^m}$ represented as $a(X) \bmod C_{p,m}(X)$, its image in $\mathbb{F}_{p^n}$, for $m \mid n$, is computed as $a(X^{(p^n-1)/(p^m-1)}) \bmod C_{p,n}(X)$, requiring very large modular exponentiations; while there are algorithms to perform this computation in $O(n^{1+o(1)})$ operations [KU11], they are known to be very inefficient in practice.

Primary towers. All other solutions are generalizations of Shoup's algorithm for computing irreducible polynomials [Sho90; Sho93; Sho94b].

We start by restricting to *primary towers of extensions*, i.e., towers of extensions of prime-power degree. Formally, for a prime $\ell \neq p$, define the *ℓ-adic closure* of $\mathbb{F}_p$ as the infinite field $\mathbb{F}_p^{(\ell)} = \bigcup_{i \geq 0} \mathbb{F}_{p^{\ell^i}}$.

To represent $\mathbb{F}_p^{(\ell)}$, we want to define an infinite family of irreducible polynomials of degree ℓ^i for $i > 0$, and algorithms to evaluate the corresponding embeddings $\mathbb{F}_{p^{\ell^{i-1}}} \subset \mathbb{F}_{p^{\ell^i}}$.

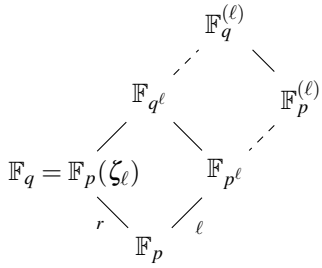
Here is a simple example: suppose that ℓ divides $p-1$, then the ℓ-th power map is not injective on $\mathbb{F}_p$, and thus $\mathbb{F}_p$ contains ℓ-adic non-residues. Let η be such a

[5] According to Lübeck [Lüb08].

[6] Most computer algebra systems switch to other methods when precomputed Conway polynomials are not available. An interesting exception is SageMath (since version 5.13 [RFB13]), that defines *pseudo-Conway polynomials* by dropping the "lexicographically first" requirement, and computes them on the fly whenever a true Conway polynomial is not available in the tables. The approach is notoriously slow: computing a pseudo-Conway polynomial for $\mathbb{F}_{p^{30}}$ takes in the order of seconds, already for $p > 1000$; compare this to the milliseconds needed to compute a random irreducible polynomial of the same degree.

non-residue, then $X_i^{\ell^i} - \eta$ is a family of irreducible polynomials, and the embeddings between the fields $\mathbb{F}_{p^{\ell^i}}$ are easily defined by the relationship $X_i^\ell = X_{i-1}$. We may call this special case the "Kummer case", for obvious reasons.

Shoup's construction generalizes the Kummer case to an arbitrary prime ℓ, by adjoining ℓ-th roots of unity to $\mathbb{F}_p$. The idea is to define an extension $\mathbb{F}_q = \mathbb{F}_p(\zeta_\ell)$ of the base field, and then construct its ℓ-adic closure $\mathbb{F}_q^{(\ell)}$. The sought closure is then identified to a subfield $\mathbb{F}_p^{(\ell)} \subset \mathbb{F}_q^{(\ell)}$ using a projection map; this is illustrated below.



Shoup explicitly computes $\mathbb{F}_q$ by factoring the ℓ-th cyclotomic polynomial, then finds an ℓ-adic non-residue $\eta \in \mathbb{F}_q$, and defines $\mathbb{F}_{q^{\ell^i}}$ as $\mathbb{F}_q(X_i)/(X_i^{\ell^i} - \eta)$. Then, a defining polynomial of $\mathbb{F}_{p^{\ell^i}}$ is found by taking the trace $\mathbb{F}_{q^{\ell^i}} \to \mathbb{F}_{p^{\ell^i}}$ of the residue class of X_i, and computing its minimal polynomial.

Shoup's construction pays an intrinsic $O(\ell)$ overhead, because of the *auxiliary extension* $\mathbb{F}_p(\zeta_\ell)$. To avoid this cost, Couveignes and Lercier's generalize the Kummer case in a different direction.

It is well known in number theory folklore that any algorithm that uses purely multiplicative properties of $\mathbb{F}_p$ can be generalized by replacing the multiplicative group $\mathbb{F}_p^\times$ with a different algebraic group $G(\mathbb{F}_p)$, e.g., the group of points of an elliptic curve $E/\mathbb{F}_p$. This principle is the basis, for example, for Lenstra's elliptic curve factoring method [Len87]. We shall see this principle applied over and over again in this chapter.

The Kummer case is based on the fact that the map $x \mapsto x^\ell$ is surjective on $\bar{\mathbb{F}}_p$, but not on $\mathbb{F}_p$. Because of its geometric properties, its fibers are irreducible and define irreducible polynomials $X^\ell - \eta$ whenever η is an ℓ-adic non-residue. Does this ring a bell?

We saw at the beginning of this document that an isogeny $\phi : E \to E'$ is a degree ℓ map, surjective on $E'(\bar{\mathbb{F}}_p)$, but not on $E'(\mathbb{F}_p)$ whenever $\ker\phi \subset E(\mathbb{F}_p)$. If we choose an elliptic curve $E/\mathbb{F}_p$ such that $\pi|E[\ell]$ acts like $\left(\begin{smallmatrix} 1 & 0 \\ 0 & \mu \end{smallmatrix}\right)$ for some $\mu \neq 1 \pmod{\ell}$, then the isogeny ϕ associated to the eigenvalue 1 is not surjective, and its fibers are irreducible.

Couveignes and Lercier take random elliptic curves, and count their points, until one with the desired Frobenius endomorphism is found. Then they compute the unique ℓ-isogeny $\phi : E \to E'$ with rational kernel, using Vélu's formulas, and find a point $P \in E'(\mathbb{F}_p)$ not in $\phi(E(\mathbb{F}_p))$. The irreducible polynomial defining $\mathbb{F}_{p^\ell}$ is then deduced from the fiber $\phi^{-1}(P)$, for example by projecting onto the x-axis. By iterating the process, we easily generalize to arbitrary primary extensions $\mathbb{F}_{p^{\ell^i}}$.

The drawback of this technique is the high cost involved in the search for a suitable elliptic curve: if $\ell < p^{1/4}$ about one curve in ℓ has the desired property [Len87; CH13], and counting the number of points takes $\tilde{O}((\log p)^5)$ operations using Schoof's

algorithm[7]. Hence, this construction is only practical for relatively small values of p and ℓ. Also, in general, ℓ may be so large that no elliptic curve exists with the desired properties. In this case Couveignes and Lercier propose extending the base field $\mathbb{F}_p$ to an extension of degree $O(\log \ell)$, and use the same projection machinery as in Shoup's original construction. Although this trick salvages the asymptotic complexity of the method, I am not sure anyone has ever dared implement it.

So far, we have only mentioned the construction of the irreducible polynomials, without talking about compatibility. In the work "Fast Algorithms for ℓ-adic Towers over Finite Fields" written with J. Doliskani and É. Schost [DDS13], we provide algorithms to efficiently evaluate embeddings, both for Shoup's and for Couveignes and Lercier's construction, and we also add incrementality to the latter. The techniques are relatively simple generalizations of the Kummer case, leveraging fast algorithms for polynomial composition and decomposition, and we will not detail them. Suffice to say that they are very efficient, both asymptotically and in practice, and achieve compatibility in quasi-linear time for the Couveignes–Lercier construction. Experiments in [DDS13] show that they beat other techniques by orders of magnitude, in particular when constructing a primary tower with many levels[8].

Couveignes and Lercier's technique, and thus our extension, does not naturally provide uniqueness: this depends intrinsically on the random choices of elliptic curves. Although it would be possible to fix the choices of the algorithm in a deterministic way, this feels even less natural than for Conway polynomials. A better option is to fix the choices made by Shoup's construction in a deterministic way; this is exactly what Lenstra and de Smit do in [LS08a], with a special focus on deterministic algorithms. However, to the best of my knowledge, no one has ever tried implementing this construction, that looks mostly of theoretical interest.

A few words on the case $\ell = p$, to finish. We can define $\mathbb{F}_p^{(p)}$ in the same way as we have defined the ℓ-adic closure, however none of the techniques described so far applies to this case. The construction of irreducible polynomials of degree p^i using Artin-Schreier theory is folklore, and already present in [AL86]. The related algorithms for computing embeddings are developed in [DS09; DS12].

Composita. Once we have ℓ-adic closures for any ℓ, we need to "glue" them together to obtain the algebraic closure. Formally, $\bar{\mathbb{F}}_p$ is isomorphic to the tensor product $\bigotimes_\ell \mathbb{F}_p^{(\ell)}$, where ℓ runs over all primes. This suggests representing an extension of degree $n = \ell_1^{e_1} \cdots \ell_k^{e_k}$ as a quotient

$$\mathbb{F}_p[X_1, \ldots, X_k] \Big/ \left| \begin{array}{l} X_k^{\ell_k^{e_k}} - f_k(X_k), \\ \quad \vdots \\ X_1^{\ell_1^{e_1}} - f_1(X_1); \end{array} \right. \tag{II.2}$$

however, arithmetic operations in this representation incur the same penalty as triangular sets.

The building block to switch to a rational univariate representation will be an algorithm to compute *composita* with explicit embeddings. Formally, given two extensions fields $\mathbb{F}_{p^m}$ and $\mathbb{F}_{p^n}$, with $m \wedge n = 1$, we want to compute a univariate representation

[7] This complexity bound could be improved using the Schoof–Elkies–Atkin algorithm, however this would add additional heuristic hypotheses to the construction.

[8] Gains are already remarkable for $\mathbb{F}_{p^{81}}$

for the field $\mathbb{F}_{p^{mn}}$, together with efficient algorithms for evaluating the embeddings $\mathbb{F}_{p^m} \hookrightarrow \mathbb{F}_{p^{mn}}$ and $\mathbb{F}_{p^n} \hookrightarrow \mathbb{F}_{p^{mn}}$. By applying this construction recursively, we can represent an arbitrary tensor product of primary extensions.

The construction of an irreducible polynomial of degree mn is folklore, and was already used by Shoup [Sho90]. Given two separable polynomials f and g, their *composed product* is the polynomial

$$(f \otimes g)(X) = \prod_{f(\alpha)=0} \prod_{g(\beta)=0} (X - \alpha\beta).$$

We similarly define the *composed sum* as the polynomial whose roots are the sums of the roots of f and g. It is well known that, if $f, g \in \mathbb{F}_p[X]$ are irreducible polynomials of coprime degree, then both their composed sum and composed product are irreducible [BC87].[9] Both polynomials can be computed in $\tilde{O}(mn)$ operations using algorithms in [Bos+06], and both have been used to construct irreducible polynomials of arbitrary degree.

The less evident part is the evaluation of the embeddings. In "Fast Arithmetic for the Algebraic Closure of Finite Fields" written with J. Doliskani and É. Schost [DDS14], and included in the appendix, we develop new techniques for embeddings of composita constructed through composed products. Our techniques are purely algebraic, and apply to any base field, however they are most interesting in practice to compute in composita of finite fields.

The main technical ingredient of our work is a representation of finite field elements using a pair of monomial/dual bases, drawing from previous work of Shoup [Sho94b; Sho95; Sho99] and Bostan, Salvy and Schost [BSS03].

If $\mathbb{F}_{p^m}$ is represented as $\mathbb{F}_p[X]/f(X)$, we define its *monomial basis* as $(1, X, \ldots, X^{m-1})$. The associated *dual basis* is $(X_0^*, X_1^*, \ldots, X_{m-1}^*)$, where X_i^* are defined by

$$\operatorname{Tr}(X^j X_i^*)_k = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

Tr denoting the trace from $\mathbb{F}_{p^m}$ to $\mathbb{F}_p$. Given the polynomial f, conversions between the monomial and the dual basis can be performed very efficiently at a cost of $\tilde{O}(n)$ operations, thus we allow ourselves to switch freely between the two representations.

Now, let $\mathbb{F}_{p^m}, \mathbb{F}_{p^n}$ and $\mathbb{F}_{p^{mn}}$ be represented as in the diagram

$$\begin{array}{ccccc} & & \mathbb{F}_{p^{mn}} = \mathbb{F}_p[Z]/(f \otimes g)(Z) & & \\ & \swarrow & & \searrow & \\ \mathbb{F}_p[X]/f(X) = \mathbb{F}_{p^m} & & & & \mathbb{F}_{p^n} = \mathbb{F}_p[Y]/g(Y) \\ & \searrow & & \swarrow & \\ & & \mathbb{F}_p & & \end{array}$$

with the natural embedding defined by $Z = XY$. We seek an algorithm for the following operation: given $b \in \mathbb{F}_{p^m}$ and $c \in \mathbb{F}_{p^n}$, compute $bc \in \mathbb{F}_{p^{mn}}$. Note that this gives a way to evaluate both embeddings, by fixing either $c = 1$ or $b = 1$.

The key observation is that the product bc is computed as a component-wise product in the dual bases of the respective fields. The embedding algorithm is thus (almost) as

[9] The hypothesis that f, g have coefficients in a finite field is important. For example, only the composed sum is necessarily irreducible if the polynomials have rational coefficients.

simple as converting both b and c to the respective dual bases, multiplying component-by-component, and converting back to the monomial basis of $\mathbb{F}_{p^{mn}}$. We refer to the appendix for further details.

There is another, quite technical, ingredient to the paper, that we have mostly ignored so far. Evaluating embeddings is only half of the job: we also want to evaluate the (partial) inverse maps, or, more generally, projections $\mathbb{F}_{p^{mn}} \to \mathbb{F}_{p^m}$ and $\mathbb{F}_{p^{mn}} \to \mathbb{F}_{p^n}$ that, when composed with the embeddings, yield the identity map.

A general *mantra* states that, whenever a field isomorphism $\phi : k \to K$ can be evaluated at a certain cost, the inverse map ϕ^{-1} can evaluated at the same cost (see [Bri+17, §8.2] for a precise statement). In our specific instance, let $b \in \mathbb{F}_{p^m}$, and let M_b be the matrix of the map $c \mapsto bc$, for any $c \in \mathbb{F}_{p^n}$, in the *dual bases* of $\mathbb{F}_{p^n}$ and $\mathbb{F}_{p^{mn}}$. Direct calculation shows that M_b^t is the matrix of the map $a \mapsto \mathrm{Tr}_{\mathbb{F}_{p^{mn}}/\mathbb{F}_{p^n}}(ab)$ in the *monomial bases* of $\mathbb{F}_{p^{mn}}$ and $\mathbb{F}_{p^n}$; thus, if $\mathrm{Tr}_{\mathbb{F}_{p^{mn}}/\mathbb{F}_{p^n}}(b) = 1$, the matrix M_b^t is a partial inverse to M_b (albeit in a different basis).[10]

To obtain an algorithm from this observation, we use a technique called *transposition principle* [Sho99; BLS03], transforming an algorithm to evaluate M_b into one to evaluate M_b^t *at the same cost*. Said otherwise, we derive an efficient projection algorithm by *transposing* the embedding one.

In conclusion, our techniques allow to compute embeddings and projections of composita in quasi-optimal time. They also allow to evaluate vector space isomorphisms (e.g., $\mathbb{F}_{p^{mn}} \simeq \mathbb{F}_{p^m}^n$ with respect to the monomial or the dual basis) with the best known complexity, although not in quasi-optimal time. By combining these techniques with any technique for primary towers, we obtain a representation for $\bar{\mathbb{F}}_p$, unique if the representation of primary towers is. Unfortunately, since we do not know how to evaluate vector space isomorphisms in quasi-optimal time, we inherit the same unsatisfactory complexity for any computation in $\bar{\mathbb{F}}_p$. Lifting this restriction is one of the problems that torments me.

II.4 Isomorphisms of finite fields

Special (compatible) families of irreducible polynomials are a fascinating topic, however they are not a universal solution. In some circumstances, we may not have the choice of the polynomial representing a field extensions; this happens quite often in computer algebra systems that let the user choose their own polynomial.

Isomorphism algorithms let us lift this restriction: using them, a computer algebra system may internally represent finite fields with special polynomials, while letting the user work with an isomorphic copy defined by a polynomial of their choice. In the next section we shall also see how isomorphism algorithms (actually, embedding algorithms) allow us to represent $\bar{\mathbb{F}}_p$ without using any special polynomial.

The isomorphism problem was first investigated by Lenstra [Len91], who showed that it can be solved in deterministic polynomial time. A thorough review of all known algorithms is given in the paper "Computing Isomorphisms and Embeddings of Finite Fields", written with L. Brieulle, J. Doliskani, J.-P. Flori and É. Schost [Bri+18], and included in the appendix. In the paper, we study the more general *embedding problem* of two finite fields $k \subset K$.

[10] I feel so guilty for shrinking one of my favorite tricks to one single paragraph. Really, go read [Bri+17, §8], please!

	Irreducible polynomials	Isomorphisms
Kummer $g = 0$	Shoup [Sho90; Sho93; Sho94b]	Lenstra [Len91], Allombert [All02a]
Kummer $g = 1$	Couveignes–Lercier [CL13]	Narayanan [Nar18]
Cyclotomic $g = 0$	Adleman–Lenstra [AL86]	Pinch [Pin92], Rains [Rai96]
Cyclotomic $g = 1$	—	Pinch [Pin92], BDDFS [Bri+18]

Table II.1: Main algorithmic ideas to compute irreducible polynomials and isomorphisms of finite fields, by inventors.

Problem II.2 (Embedding description)**.** Given two finite fields $k = \mathbb{F}_p[X]/f(X)$ and $K = \mathbb{F}_p[Y]/g(Y)$, with $\deg f$ dividing $\deg g$, determine elements $\alpha \in k$ and $\beta \in K$ such that $k = \mathbb{F}_p(\alpha)$, and such that there exists an embedding ϕ mapping α to β.

It is easily seen that (α, β) describes an embedding if and only if α and β share the same minimal polynomial. Therefore, the simplest solution, although not the most efficient, consists in taking the class of X for α, and a root of $f(X)$ in K for β.

Given an embedding description (α, β), the associated embedding (and its partial inverse) can be evaluated on any element of k by linear algebra. More efficient techniques, similar to those employed in the previous section for composita, are also available. They are not discussed in the appendix, but they are described in the extended version [Bri+17] of [Bri+18].

The algorithms for embedding description are very similar to those to compute irreducible polynomials. For simplicity, we shall only discuss here isomorphisms, i.e., $k \simeq K$. We refer to the appendix for the general problem.

There essentially exist two big families: the "Kummer family", a generalization of the "Kummer case" for primary towers discussed in Section II.3; and the "Cyclotomic family", a generalization of the Adleman–Lenstra algorithm. Each of these families can come in a "genus 0" flavor, like the Kummer case for primary towers, or in a "genus 1" flavor, like the Couveignes–Lercier construction.[11] In Table II.1 we summarize the known constructions for computing irreducible polynomials and isomorphism descriptions, according to these categories.

We will now describe Kummer-type algorithms first, then cyclotomic-type. The common principle for all of them is to compute a special element, say in $\mathbb{F}_{p^n}$, uniquely defined up to automorphisms of $\mathbb{F}_{p^n}$.

Lenstra–Allombert algorithm. In [Len91], Lenstra proved that the isomorphism problem can be solved in deterministic polynomial time, without focusing much on getting the best possible complexity. In [All02a; All02b], Allombert modified Lenstra's algorithm to obtain an efficient one; in doing so, he replaced some routines with polynomial factorization, thus dropping determinism. In [Bri+18], we give several variants of Allombert's algorithm that are more efficient, both asymptotically and in practice. Allombert has included in PARI/GP both his variants and ours.

The Lenstra–Allombert algorithm is similar in spirit to Shoup's algorithm for computing irreducible polynomials [Sho90; Sho93; Sho94b]. Let k, K be two isomorphic copies of $\mathbb{F}_{p^n}$, the idea is to choose $\alpha \in k$ and $\beta \in K$ to be solutions to Hilbert's theorem 90: these are not uniquely defined up to automorphisms, but at least they are unique *up to a scalar*.

[11] Note that higher genus generalization are possible, but they seldom give efficient algorithms.

Assume, for example, that n divides $p-1$, and fix a n-th root of unity $\zeta_n \in \mathbb{F}_p$. An element $\alpha \in k$ is a solution to Hilbert 90 if and only if $\alpha^p = \zeta_n \alpha$; it is immediate to see that any other solution is of the form $c\alpha$ for $c \in \mathbb{F}_p$. Hence, if $\beta \in K$ is also a solution to Hilbert 90, any isomorphic image of β in k is such that $\alpha/\beta = c \in \mathbb{F}_p$; the problem is then to find this scalar.

Lenstra and Allombert differ in the way the scalar c is found. We describe here Allombert's variant:

1. Compute a n-th root of unity $\zeta_n \in \mathbb{F}_p$;
2. Find solutions $\alpha \in k$ and $\beta \in K$ to Hilbert 90 relative to ζ_n;
3. Compute $a = \alpha^n$ and $b = \beta^n$, they are both in $\mathbb{F}_p$;
4. Compute $c = \sqrt[n]{a/b} \in \mathbb{F}_p$ using a polynomial factoring algorithm;
5. Return α and $c\beta$.

To extend the algorithm to arbitrary n, assume first that p does not divide n. Like in Shoup's algorithm, we adjoin the necessary roots of unity to the base fields: Allombert factors the n-th cyclotomic polynomials Φ_n and extends scalars to $A_n = \mathbb{F}_p(\zeta_n)$, whereas Lenstra uses the unfactored polynomial and constructs the ring $A_n = \mathbb{F}_p[Z]/\Phi_n(Z)$. Then, the same algorithm as above is run in $\mathbb{F}_{p^n} \otimes A_n$, and the result is descended to $\mathbb{F}_{p^n}$ using a projection. Note that, in both cases, $\mathbb{F}_{p^n} \otimes A_n$ is not necessarily a field.

Finally, extensions of degree p^k are dealt with an *additive variant* of the algorithm above, analogous to Shoup's construction based on Artin–Schreier theory. Then, a solution for a generic $n = p^k n'$ is obtained by combining (either by multiplying or by adding) the multiplicative solution for n' and the additive one for p^k.

The dominant cost in the Lenstra–Allombert algorithm is computing the solution to Hilbert 90. In general, $\mathbb{F}_{p^n} \otimes A_n$ is an algebra of degree $O(n^2)$, thus there is no hope to obtain an algorithm better than quadratic. In [Bri+18] we show that it is indeed possible to achieve the optimum, and even lower than that in favorable cases.

Proposition II.3 (Brieulle, D., Doliskani, Flori, Schost [Bri+18])**.** *Let k, K be two extensions of degree r over $\mathbb{F}_p$, where r is a prime power. Let s be the order of p in $(\mathbb{Z}/r\mathbb{Z})^\times$. Let ω be an exponent such that $n \times n$ matrices with coefficients in $\mathbb{F}_p$ can be multiplied using n^ω operations, and let $\mathsf{M}(n)$ be the cost of multiplying polynomials of degree at most n in $\mathbb{F}_p[X]$. Allombert's algorithm computes its output using on average*

- $O(s^{(\omega-1)/2} r^{(\omega+1)/2} \log(r) + \mathsf{M}(r)\log(p))$ *operations if* $s \in O(r^{(\omega-3)/(\omega-5)})$*, or*
- $O(r^{(\omega^2-4\omega-1)/(\omega-5)} + (s + r^{2/(5-\omega)})\mathsf{M}(r)\log(p) + s^{\omega-1} r \log(r)\log(s))$ *operations if* $s \in \Omega(r^{(\omega-3)/(\omega-5)})$ *and* $s \in O(r^{1/(\omega-1)})$*, or*
- $O(\mathsf{M}(r^2)\log^2(r) + \mathsf{M}(r)\log(r)\log(p))$ *operations otherwise.*

Note that the bounds for the first two cases could be improved using algorithms of Kedlaya and Umans [KU11], however we do not consider these algorithms as they are deemed unpractical.

An elliptic curve variant of the Lenstra–Allombert algorithm was recently proposed by Narayanan [Nar18]. Historically, it is the first isomorphism algorithm to achieve a quasi-quadratic complexity, however it is deeply dependent on Kedlaya and Umans' results, thus unlikely to be practical.

Rains' algorithm. Rains' algorithm is an improvement over an idea of Pinch [Pin92]. Rains never published his findings, however his algorithm was eventually implemented in Magma v2.14. The original paper is still unpublished, the only publicly available sources for it being, at the moment, Magma's source code[12], and our paper [Bri+18]. An elliptic curve generalization of the algorithm was also originally proposed by Pinch, and then improved in [Bri+18] using ideas similar to Rains'.

The key idea is similar to the Adleman–Lenstra algorithm. If k, K are two isomorphic copies of $\mathbb{F}_{p^n}$, find an integer ℓ such that $\mathbb{F}_{p^n} \simeq \mathbb{F}_p(\zeta_\ell)$, where ζ_ℓ is an ℓ-th root of unity; then the roots of the cyclotomic polynomial Φ_ℓ generate both k and K over $\mathbb{F}_p$. However, the roots of Φ_ℓ are only uniquely defined (up to isomorphism) if Φ_ℓ is irreducible over $\mathbb{F}_p$, i.e., if $n = \varphi(\ell)$. In order to uniquely define an element of k, K in general, Rains suggested using *Gaussian periods*, i.e., traces of roots of unity in a number field.

Definition II.4 (Gaussian period)**.** Let p be a prime, and let ℓ be a squarefree integer such that $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle p \rangle \times S$ for some S. For any primitive ℓ-th root of unity ζ_ℓ in $\bar{\mathbb{F}}_p$, define the Gaussian period $\eta_p(\zeta_\ell)$ as

$$\eta_p(\zeta_\ell) = \sum_{\sigma \in S} \zeta_\ell^\sigma.$$

It is a classic fact that the periods $\eta(\zeta_\ell)$, as ζ_ℓ runs through the roots of Φ_ℓ, form a normal basis of $\mathbb{F}_p(\zeta_\ell)$; thus $\eta(\zeta_\ell)$ is uniquely defined up to isomorphism. Rains' algorithm follows immediately:

1. Find a *small* ℓ such that $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle p \rangle \times S$, with $\#\langle p \rangle = n$;
2. Take random roots of unity $\zeta_\ell \in k$ and $\zeta'_\ell \in K$;
3. Return the Gaussian periods $\alpha = \eta(\zeta_\ell)$ and $\beta = \eta(\zeta'_\ell)$.

However it may happen that no such *small* ℓ exists. Rains' solution to the problem (identical to Adleman and Lenstra's) is to extend scalars to a small degree auxiliary extension $\mathbb{F}_q = \mathbb{F}_{p^s}$, with $s \wedge n = 1$, and apply the same algorithm to $k \otimes \mathbb{F}_q$ and $K \otimes \mathbb{F}_q$; then the result is descended to an isomorphism of k, K by taking a trace.

A different way to solve the problem is to generalize Gaussian periods to *elliptic periods*. Let $E/\mathbb{F}_p$ be an elliptic curve, and suppose that $\pi | E[\ell]$ acts like a matrix $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \mu \end{smallmatrix}\right)$; suppose that the order in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ of, say, λ is equal to n, then the eigenspace of λ is contained in $E(\mathbb{F}_{p^n})$. By mimicking the definition of Gaussian periods, we can obtain a uniquely defined element of $\mathbb{F}_{p^n}$.

Definition II.5 (Elliptic period)**.** Let $E/\mathbb{F}_p$ be an elliptic curve of j-invariant not 0 or 1728. Let $\ell > 3$ be an Elkies prime for E, λ an eigenvalue of π, and P a point of order ℓ such that $\pi(P) = \lambda P$. Suppose that there is a subgroup S of $(\mathbb{Z}/\ell\mathbb{Z})^\times$ such that

$$(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle \lambda \rangle \times S.$$

Then we define an *elliptic period* as

$$\eta_{\lambda,S}(P) = \begin{cases} \sum_{\sigma \in S/\{\pm 1\}} x([\sigma]P) & \text{if } -1 \in S, \\ \sum_{\sigma \in S} x([\sigma]P) & \text{otherwise,} \end{cases}$$

where $x(P)$ denotes the abscissa of P.

[12] Precisely, in file `package/Ring/FldFin/embed.m`.

It is immediate to modify Rains' algorithm to use elliptic periods instead of Gaussian ones, the freedom in the choice of the curve $E/\mathbb{F}_p$ enabling us to find a smaller ℓ without the need for an auxiliary extension. However, we encounter a problem: elliptic periods do not form normal bases, in general. While it is easy to find examples where elliptic periods are not normal, it is enough for our purposes that they generate $\mathbb{F}_{p^n}$ as a field. Heuristically, this is an extremely likely event, hence, in principle, our elliptic variant of Rains' algorithm has a very tiny failure probability; more detailed statements are given in Section C.5.

However, when we tried to find example inputs on which the elliptic variant fails, we were surprised to find none, despite an extensive search over more than 43 million curves, carefully documented in [Bri+17]. Our failed search has thus led us to state the conjecture below, that elliptic period always generate their field of definition, implying that the elliptic variant of Rains algorithm never fails.

Conjecture II.6 (Brieulle, D., Doliskani, Flori, Schost [Bri+18])**.** Let $E/\mathbb{F}_p$ be an elliptic curve with j-invariant not 0 or 1728; let ℓ be an Elkies prime for E, and $P \in E[\ell]$ a point in the eigenspace of a Frobenius eigenvalue λ for ℓ. Assume that $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle\lambda\rangle \times S$, then the elliptic period $\eta_{\lambda,S}(P)$ generates $\mathbb{F}_p(x(P))$ over $\mathbb{F}_p$.

The complexity of Rains' algorithm is extremely sensitive to how small the integer ℓ is. Unfortunately, even assuming the generalized Riemann hypothesis, bounds on ℓ are very loose, thus provable bounds on the complexity of the algorithm are very bad. However, in practice we expect that $\ell = O(n \log n)$, and we give enough experimental evidence in [Bri+18] to support this heuristic. Assuming this heuristic bound, we can prove that Rains' algorithm runs in average time $\tilde{O}(n^{(\omega+1)/2} + n \log q)$, where ω is the *exponent of linear algebra*, while the elliptic variant runs in time $\tilde{O}(n^2 \log q)$.

These bounds are remarkable: according to them, Rains' original algorithm is the only isomorphism algorithm with subquadratic complexity. However this bound is only heuristic, and experiments show that, both the original algorithm and the elliptic variant, perform faster than Allombert's algorithm only in very limited cases.

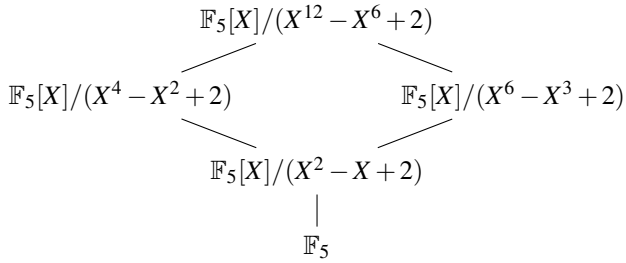
II.5 Lattices of finite fields

We end this chapter coming back to the problem of representing the algebraic closure $\bar{\mathbb{F}}_p$. We saw how special families of polynomials can be used to represent finite extensions of $\mathbb{F}_p$ in a *compatible*, *incremental*, and possibly *unique* way, and thus can also be used to represent the whole $\bar{\mathbb{F}}_p$. At the opposite end of the spectrum we have the possibility of representing finite extensions of $\mathbb{F}_p$ by arbitrary polynomials, and provide compatibility through a general purpose embedding algorithm such as those presented in the previous section.

This approach was first formalized by Bosma, Cannon and Steel, in "Lattices of Compatibly Embedded Finite Fields" [BCS97a]. They implemented it as the default system for Magma; to compute embeddings, they originally used the naive approach based on polynomial factoring, then added Rains' algorithm as an alternative. To this day, Magma still has the most efficient system to represent lattices of arbitrary extensions of $\mathbb{F}_p$.

The Bosma–Cannon–Steel framework maintains a single data structure: a collection (a lattice) of compatible extension fields. We may add new extensions to the lattice, at the user's request, by computing embeddings into each of the fields already present in it, and storing the embedding data along.

However, a naive implementation of this idea may easily get stuck. Take for example $p = 5$, and a lattice containing $\mathbb{F}_{p^2}$, $\mathbb{F}_{p^4}$ and $\mathbb{F}_{p^6}$; assume that the extensions are defined by the polynomials in the diagram below.



Denote by η_2, η_4, η_6 the classes of X in $\mathbb{F}_{p^2}, \mathbb{F}_{p^4}, \mathbb{F}_{p^6}$ respectively, and assume that we have chosen the embeddings $\eta_2 \mapsto \eta_4^2$ and $\eta_2 \mapsto \eta_6^3$. Now, suppose that we want to add $\mathbb{F}_{p^{12}}$ to the lattice. We start by embedding $\mathbb{F}_{p^4} \hookrightarrow \mathbb{F}_{p^{12}}$ with $\eta_4 \mapsto \eta_{12}^{15}$; at this point, we are not anymore free to choose any embedding for $\mathbb{F}_{p^6} \hookrightarrow \mathbb{F}_{p^{12}}$: indeed, choosing $\eta_6 \mapsto \eta_{12}^2$ would imply $\eta_{12}^6 = \eta_2 = \eta_{12}^{30}$, and thus $\eta_{12}^{24} = 1$, which is impossible because all 24th roots of unity are in $\mathbb{F}_{5^2}$.

The *tour de force* by Bosma, Cannon and Steel consists in showing that it is always possible to construct compatible lattices, if one does so carefully. They define six conditions that the lattice must satisfy in order to ensure compatibility; we paraphrase them below:

Uniqueness: For every pair of fields k, K in the lattice, there is at most one embedding $k \hookrightarrow K$.

Reflexivity: Every field is embedded in itself through the identity morphism.

Prime subfield: $\mathbb{F}_p$ is embedded in every field via the canonical embedding.

Invertibility: If $k \simeq K$, then the embeddings $k \hookrightarrow K$ and $K \hookrightarrow k$ are inverse to one another.

Transitivity: For any triple $k \subset K \subset L$ the embeddings $\phi_{k,K} : k \to K$, $\phi_{K,L} : K \to L$, $\phi_{k,L} : k \to L$, are compatible, i.e., $\phi_{k,L} = \phi_{K,L} \circ \phi_{k,K}$.

Intersection: For any triple k, K, L such that $k \subset L$ and $K \subset L$, the subfield $k \cap K$ is also in the lattice and is compatibly embedded in k, K, L.

By maintaining these properties throughout a session, Magma is able to ensure compatibility regardless of the number of finite fields created by the user. If a criticism must be addressed to this data structure, it is on the combinatorial explosion that it entails: for any field in the lattice, the data on embeddings to any other subfield must be stored, thus storage grows quadratically in the number of fields in the lattice. Also, as more and more fields are added, more and more computations must be performed to keep compatibility. Nevertheless, Magma has still today undoubtedly the most efficient system to represent the algebraic closure of a finite field.

II.6 Perspectives

Despite 40 years of research, the question of computing in extensions of $\mathbb{F}_p$ is not closed yet: there is still progress to be made both on the practical and the theoretical side.

Implementations. As we have seen, the only computer algebra systems implementing a generic mechanism to compute in $\bar{\mathbb{F}}_p$ are Magma and SageMath. Neither can be said to be really efficient, but the prize for the fastest and most feature-rich undoubtedly goes to Magma.

The spectrum of available algorithms is nowadays considerably larger than what is used in these two systems, and it would be worth experimenting with them with the goal of beating the current implementations.

With H. Randriam, É. Rousseau and É. Schost, we have started experimenting with the Bosma–Cannon–Steel framework in the new computer algebra system Nemo [Fie+17]. A reproduction of the system in Magma has already been implemented by É. Rousseau[13], and we are now exploring possible ways to improve its efficiency.

Unfortunately, no single catch-all technique emerges from the theory, thus experiments will necessarily have to take many different techniques into account.

After having obtained convincing results, it will be highly desirable to port the techniques that worked best to SageMath. This will be facilitated by the fact that Nemo and SageMath share the same low-level C library in Flint, which is where we are adding the critical parts of the algorithms we test.

Special families. Special families of irreducible polynomials are the most promising option for efficiently representing $\bar{\mathbb{F}}_p$, and the only one granting uniqueness. However, at the moment they have serious drawbacks that make them an unrealistic target for implementation. For one, they are quite complex to implement: the Couveignes–Lercier construction, for example, requires algorithms for counting points of elliptic curves.[14] Finally, when the base field gets relatively large, the Couveignes–Lercier construction becomes too costly, and must be replaced by Shoup's construction, which is not quasi-optimal. Thus, a first goal would be to find a simple algorithm for constructing primary towers, that is reasonably efficient for any parameter.

Even with a good construction for primary towers, we then need to construct composita in order to deal with general extensions. Our construction is reasonably simple and efficient, however it is not optimal when it comes to vector space isomorphisms, and thus it is not optimal for $\bar{\mathbb{F}}_p$ either. A construction for composita with quasi-linear complexity would be a big discovery, and would likely pave the way for a larger adoption of special families. In the meantime, our construction is by far the one with the best available complexity, and is simple enough that it would be worth experimenting with it.

A radically different approach would be to keep the decomposition of $\bar{\mathbb{F}}_p$ as a tensor product of primary towers, i.e., to represent elements as multivariate polynomials modulo an ideal of univariate polynomials. We have already said that known techniques, e.g., Kronecker substitution, incur an exponential penalty in the number of variables, i.e., the number of primary factors. Improving this complexity would be a major breakthrough in computer algebra, with consequences that reach well beyond implementing $\bar{\mathbb{F}}_p$.

On uniqueness and elegance. Uniqueness in families of polynomials is an interesting question, because there is a social aspect to it.

[13] https://github.com/erou/LatticesGF.jl

[14] Realistically, though, a naive algorithm is enough, given that the Couveignes–Lercier algorithm is only practical for relatively small base fields.

Unfortunately, there is no "natural" choice for a unique family of irreducible polynomials. What come closest to a canonical definition of an algebraic closure of a finite field is Conway's construction of the field $\mathbf{On}_2$, which contains $\bar{\mathbb{F}}_2$ as a subfield [Con00]. However, as Lenstra showed [Len77], computing the finite subfields of $\mathbf{On}_2$ is not trivial. Furthermore, no satisfactory generalization of Conway's work is known for general p.

Thus even reasonably simple proposals, such as Conway polynomials (which are mostly unrelated to $\mathbf{On}_2$), require a certain dose of *ad hocness*, such as a "lexicographically constraint".

Any algorithm for computing irreducible polynomials can be artificially turned into a deterministic one by fixing random choices. However, the main motivation for uniqueness is data portability: i.e., the possibility to run the same computation in two computer algebra system (two different systems, the same system but different sessions, or versions, ...), and obtain the same results. Thus, for a unique construction of irreducible polynomials to be accepted, it must be simple enough that all systems can implement it correctly, test it easily, and it must also be popular enough that all systems *want to* implement it.

I am afraid that the Couveignes–Lercier construction does not fit the bill. Shoup's construction would be acceptable, but as long as its advantages over Conway polynomials are not clearly visible, I suspect few developers will be willing to implement it.

Therefore, I am convinced that a simple enough construction, even if not optimal, offering advantages similar to Conway polynomials at a lesser cost, would be extremely valuable to the community.

Isomorphisms. I am sure that the hole in Table II.1 has not gone unnoticed. Before the reader jumps on filling it, let me tell that I do not see how to obtain an interesting algorithm from it.

One obvious idea would be to look for curves $E/\mathbb{F}_p$ with a certain number of points, construct some isogeny of degree ℓ, and use its kernel polynomial to define an irreducible polynomial of the wanted degree (maybe using elliptic periods?). However, this approach would not have a better complexity than the Adleman–Lenstra algorithm, in the same way that the elliptic variant of Rains' algorithm does not have a better complexity than the original one. It would also quite probably require the use of the ℓ-th modular polynomial, since the computations involved are very similar to those for the Explicit isogeny problem we saw in the previous chapter; thus it would probably also be unpractical.

A different approach would use curves defined over number fields. This is probably even more desperate, given how hard it is to find torsion points on them.

Coming to something concrete, now, the obvious breakthrough would be to find an isomorphism algorithm with subquadratic complexity. Rains' algorithm comes close to it, however it requires heuristic hypotheses that are probably out of reach.

Less ambitiously, we may look for practical algorithms. Our experiments show that (one of the variants of) Allombert's algorithm is by far the most practical of all; beating its performance using a different paradigm would be a nice challenge.

Arbitrary extensions. Moving to lattices of arbitrary extensions, we have already discussed the combinatorial explosion that the Bosma–Cannon–Steel data structure suffers from. I am currently interested in taking ideas from isomorphism algorithms,

and applying them to lattices of extensions, in the hope of reducing the amount of information that needs to be stored.

More in detail, we have seen that all isomorphism algorithms are based on the principle of finding some "(almost) uniquely defined" generators for the finite fields. We may imagine more than this: we may hope to find a *lattice of uniquely defined generators* such that generators are "compatible" in some way. This would allow storing only the generators, one per field, and deduce all embeddings from them.

The analogy between isomorphism algorithms and algorithms for irreducible polynomials, shown in Table II.1, also hints at the fact that these *lattices of generators* would be a sort of "half-way" construction, between special families of polynomials and arbitrary lattices, and may potentially lead to new solutions to the uniqueness problem mentioned above.

Limited examples of this idea are easy to produce: think for example of the lattice of roots of unity, defining the "cyclotomic extensions" of $\mathbb{F}_p$. However finding a general construction capable of describing arbitrary extensions seems harder. We are currently exploring this idea in conjunction with the Lenstra–Allombert algorithm, and hope to have interesting results soon.

III Telemetry

In Chapter I we learned how to classify isogenies over finite fields. For ordinary curves, we defined the depth of $E/\mathbb{F}_q$ as the valuation of the conductor $[\mathcal{O}_K : \mathrm{End}(E)]$ at some prime ℓ. We saw how to effectively compute the depth, and how to recognize ascending, descending, and horizontal isogenies. The structure theorems revealed that there are three possible shapes of ℓ-isogeny graphs, only differing in the structure of the surface: a single curve in the Atkin case, a pair of curves in the ramified case, and a *crater* (a cycle) in the Elkies case.

However, we left two questions unanswered: for a given curve $E/\mathbb{F}_q$ and a prime ℓ, how many vertices does its ℓ-isogeny graph contain? And, for a given quadratic imaginary field $\mathbb{Q}(\pi)$ and a prime ℓ, how many distinct ℓ-isogeny graphs are there?

The first question is easy to answer for the Atkin and the ramified case: we know indeed that isogeny volcanoes have height $h = v_\ell(\sqrt{\Delta_\pi/\Delta_K})$, therefore an Atkin volcano contains $((\ell+1)^{h+1}-1)/\ell$ curves, whereas a ramified volcano contains $2(\ell+1)^h$ curves. In order to determine the size of the crater in the Elkies case, and to answer the second question, we will have to resort to the theory of *complex multiplication*. We will then learn that, while heights tend to be "small" (i.e., logarithmic in q), craters tend to be "large" (polynomial in q).

At this point, we will begin studying large isogeny graphs containing isogenies of mixed degree. We will be faced with a new problem that is finding a "short" isogeny path between two curves in a graph, and upon this problem we will build a new cryptographic primitive. We will then turn our attention to supersingular isogeny graphs. The unique properties of the Frobenius endomorphism of supersingular curves will allow us to build a much more efficient cryptographic primitive, known by the name of CSIDH. Finally, by pushing the study of supersingular graphs further, we will get to the primitive known as SIDH, the building block of SIKE [SIKE], one of the candidates to the NIST call for post-quantum public key encryption [Nat16].

III.1 Complex multiplication

Let $\mathcal{O}$ be an order in a quadratic imaginary field $K = \mathbb{Q}(\sqrt{-D})$, we say that an elliptic curve E has *complex multiplication by* $\mathcal{O}$ (or, in short, *CM* by $\mathcal{O}$) if $\mathrm{End}(E) \simeq \mathcal{O}$. We have already seen that all ordinary elliptic curves over finite fields have complex multiplication, with $K = \mathbb{Q}(\pi)$.

Supersingular curves have the similar property $\mathrm{End}_{\mathbb{F}_q}(E) \simeq \mathcal{O} \subset \mathbb{Q}(\sqrt{-q})$, whenever q is an odd power of a prime $p > 3$. Strictly speaking, no supersingular curve should be said to have complex multiplication, because the ring of endomorphisms defined over the algebraic closure is larger (more on this later); we will nevertheless use the name for this special case.

We now present a group action on the set of all curves having complex multiplication by a fixed order $\mathcal{O}$. We will denote by $\mathrm{Ell}_q(\mathcal{O})$ the set of isomorphism classes over $\bar{\mathbb{F}}_q$ of curves with complex multiplication by $\mathcal{O}$, and we will assume that it is non-empty.

Let $\mathfrak{a}$ be an invertible ideal in $\mathrm{End}(E) \simeq \mathcal{O}$, of norm coprime to q, and define the $\mathfrak{a}$-*torsion* subgroup of E as

$$E[\mathfrak{a}] = \{P \in E(\bar{\mathbb{F}}_q) \mid \sigma(P) = 0 \text{ for all } \sigma \in \mathfrak{a}\}.$$

This subgroup is the kernel of a separable isogeny $\phi_\mathfrak{a} : E \to E/E[\mathfrak{a}]$; it can be proven that $\phi_\mathfrak{a}$ is horizontal, and that its degree is the *norm* of $\mathfrak{a}$. By composing with an appropriate

purely inseparable isogeny, the definition of $\phi_{\mathfrak{a}}$ is easily extended to invertible ideals of any norm.

Writing $\mathfrak{a} \cdot E$ for the isomorphism class of the image of $\phi_{\mathfrak{a}}$, we get an action $\cdot : \mathscr{I}(\mathscr{O}) \times \mathrm{Ell}_q(\mathscr{O}) \to \mathrm{Ell}_q(\mathscr{O})$ of the group of invertible ideals of $\mathscr{O}$ on $\mathrm{Ell}_q(\mathscr{O})$. It is then apparent that endomorphisms of E correspond to principal ideals in $\mathscr{O}$, and act trivially on $\mathrm{Ell}_q(\mathscr{O})$. Recall that the *class group* $\mathrm{Cl}(O)$ is defined as the quotient of $\mathscr{I}(\mathscr{O})$ by the subgroup $\mathscr{P}(\mathscr{O})$ of principal ideals; since the above action factors through $\mathscr{P}(\mathscr{O})$, it natural to consider the induced action of $\mathrm{Cl}(\mathscr{O})$ on $\mathrm{Ell}_q(\mathscr{O})$. The main theorem of complex multiplication states that this action is *simply transitive*.

Theorem III.1 (Complex multiplication)**.** *Let $\mathbb{F}_q$ be a finite field, $\mathscr{O} \subset \mathbb{Q}(\sqrt{-D})$ an order in a quadratic imaginary field, and $\mathrm{Ell}_q(\mathscr{O})$ the set of $\bar{\mathbb{F}}_q$-isomorphism classes of curves with complex multiplication by $\mathscr{O}$.*

Assume $\mathrm{Ell}_q(\mathscr{O})$ is non-empty, then it is a principal homogeneous space *for the class group $\mathrm{Cl}(\mathscr{O})$, under the action*

$$\begin{aligned} \mathrm{Cl}(\mathscr{O}) \times \mathrm{Ell}_q(\mathscr{O}) &\longrightarrow \mathrm{Ell}_q(\mathscr{O}), \\ (\mathfrak{a}, E) &\longmapsto \mathfrak{a} \cdot E \end{aligned}$$

defined above.

Being a principal homogeneous space means that, for any fixed base point $E \in \mathrm{Ell}_q(\mathscr{O})$, there is a bijection

$$\begin{aligned} \mathrm{Cl}(\mathscr{O}) &\longrightarrow \mathrm{Ell}_q(\mathscr{O}) \\ \text{Ideal class of } \mathfrak{a} &\longmapsto \text{Isomorphism class of } \mathfrak{a} \cdot E. \end{aligned}$$

Recall that $\mathrm{Cl}(\mathscr{O})$ is abelian and finite, and that its order is called the *class number* of $\mathscr{O}$, and denoted by $h(\mathscr{O})$. We have just proven that $\#\mathrm{Ell}_q(\mathscr{O}) = h(\mathscr{O})$, and we also have answered both questions we had asked at the beginning of the chapter.

Corollary III.2. *Let $\mathscr{O}$ be a quadratic imaginary order, and assume that $\mathrm{Ell}_q(\mathscr{O})$ is non-empty. Let ℓ be a prime such that $\mathscr{O}$ is ℓ-maximal, i.e., such that ℓ does not divide the conductor of $\mathscr{O}$. All ℓ-isogeny volcanoes of curves in $\mathrm{Ell}_q(\mathscr{O})$ are isomorphic. Furthermore, one of the following is true.*

(0) If the ideal (ℓ) is prime in $\mathscr{O}$, then there are $h(\mathscr{O})$ distinct ℓ-isogeny volcanoes of Atkin type, with surface in $\mathrm{Ell}_q(\mathscr{O})$.

(1) If (ℓ) is ramified in $\mathscr{O}$, i.e., if it decomposes as a square $\mathfrak{l}^2$, then there are $h(\mathscr{O})/2$ distinct ℓ-isogeny volcanoes of ramified type, with surface in $\mathrm{Ell}_q(\mathscr{O})$.

(2) If (ℓ) splits as a product $\mathfrak{l} \cdot \hat{\mathfrak{l}}$ of two distinct prime ideals, then there are $h(\mathscr{O})/n$ distinct ℓ-isogeny volcanoes of Elkies type, with craters in $\mathrm{Ell}_q(\mathscr{O})$ of size n, where n is the order of $\mathfrak{l}$ in $\mathrm{Cl}(\mathscr{O})$.

Like in Chapter I, we are mostly interested in Elkies volcanoes. We already saw that if $\pi | T_\ell(E)$ diagonalizes as $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \mu \end{smallmatrix}\right)$ with $\lambda \neq \mu$, to each eigenvalue we can associate a direction on the crater of the ℓ-volcano. The same phenomenon can be observed through complex multiplication: associate to λ and μ the prime ideals $\mathfrak{a} = (\pi - \lambda, \ell)$ and $\hat{\mathfrak{a}} = (\pi - \mu, \ell)$, both of norm ℓ; then $E[\mathfrak{a}]$ is the eigenspace of λ, and $E[\hat{\mathfrak{a}}]$ that of μ. Because $\mathfrak{a}\hat{\mathfrak{a}} = \hat{\mathfrak{a}}\mathfrak{a} = (\ell)$, the ideal classes $\mathfrak{a}$ and $\hat{\mathfrak{a}}$ are the inverse of one another in

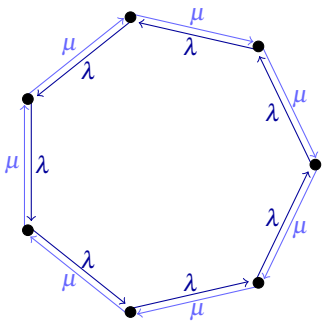


Figure III.1: An isogeny cycle for an Elkies prime ℓ, with edge directions associated with the Frobenius eigenvalues λ and μ.

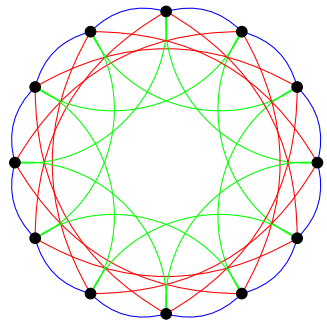


Figure III.2: Graph of horizontal isogenies on 12 curves, with isogenies of three different degrees (represented in different colors).

$\mathrm{Cl}(\mathcal{O})$, therefore the isogenies $\phi_{\mathfrak{a}} : E \to \mathfrak{a} \cdot E$ and $\phi_{\hat{\mathfrak{a}}} : \mathfrak{a} \cdot E \to E$ are dual to one another (up to isomorphism).

We see, once again, that the eigenvalues λ and μ define two opposite directions on the ℓ-isogeny crater, independent of the starting curve, as shown in Figure III.1. The size of the crater is the order of $(\pi - \lambda, \ell)$ in $\mathrm{Cl}(\mathcal{O})$, and the set $\mathrm{Ell}_q(\mathcal{O})$ is partitioned into craters of equal size.

For the rest of the chapter, we will focus mostly on craters of isogeny volcanoes, with horizontal isogenies. In some cases, we will also assume that volcanoes have height 0, so that the crater is the whole graph; for obvious reasons, these are also called *isogeny cycles* in the literature [CM94].

III.2 Quaternion algebras

Supersingular curves are generally not covered by the theory of complex multiplication. For most of them, indeed, the Frobenius endomorphism acts like an element of $\mathbb{Z}$, instead of acting like a "complex multiplier".

Supersingular curves are defined by the fact that multiplication by p is purely inseparable, i.e., $E[p]$ is trivial. This implies that the curve $E^{(p^2)}$ is isomorphic to E, and thus that both are isomorphic to a curve defined over $\mathbb{F}_{p^2}$.

If $E/\mathbb{F}_{p^2}$ is a supersingular curve, its Frobenius endomorphism must satisfy $\pi^2 - t\pi + p^2 = 0$, with t a multiple of p; hence, by Hasse's theorem, $t \in \{0, \pm p, \pm 2p\}$. The cases $t \in \{0, \pm p\}$ only happen for a very limited number of curves with j-invariant 0 or 1728; we are thus mostly interested in the case $t = \pm 2p$, i.e., $\pi = \pm p$. In this case, $\pi | T_\ell(E)$ acts like a scalar matrix for any $\ell \neq p$, hence, by Tate's theorem, $\mathrm{End}(E) \otimes \mathbb{Q}_\ell$ is isomorphic to the full space of 2×2 matrices over $\mathbb{Q}_\ell$. With a little more work, we can prove that $\mathrm{End}(E) \otimes \mathbb{Q}$ is isomorphic to the quaternion algebra $B_{p,\infty}$ ramified at p and at infinity.

With more effort, we can prove that $\mathrm{End}(E)$ is isomorphic to a *maximal* order $\mathcal{O} \subset B_{p,\infty}$. Like the CM case, isogenies are in correspondence with (left) ideals of $\mathcal{O}$. Unlike the CM case, $B_{p,\infty}$ has more than one maximal order, and there is no concept of *depth*, thus no ascending, descending or horizontal isogenies.

More precisely, let $\mathfrak{a} \subset B_{p,\infty}$ a lattice, the *left order* of $\mathfrak{a}$ is the ring $\mathcal{O}(\mathfrak{a}) = \{x \in B_{p,\infty} \mid x\mathfrak{a} \subset \mathfrak{a}\}$. Two lattices $\mathfrak{a}, \mathfrak{b}$ are said to be *right isomorphic* if $\mathfrak{a} = \mathfrak{b}x$ for some $x \in B_{p,\infty}$. If $\mathcal{O} \subset B_{p,\infty}$ is an order, $\mathfrak{a}$ is called a *left ideal* of $\mathcal{O}$ if $\mathcal{O} \subset \mathcal{O}(\mathfrak{a})$; the *left class set* $\mathrm{Cl}(\mathcal{O})$ is the set of right ideal classes of left ideals of $\mathcal{O}$. The order $\#\mathrm{Cl}(\mathcal{O})$ only depends on the quaternion algebra, and is called the *class number* of $B_{p,\infty}$. Analogous definitions can be given by swapping left and right; we refer to [Voi18, Chapter 42] for more properties and definitions.

Like in the CM case, the set $\mathrm{Cl}(\mathcal{O})$ is in bijection with the vertex set of a supersingular graph.

Theorem III.3. *Let $B_{p,\infty}$ be the quaternion algebra ramified at p and infinity, and let $\mathcal{O} \subset B_{p,\infty}$ be a maximal order. Let E_0/F_{p^2} be a supersingular elliptic curve with $\mathrm{End}(E_0) \simeq \mathcal{O}$.*

1. *The number of isomorphism classes of supersingular elliptic curves is equal to the class number of $B_{p,\infty}$.*
2. *There is a one-to-one correspondence $\mathfrak{a} \mapsto \mathfrak{a} \cdot E_0$ between $\mathrm{Cl}(\mathcal{O})$ and the set of isomorphism classes of supersingular elliptic curves, such that $\mathrm{End}(\mathfrak{a} \cdot E_0)$ is isomorphic to the right order of $\mathfrak{a}$.*

This theorem can be turned into an equivalence of categories, see [Koh96, Theorem 45]. Thanks to the Eichler mass formula, we obtain the exact size of the isogeny class.

Corollary III.4. *The number of isomorphism classes of supersingular elliptic curves is equal to*

$$\left\lfloor \frac{p}{12} \right\rfloor + \begin{cases} 0 & \text{if } p = 1 \mod 12, \\ 1 & \text{if } p = 5, 7 \mod 12, \\ 2 & \text{if } p = 11 \mod 12. \end{cases}$$

We thus have a bound on the size of a supersingular isogeny graph over $\mathbb{F}_{p^2}$. Since the Frobenius acts like a scalar, all isogenies are defined over $\mathbb{F}_{p^2}$, hence supersingular ℓ-isogeny graphs are necessarily $(\ell+1)$-regular. In the next section we will learn that the supersingular ℓ-isogeny graph has a unique connected component.

III.3 Expander graphs from isogenies

We are now going to introduce new families of isogeny graphs suitable for cryptographic use. We will want them to somehow "behave like large random graphs", while at the same time having a strong algebraic structure: the first is needed for security, the second to produce complex protocols such as key exchange.

The random-like properties of isogeny graphs are typically expressed in terms of *expansion*. An undirected graph on n vertices has n real eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$, and, if the graph is k-regular, it can be proven that $k = \lambda_1 \geq \lambda_n \geq -k$. Because of this equality, λ_1 is called the *trivial eigenvalue*. An *expander graph* is a k-regular graph such that its non-trivial eigenvalues are bounded away, in absolute value, from k. We recall here some basic facts about expanders; for an in depth review, see [Gol11; Tao11].

Definition III.5 (Expander graph)**.** Let $\varepsilon > 0$ and $k \geq 1$. A k-regular graph is called a (one-sided) *ε-expander* if

$$\lambda_2 \leq (1 - \varepsilon)k;$$

and a *two-sided ε-expander* if it also satisfies

$$\lambda_n \geq -(1-\varepsilon)k.$$

A sequence $G_i = (V_i, E_i)$ of k-regular graphs with $\#V_i \to \infty$ is said to be a one-sided (resp. two-sided) *expander family* if there is an $\varepsilon > 0$ such that G_i is a one-sided (resp. two-sided) ε-expander for all sufficiently large i.

Theorem III.6 (Ramanujan graph)**.** *Let $k \geq 1$, and let G_i be a sequence of k-regular graphs. Then*

$$\max(|\lambda_2|, |\lambda_n|) \geq 2\sqrt{k-1} - o(1),$$

as $n \to \infty$. A graph such that $|\lambda_j| \leq 2\sqrt{k-1}$ for any λ_j except λ_1 is called a Ramanujan graph*.*

Two related properties of expander graphs are relevant to us. First, they have *short diameter*: as $n \to \infty$ the diameter of an expander is bounded by $O(\log n)$, with the constant depending only on k and ε. Second, expanders have *rapidly mixing walks*: loosely speaking, the next proposition says that random walks of length close to the diameter terminate on any vertex with probability close to uniform.

Proposition III.7 (Mixing theorem ([JMV09]))**.** *Let $G = (V, E)$ be a k-regular two-sided ε-expander. Let $F \subset V$ be any subset of the vertices of G, and let v be any vertex in V. Then a random walk of length at least*

$$\frac{\log(\#F^{1/2}/(2\#V))}{\log(1-\varepsilon)}$$

starting from v will land in F with probability at least $\#F/(2\#V)$.

The walk length in the mixing theorem is also called the *mixing length* of the expander graph.

Random regular graphs typically make good expanders, but only a handful of deterministic constructions is known, most of them based on Cayley graphs [LPS88; Chu89; Gol11].

Definition III.8 (Cayley graph)**.** Let G be a group and $S \subset G$ be a symmetric subset (i.e., $s \in S$ implies $s^{-1} \in S$). The *Cayley graph* of (G, S) is the undirected graph whose vertices are the elements of G, and such that there is an edge between g and sg if and only if $s \in S$.

In our case, we will construct a Cayley graph by "gluing many isogeny cycles together": we take $\mathrm{Ell}_q(\mathcal{O})$ as vertex set, select a subset of ideals $S \subset \mathrm{Cl}(\mathcal{O})$ represented by isogenies of bounded prime degree, and draw an edge between E and $\mathfrak{a} \cdot E$ for any $\mathfrak{a} \in S$. This graph is called the *Schreier graph* of $(\mathrm{Cl}(\mathcal{O}), S, \mathrm{Ell}_q(\mathcal{O}))$, and is isomorphic to the Cayley graph of $(\mathrm{Cl}(\mathcal{O}), S)$; an example is shown in Figure III.2.

Theorem III.9 (Jao, Miller, Venkatesan [JMV09])**.** *Let $\mathcal{O}$ be a quadratic imaginary order, and assume that $\mathrm{Ell}_q(\mathcal{O})$ is non-empty. Let $\delta > 0$, and define the graph G on $\mathrm{Ell}_q(\mathcal{O})$ where two vertices are connected whenever there is a horizontal isogeny between them of prime degree bounded by $O((\log q)^{2+\delta})$.*

Then G is a regular graph and, under the generalized Riemann hypothesis for the characters of $\mathrm{Cl}(\mathcal{O})$, there exists an ε independent of $\mathcal{O}$ and q such that G is a two-sided ε-expander.

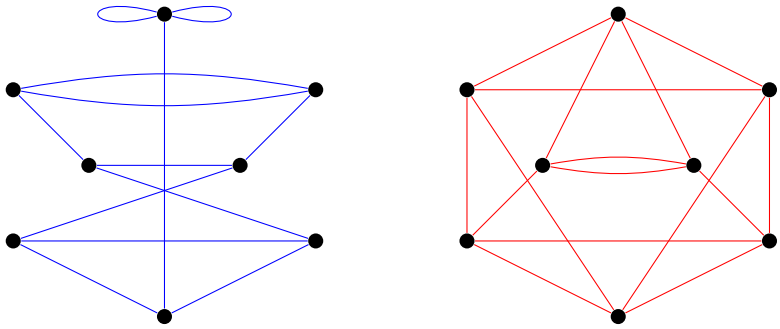


Figure III.3: Supersingular isogeny graphs of degree 2 (left, blue) and 3 (right, red) on $\mathbb{F}_{97^2}$.

The theorem is readily generalized to supersingular curves and isogenies defined over $\mathbb{F}_p$.

A radically different construction of expander graphs is given by graphs of supersingular curves *defined over* $\mathbb{F}_{p^2}$ with ℓ-isogenies, for a *single* prime $\ell \neq p$. Two examples of such graphs are shown in Figure III.3. This construction is related to LPS graphs [LPS88; Lub94; Cos+18], but is not isomorphic to a Cayley graph.

Theorem III.10 (Mestre [Mes86], Pizer [Piz90; Piz98])**.** *Let $\ell \neq p$ be two primes. The ℓ-isogeny graph of supersingular curves in $\bar{\mathbb{F}}_p$, is connected, $(\ell+1)$-regular, and has the Ramanujan property.*

Both of these isogeny graphs will be used in the next sections to build key exchange protocols. For reasons that will be apparent soon, there will only be a mild connection between the expansions properties of the graphs and the security of the protocols: the expansion theorems will mostly serve as a blueprint for devising good cryptosystems, but will have no provable impact.

III.4 Key exchange from CM graphs

The first isogeny-based protocol was introduced by Couveignes during a talk at the École Normale Superieure in 1997, although it was only published ten years later in [Cou06]; independently, Rostovtsev and Stolbunov proposed similar protocols in [RS06; Sto10]. Couveignes' key exchange protocol was presented in a more general setting, applying to any principal homogeneous space satisfying some cryptographic properties.

Recall that a principal homogeneous space (PHS) for a group G is a set X with an action of G on X such that for any $x, x' \in X$, there is a unique $g \in G$ such that $g \cdot x = x'$. Equivalently, the map $\varphi_x : g \mapsto g \cdot x$ is a bijection between G and X for any $x \in X$. Couveignes defines a *hard homogeneous space* (HHS) to be a PHS where the action of G on X is efficiently computable, but inverting the isomorphism φ_x is computationally hard for any x.

Any HHS X for an abelian group G can be used to construct a key exchange based on the hardness of inverting φ_x: the system parameters are a HHS (G, X), and a starting point $x_0 \in X$; a secret key is a random element $g \in G$, and the associated public key is $g \cdot x_0$. If Alice and Bob have keypairs (g_A, x_A) and (g_B, x_B), respectively, then the

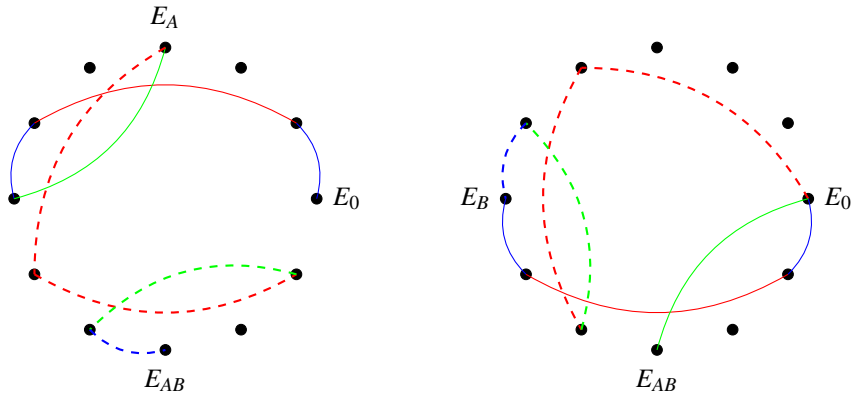


Figure III.4: Example of key exchange on the isogeny graph of Figure III.2. Alice's path is represented by continuous lines, Bob's path by dashed lines. On the left, Bob computes the shared secret starting from Alice's public data. On the right, Alice does the analogous computation.

commutativity of G lets them derive a shared secret

$$g_A \cdot x_B = g_A \cdot g_B \cdot x_0 = g_B \cdot g_A \cdot x_0 = g_B \cdot x_A.$$

The analogy with classic group-based Diffie–Hellman is evident.

Couveignes suggested to use $\mathrm{Ell}_q(\mathscr{O})$ as an instance of a HHS: the system parameters are a starting curve $E/\mathbb{F}_q$, and the associated class group $\mathrm{Cl}(\mathscr{O})$; the secret keys are random elements of $\mathrm{Cl}(\mathscr{O})$, and public keys are j-invariants of curves in $\mathrm{Ell}_q(\mathscr{O})$. However, given a generic element of $\mathrm{Cl}(\mathscr{O})$, the best algorithm [JS10] to evaluate its action on $\mathrm{Ell}_q(\mathscr{O})$ has subexponential complexity in q, making the protocol infeasible.

Instead, following Rostovtsev and Stolbunov [RS06], we may choose to represent elements of $\mathrm{Cl}(\mathscr{O})$ in a way that makes it easy to evaluate the group action. We fix a set S of ideals in $\mathrm{Cl}(\mathscr{O})$ of small degree, possibly in such a way that the associated Cayley graph is an expander. Instead of sampling uniformly random elements of $\mathrm{Cl}(\mathscr{O})$, we sample random walks in the Schreier graph of $(\mathrm{Cl}(\mathscr{O}), S, \mathrm{Ell}_q(\mathscr{O}))$. The walks can be computed efficiently as a composition of small degree isogenies, and, if they are long enough, they approach the uniform distribution on $\mathrm{Ell}_q(\mathscr{O})$. The protocol is illustrated in Figure III.4.

Towards a practical key exchange. Even with these adjustments, the protocol is far from practical: Stolbunov managed to run a 108 bit secure implementation in around 5 minutes [Sto12]. To understand why, let's see how a random element of $\mathrm{Cl}(\mathscr{O})$ is sampled and the group action evaluated. We have a set S of prime ideals of $\mathscr{O}$, represented as $(\pi - \lambda, \ell)$ for some eigenvalue λ modulo a prime ℓ. A secret key corresponds to a product of ideals in S:

$$\mathfrak{s} = \prod_{\mathfrak{a}_i \in S} \mathfrak{a}_i^{e_i}. \tag{III.1}$$

For simplicity, we may assume that the exponents e_i are taken in a box $[-B, B]$,[1] then the size of the key space is at most $(2B+1)^{\#S}$.

[1] Negative values represent the dual direction to $(\pi - \lambda, \ell)$, associated to the ideal $(\pi - \mu, \ell)$.

On the other hand, evaluating the action of $\mathfrak{s}$ requires computing at most $\#S \cdot B$ isogenies. We see that, for a fixed set S, increasing B only increases the key space polynomially, while it also increases the running time linearly. On the other hand, for a fixed B, increasing $\#S$ exponentially increases the key space, while it only increases the running time linearly. Thus, to strike a balance between security and running time, we need to use a fairly large set S: values in the hundreds are typical for $\#S$, and all ideals in S must have different (prime) norms to avoid duplicates. Hence, evaluating the action of $\mathfrak{s}$ implies computing up to $\#S \cdot B$ isogenies of degrees as large as a few thousands!

What algorithms do we have at our disposal to compute these isogenies? We have a curve E, a prime ℓ and a *direction* $\pi - \lambda$. Without further assumptions, we have an instance of the Explicit isogeny problem: we want to enumerate the isogenies of degree ℓ, and choose the one that is horizontal of direction $\pi - \lambda$. We are thus stuck with Elkies' or Couveignes' algorithm, both requiring to evaluate and factor the modular polynomial Φ_ℓ in the first place. It is no surprise then that evaluating one $\mathrm{Cl}(\mathscr{O})$-action takes several minutes.

Is it possible to do better? One idea that comes to mind is to use Vélu's formulas instead. This idea was explored in "Towards practical key exchange from ordinary isogeny graphs", written with J. Kieffer and B. Smith [DKS18], and included in the appendix to this document. Suppose, for example, that $\pi|E[\ell]$ acts like $\left(\begin{smallmatrix} 1 & 0 \\ 0 & \mu \end{smallmatrix}\right)$, with $\mu \neq 1$. In this case, there is an easily recognizable direction associated to the eigenvalue 1: the corresponding eigenspace is the cyclic group of rational ℓ-torsion points. A point in this eigenspace can be computed by taking a random point in $E(\mathbb{F}_q)$, and multiplying it by $\#E/\ell$: there is a $(\ell-1)/\ell$ chance that the result is not zero, and can thus be used to compute the ℓ-isogeny of direction $\pi - 1$ using Vélu's formulas.

We can do even better. Suppose that $\pi|E[\ell]$ acts like $\left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$, then both directions are recognizable: $\pi - 1$ is obtained like before, while $\pi + 1$ corresponds to the rational ℓ-torsion subgroup of a *quadratic twist*[2] of E. More generally, we can use primes such that $\pi|E[\ell]$ acts like $\left(\begin{smallmatrix} -\lambda & 0 \\ 0 & \lambda^{-1} \end{smallmatrix}\right)$: then, if r is the order of $\lambda \pmod{\ell}$, one direction is identified with the rational ℓ-torsion subgroup of $E(\mathbb{F}_{q^r})$, and the other one with that of a quadratic twist. If r is not too large, this approach is still faster than using Elkies' algorithm.

At any rate, the constraints we are putting on π force three conditions:

1. $q = -1 \mod \ell$,
2. $\left(\frac{\Delta_\pi}{\ell}\right) = 1$,
3. the roots of $\pi^2 - t\pi + q \mod \ell$ have small multiplicative order,

and this for each of the primes ℓ we want to include in the set S.

The first condition is easy to fulfill: choose a prime $q = f \cdot \prod_i \ell_i - 1$ for some cofactor f. The other two are much harder, because they essentially require finding a curve $E/\mathbb{F}_q$ with a specific trace t. The best technique at our disposal consists in taking random curves $E/\mathbb{F}_q$ and computing $\#E$, until a suitable one is found.

In [DKS18] we go at great length optimizing the search for a good curve, using *modular curves* and an *early abort* variation on the SEA point counting algorithm. Despite all our efforts, the search is still extremely hard, and the best curve we could

[2] A *quadratic twist* is a curve isomorphic to E over $\mathbb{F}_{q^2}$, hence it represents the same point in the isogeny graph.

find in 17,000 CPU-hours satisfies the constraints above for only 12 primes, plus relaxed constraints for 11 other primes.

The final result is still disappointing: a 128-bit secure key exchange that runs in about 5 minutes.

CSIDH. Somehow, we have not been bold enough. What if we asked even more stringent constraints?

For example, we may ask that $\pi|E[\ell] = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$ for all primes in S. This forces the trace t of π to be $0 \pmod{\ell}$. If we impose this constraint for enough primes, because of Hasse's theorem, we actually force it for *every* prime. Thus we end up with a trace zero supersingular curve.

We may be tempted[3] to quickly dismiss this case, because supersingular curves do not have complex multiplication. However, as already mentioned in Section III.1, Delfs and Galbraith [DG16] showed that if we restrict to curves, endomorphisms and isogenies defined over a prime field $\mathbb{F}_p$, we have a perfect clone of the ordinary complex multiplication case. Indeed, if E is a supersingular curve defined over a prime field $\mathbb{F}_p$ with $p > 3$, it has trace zero and its endomorphism ring is isomorphic to either $\mathbb{Z}[\sqrt{-p}]$ or $\mathbb{Z}[(1+\sqrt{-p})/2]$. For such a curve, every prime ℓ such that $\left(\frac{-4p}{\ell}\right) = 1$ is an Elkies prime, with eigenvalues equal to $\pm\sqrt{-p}$.

In [Cas+18], Castryck, Lange, Martindale, Panny and Renes introduce CSIDH[4]: a variant of the Couveignes–Rostovtsev–Stolbunov system where all isogenies can be computed using Vélu's formulas. CSIDH uses a prime p of the form $4 \cdot \prod_i \ell_i - 1$, and a supersingular curve $E/\mathbb{F}_p$ as starting point, so that $\pi|E[\ell_i] = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$ for all ℓ_i. By cleverly optimizing computations, they achieve a key-exchange at the 128 bits security level in only 0.1 seconds.

Now, I would love to add [Cas+18] to the appendix, but it seems that the rules do not allow me to. I strongly encourage the reader to look for the paper online and read it for themselves.

In the next section we are going to present another key exchange protocol based on supersingular isogeny graphs. The graph structure will be radically different, but Vélu's formulas will still play a crucial role for its performance.

III.5 Key exchange from supersingular graphs

In the previous section we saw how supersingular curves allowed us to go from a dramatically slow protocol to a fairly efficient one. The upshot is the following: we can control the group structure of supersingular curves simply by controlling the order of the base field $\mathbb{F}_q$; this lets us choose curves with many rational points of small order, which in turn can be used to construct small degree isogenies via Vélu's formulas. Ultimately, specially crafted supersingular curves let us navigate their isogeny graph very efficiently.

Can we apply the same principle to the Ramanujan graphs of Theorem III.10? This is the idea behind the two papers "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies", written with D. Jao and J. Plût [JD11; DJP14], the second of which is included in the appendix. In this section we will briefly describe the ideas behind this protocol, that has come to be known as SIDH[5].

[3] As I was.

[4] Pronounced "sea-side".

[5] An acronym for Supersingular Isogeny Diffie Hellman

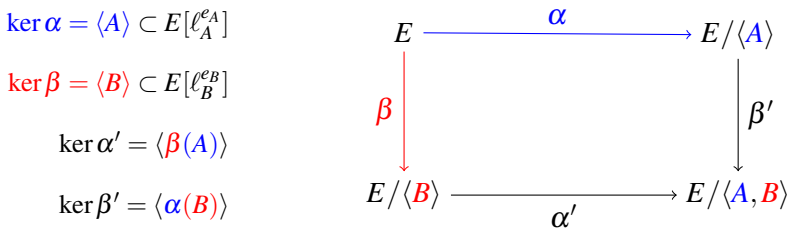


Figure III.5: Commutative isogeny diagram constructed from Alice's and Bob's secrets. Quantities known to Alice are drawn in blue, those known to Bob are drawn in red.

SIDH uses supersingular curves $E/\mathbb{F}_{p^2}$ with trace $\pm 2p$, for a specially chosen p.[6] For these curves $\pi|E[\ell] = \pm\left(\begin{smallmatrix} p & 0 \\ 0 & p \end{smallmatrix}\right)$ for any $\ell \neq p$, and there are exactly $\ell+1$ isogenies of degree ℓ.

Compared to the complex multiplication case, graphs of supersingular isogenies have two attractive features. First, one isogeny degree is enough to obtain an expander graph: this allows us to use isogenies of a single small prime degree, e.g., 2 or 3, instead of many small prime degrees up to the thousands. Second, there is no action of an abelian group, such as $\mathrm{Cl}(\mathcal{O})$, on them: we will see in the next section how this thwarts attacks by quantum computers.

The key idea of SIDH is to let Alice and Bob take random walks in two distinct ℓ-isogeny graphs on the same vertex set of all supersingular j-invariants defined over $\mathbb{F}_{p^2}$. We will denote by ℓ_A and ℓ_B the isogeny degrees used by Alice and Bob respectively. Figure III.3 shows a toy example of such graphs, where $p = 97$, $\ell_A = 2$ and $\ell_B = 3$.

Like in CSIDH, we want to be able to evaluate ℓ-isogenies using Vélu's formulas, thus we need $p = \pm 1 \pmod{\ell}$. However, this is not enough to define a key exchange protocol, as we shall see. Instead, we will use Vélu's formulas to evaluate an isogeny of degree ℓ^e, for some large exponent e, all at once. Therefore, we select a prime of the form $p \mp 1 = \ell_A^{e_A}\ell_B^{e_B} f$, where e_A and e_B are exponents to be determined and f is a small cofactor, so that $E/\mathbb{F}_{p^2}$ contains the full subgroups $E[\ell_A^{e_A}]$ and $E[\ell_B^{e_B}]$. Typical values are $p = 2^{250}3^{159} - 1$ or $p = 2^{372}3^{239} - 1$ (see [SIKE]).

The protocol now proceeds similarly to the Couveignes–Rostovtsev–Stolbunov key exchange: Alice chooses a secret walk of length e_A in the ℓ_A-isogeny graph; this is equivalent to her choosing a secret cyclic subgroup $\langle A\rangle \subset E[\ell_A^{e_A}]$. Bob does the same in the ℓ_B-isogeny graph, choosing a secret $\langle B\rangle \subset E[\ell_B^{e_B}]$. Then, there is a well defined subgroup $\langle A\rangle + \langle B\rangle = \langle A, B\rangle$, defining an isogeny to $E/\langle A,B\rangle$. Since we have taken care to choose $\ell_A \neq \ell_B$, the group $\langle A,B\rangle$ is cyclic of order $\ell_A^{e_A}\ell_B^{e_B}$. This is illustrated in Figure III.5.

After Alice and Bob have computed their respective secrets $\langle A\rangle$ and $\langle B\rangle$, we need them to exchange enough information to both compute $E/\langle A,B\rangle$ (up to isomorphism). However, publishing $E/\langle A\rangle$ and $E/\langle B\rangle$ does not give enough information to the other party, and the diagram in Figure III.5 shows no way by which they could compute $E/\langle A,B\rangle$ without revealing their secrets.

We solve this problem by a very peculiar trick, which sets SIDH apart from other isogeny based protocols. The idea is to let Alice and Bob publish some additional information to help each other compute the shared secret. Let us summarize what are

[6] Note that this case includes trace zero curves $E/\mathbb{F}_p$, after extending scalars to $\mathbb{F}_{p^2}$.

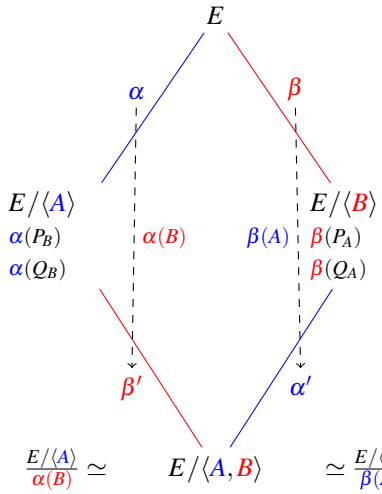


Figure III.6: Schematics of SIDH key exchange. Quantities only known to Alice are drawn in blue, quantities only known to Bob in red.

the quantities known to Alice and Bob. To set up the cryptosystem, they have publicly agreed on a prime p and a supersingular curve E such that

$$E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}/\ell_A^{e_A}\mathbb{Z})^2 \oplus (\mathbb{Z}/\ell_B^{e_B}\mathbb{Z})^2 \oplus (\mathbb{Z}/f\mathbb{Z})^2.$$

It will be convenient to also fix public bases of their respective torsion groups:

$$E[\ell_A^{e_A}] = \langle P_A, Q_A \rangle,$$
$$E[\ell_B^{e_B}] = \langle P_B, Q_B \rangle.$$

To start the protocol, they choose random secret subgroups

$$\langle A \rangle = \langle [m_A]P_A + [n_A]Q_A \rangle \subset E[\ell_A^{e_A}],$$
$$\langle B \rangle = \langle [m_B]P_B + [n_B]Q_B \rangle \subset E[\ell_B^{e_B}],$$

of respective orders $\ell_A^{e_A}, \ell_B^{e_B}$, and compute the secret isogenies

$$\alpha : E \to E/\langle A \rangle,$$
$$\beta : E \to E/\langle B \rangle.$$

They respectively publish $E_A = E/\langle A \rangle$ and $E_B = E/\langle B \rangle$.

Now, to compute the shared secret $E/\langle A,B \rangle$, Alice needs to compute the isogeny $\alpha' : E/\langle B \rangle \to E/\langle A,B \rangle$, whose kernel is generated by $\beta(A)$. We see that the kernel of α' depends on both secrets, thus Alice cannot compute it without Bob's assistance. The trick here is for Bob to publish the values $\beta(P_A)$ and $\beta(Q_A)$: they do not require the knowledge of Alice's secret, and we will assume that they do not give any advantage in computing $E/\langle A,B \rangle$ to an attacker. From Bob's published values, Alice can compute $\beta(A)$ as $[m_A]\beta(P_A) + [n_A]\beta(Q_A)$, and complete the protocol. Bob performs the analogous computation, with the help of Alice. The protocol is schematized in Figure III.6.

III.6 Security and quantum computers

We end this chapter with a quick review of the security of the key exchange protocols presented so far. The problem that is often cited as the cornerstone of isogeny based cryptography is the *isogeny walk problem*.

Problem III.11 (Isogeny walk problem)**.** Let $\mathbb{F}_q$ be a finite field. Given two elliptic curves E, E' defined over $\mathbb{F}_q$ such that $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$, find an isogeny $E \to E'$ of smooth degree.

The "smooth degree" requirement is there so that the isogeny can be represented compactly as a composition of small degree isogenies. We are purposefully vague on the distribution where E, E' are taken from, because this is going to depend on the cryptosystem. Naturally, the first parameter to look at is the size of the isogeny class of E, E': too small, and we can find the isogeny by brute force.

Security of CM constructions. In the CM case, using the *bathymeter* developed in Chapter I, we can find ascending paths from E and E' to two curves $\hat{E}, \hat{E}'$ with complex multiplication by the maximal order; then, we are left with the problem of finding a horizontal isogeny between $\hat{E}$ and $\hat{E}'$. Since the horizontal isogeny class of $\mathcal{O}_K$ is the smallest among all horizontal isogeny classes of curves with complex multiplication by some $\mathcal{O} \subset \mathcal{O}_K$, it makes sense to reduce to this case, as first noted by Galbraith, Hess and Smart [GHS02; GS13].

Problem III.12 (Horizontal isogeny walk problem)**.** Let $\mathbb{F}_q$ be a finite field, and let $\mathcal{O}_K$ be the ring of integers of a quadratic imaginary field $K = \mathbb{Q}(\sqrt{-D})$. Given two elliptic curves E, E' defined over $\mathbb{F}_q$ with complex multiplication by $\mathcal{O}_K$, find an isogeny $E \to E'$ of smooth degree.

The size of the horizontal isogeny class is $h(\mathcal{O}_K)$; it is known by the class number formula that this is in $O(\sqrt{\Delta_K}\log\Delta_K)$, and, for the typical isogeny class[7], $\Delta_K = O(q)$. The best generic attack against the Horizontal isogeny walk problem is a Pollard-rho style algorithm, performing random walks from E and E' until a collision is found [GHS02]. Its average complexity is $O(\sqrt{h(\mathcal{O}_K)})$, thus $O(q^{1/4})$ for a typical isogeny class. This justifies choosing a prime q of $4n$ bits, for a security level of 2^n, and this is indeed what we do in [DKS18] and what CSIDH does [Cas+18].

However, we must also ensure that the key space covers the whole $\mathrm{Ell}_q(\mathcal{O}_K)$, possibly approaching the uniform distribution. This means that isogeny walks, as in Eq. (III.1), must be sampled from a relatively large subset $S \subset \mathrm{Cl}(\mathcal{O}_K)$, implying that $\#S \gg \log q$. For efficiency reasons, practical instantiations take S just large enough: $\#S \sim (\log q)/2$;[8] however it will not go unnoticed that this choice is insufficient to apply Theorem III.9. We may as well live with it, changing our security assumptions to take into account the biased distributions given by random walks in graphs that are not known to be expanders, as it is done in [DKS18].

Security of SIDH. Things are quite different in SIDH. We know that the supersingular isogeny graph over $\mathbb{F}_{p^2}$ has $\approx p/12$ vertices, thus in general we can find a smooth isogeny between two supersingular curves in $O(\sqrt{p})$ operations using the same kind of random walk algorithm.

[7] Including the isogeny class of trace zero supersingular curves used in CSIDH.

[8] Additional constraints in CSIDH force $\#S$ to grow as $(\log q)/(\log\log q)$.

However, this is not the best attack against SIDH. To understand why, we need to look at the key space. Recall that the prime in SIDH is chosen of the form $p \pm 1 = \ell_A^{e_A} \ell_B^{e_B} f$. Alice's secrets are uniformly random cyclic subgroups of $E[\ell_A^{e_A}]$; Alice's key space contains thus at most $(\ell_A + 1)\ell^{e_A - 1}$ elements. Similarly, Bob's keyspace contains at most $(\ell_B + 1)\ell^{e_B - 1}$ elements. To balance out the size of the two key spaces, we need $\ell_A^{e_A} \approx \ell_B^{e_B} \approx \sqrt{p}$. Thus, Alice's and Bob's key spaces only cover a tiny fraction of the whole supersingular graph, much less they satisfy the conditions to Theorem III.10. An isogeny path in any of the two subgraphs can be found by a meet-in-the middle strategy (also called a *claw finding* algorithm) in only $O(p^{1/4})$ steps.[9]

Hence, like in the CM setting, we need to take $\log p \sim 4n$ for a security of 2^n operations. However SIDH j-invariants are elements of $\mathbb{F}_{p^2}$, thus they will typically be twice as big as j-invariants in CSIDH. Adding to that the fact that SIDH public keys contain more than a j-invariant (see Figure III.6), we see that CSIDH consumes considerably less bandwidth than SIDH. This is offset by the fact that SIDH is an order of magnitude faster than CSIDH, owing to the smaller isogeny degrees.

However, we just highlighted a very important point on SIDH: it transmits more information than what we would normally feel comfortable sharing. Indeed, SIDH transmits not only the image curve $E/\langle A \rangle$, but also the image of the basis points P_B, Q_B. This is enough information to interpolate the secret isogeny by a Couveignes-like algorithm, however we do not know how to exploit the fact that the isogeny has smooth degree, and in fact we do not know any algorithm that takes advantage of this auxiliary information.[10]

At any rate, the security of SIDH cannot be founded on the Isogeny walk problem. Instead, it is necessary to make *ad hoc* assumptions taking into account the information communicated by the protocol. These assumptions, going under the names of CSSI, SSCDH, SSDDH [JD11; DJP14] or SIDH [SIKE], are too *ad hoc* to deserve a space here; we refer the reader to the appendix for their definitions.

Quantum security. The discussion on security would not be complete without surveying quantum attacks. Indeed, the main selling point of isogeny-based key exchange protocols is their (conjectured) resistance to quantum algorithms.

Let's start with CM constructions. Couveignes' Hard Homogeneous Spaces setting is scarily similar to the Diffie–Hellman key exchange, which is indeed a special case of it. Shor's algorithm [Sho94a] solves the discrete logarithm problem in polynomial time on a quantum computer, and thus breaks the Diffie–Hellman protocol. But is there a variant of Shor's algorithm that also breaks generic HHS constructions?

Definition III.13 (Hidden Subgroup Problem (HSP))**.** Let $f : G \to X$ be a function from a group G to a set X. Assume that there is a subgroup $H \subset G$ such that $f(g) = f(g')$ if and only if $g' \in gH$. The function f is said to *hide* the subgroup H, and the *hidden subgroup problem* consists in finding generators for H, given access to f.

It is well known that Kitaev's generalization of Shor's algorithm [Kit95] solves the hidden subgroup problem in quantum polynomial time, when G is a finitely generated abelian group.

[9] A Pollard-rho style of algorithm is not possible in this case, since its complexity would depend on the size of the whole graph. The claw finding algorithm is very memory hungry, and some argue that the RAM model is not appropriate to study its complexity. In a constant memory model, the currently best attack against SIDH is estimated to take $O(p^{3/8})$ steps [Adj+18].

[10] See [Pet17] for a distant cousin of SIDH for which it is possible to extract useful information out of the auxiliary data.

Definition III.14 (Hidden Shift Problem (HShP))**.** Let $f_0, f_1 : G \to X$ be two injective functions from a group G to a set X. Assume that there is an element $s \in G$ such that $f_0(g) = f_1(gs)$ for any $g \in G$. The element s is called a *hidden shift* for f_0, f_1, and the *hidden shift problem* is to find s, given access to f_0 and f_1.

For any group G, the hidden shift problem reduces to the hidden subgroup problem for the (generalized) dihedral group $G \rtimes C_2$.[11] No generalization of Kitaev's algorithm is known for non-abelian groups, but a different family of algorithms, due to Kuperberg [Kup05; Kup13] and Regev [Reg04], solves the HShP in subexponential quantum time $\exp(\sqrt{\log \#G})$.

As first noted in [CJS14] and then improved in [BS18; BIJ18; Jao+18], Kuperberg's algorithm can be used to solve the Horizontal isogeny walk problem as follows: let E, E' be the two curves with complex multiplication by $\mathcal{O}_K$, define two functions $f_0, f_1 : \mathrm{Cl}(\mathcal{O}_K) \to \mathrm{Ell}_q(\mathcal{O}_K)$ as $f_0(\mathfrak{a}) = \mathfrak{a} \cdot E$ and $f_1(\mathfrak{a}) = \mathfrak{a} \cdot E'$, then the hidden shift defines a horizontal isogeny between E and E'.

Kuperberg's algorithm is a game changer for protocols based on complex multiplication: indeed, to ensure 2^n quantum security we need to take $\log q = O(n^2)$. The actual constant depends on the variant of Kuperberg's algorithm, and various parameters such as available quantum memory; its exact value is currently debated, but it appears that taking $\log q$ somewhere between 512 and 1024 bits grants a security of 2^{64} quantum gates [Cas+18; DKS18; BS18; Ber+18].

For SIDH, on the other hand, there is no group structure[12] that can be exploited by Kuperberg's algorithm. Currently, the best quantum attack against SIDH is a Grover-like claw finding algorithm due to Tani [Tan09], requiring $O(p^{1/6})$ quantum gates (and as many qubits!). For this reason, p is typically chosen so that $\log p \sim 6n$ for a quantum security of 2^n gates, although it is debated whether Tani's algorithm actually presents an advantage over the classical claw finding attack.

III.7 Perspectives

The field of isogeny-based cryptography is a relatively young one, and still confidential compared to other post-quantum families.

Other protocols. In this chapter we have only presented key exchange systems, however it is possible to obtain other interesting protocols from isogeny graphs. Public key encryption *à la* El Gamal is an easy exercise, whereas CCA-secure Key Encapsulation Methods (KEMs) already pose a riddle. On one hand, key validation is problematic for SIDH, forcing the use of generic transforms such as Fujisaki and Okamoto's [FO99]. On the other hand, key validation in CM systems essentially amounts to verifying the order of the elliptic curves; this allows CCA-secure systems à la DHIES [ABR99; ABR01; CS03], but also static-static non-interactive key exchange (NIKE), making CSIDH the first practical post-quantum NIKE.

Signatures are another soft spot of isogeny-based cryptography. No analogue of Schnorr signatures is known for any of the primitives presented so far. Instead, we have zero-knowledge identification schemes for SIDH [DJP14; Yoo+17; GPS17] and,

[11] To reduce HShP to HSP, simply define the function f by $f(g,1) = f_0(g)$ and $f(g,-1) = f_1(g)$, so that the hidden subgroup is generated by $(s,-1)$.

[12] Outside of the subgraph of $\mathbb{F}_p$-rational curves, but the algorithm in [BJS14] does not impact the security of SIDH.

very recently, CSIDH [DG18]. From these, we can derive signature schemes via the Fiat–Shamir transform, but these are either slow, or have large signatures, or both. It is currently an open problem to build a compact and efficient signature scheme from isogeny primitives.

Finally, none of the advanced protocols derived from Diffie–Helman, such as identity based encryption, blind signatures, etc., is known to have an isogeny-based analogue. One of the most advanced cryptographic ideas based on isogenies, a sort of multilinear map, has been recently introduced by Boneh, Glass, Krashen, Lauter, Sharif, Silverberg, Tibouchi and Zhandry [Bon+18]; however the idea fails to give an actual protocol, because a key mathematical ingredient (a generalization of the j-invariant of elliptic curves) is missing, and finding it was left as an open problem by the authors.

Other graphs. It is also natural to ask whether other families of expander graphs could be used for cryptography. LPS graphs [LPS88], for example, are very much related to supersingular graphs, and have already been proposed as a basis for (symmetric) cryptography [CGL09], although they have been broken [TZ08; PLQ08].

As another example, groupoids of maximal orders of quaternion algebras are isomorphic to supersingular isogeny graphs; however it has been shown that the equivalent of the Isogeny walk problem in these groupoids can be solved in (classical) polynomial time [Koh+14].[13] We do not know how to use this result to break SIDH, however it has been employed in various security reductions [Gal+16; Eis+18].

An obvious generalization of isogeny based protocols would consist in replacing elliptic curves with higher dimensional varieties. Some advantages are to be expected: higher dimensional varieties have larger torsion groups, thus more isogenies for a fixed degree. This has the potential to produce smaller parameters, however the current knowledge on isogenies in higher dimension is still very rudimentary, and few algorithms exist. Some early progress in understanding them has been made in [LR12; LR15; CR15; IT14; BJW17], it would be interesting to further develop this field.

Efficiency. Both SIDH and CSIDH have very small keys, compared to other post-quantum candidates (or even to RSA). However, SIDH is among the slowest candidates to the NIST post-quantum competition, and CSIDH is currently an order of magnitude slower.

Considerable efforts have been devoted to speed up SIDH, using ideas such as *optimal strategies* for Vélu's formulas [DJP14], *projectivized* curve equations [CLN16], *key compression* [Aza+16; Cos+17; Zan+18], arithmetic modulo primes of special form [CLN16; Kar+16; BF17], dedicated *Montgomery ladders* [Faz+17], and various software and hardware level optimizations [MRC17]. Owing to this, we seem to have pretty much hit a wall on how much SIDH can be further sped up: barring spectacular discoveries[14], its efficiency is bound to receive only minor improvements in the coming years. In particular, this poses a problem for IoT and other embedded devices, where SIDH is still unsuitable owing to its large memory and CPU requirements.

[13] To put this result into perspective, note that its equivalent for the CM case would amount to solve the discrete logarithm problem of $\mathrm{Cl}(\mathcal{O})$ in classical polynomial time!

[14] One particular trick in CSIDH that is completely absent in SIDH is using the quadratic twist to perform part of the computations. I have thought of this for a while, and I see no fundamental reason why it should not work for SIDH, if it was not for the fact that finding suitable parameters seems computationally unfeasible. My favorite example is $p = 17$, so $p^2 - 1 = 2^5 3^2$; if it were possible to find large primes with similar properties, the gain would be spectacular.

On the other hand, CSIDH is still very young, and a lot of optimization avenues are yet to be explored. In particular, CSIDH still lacks a constant-time implementation. The abysmal performance of the ordinary curve Couveignes–Rostovtsev–Stolbunov protocol could also use some improvements, although I am out of ideas at the moment.

In the CSIDH setting, different contexts call for different parameter choices. Note that, while p must grow like $O(n^2)$ due to Kuperberg's algorithm, the key space is only forced to grow at a much slower rate of $O(n)$, the best available attacks being the same claw finding algorithms used against SIDH. While the authors of [Cas+18] have the key-space grow at the same rate as the prime p, it is more convenient for signatures to have a smaller key space [DG18]. This of course leads to a probability distribution on $\mathrm{Ell}_q(\mathcal{O})$ very far from uniform, potentially affecting security proofs. It is an interesting problem to study the various compromises that can be made on parameters, and their impact on security.

Security. Obviously, confidence in isogeny based cryptography can only be gained through more research on security.

This means, of course, working on attacks. A fundamental topic, at the moment, is establishing the quantum security of CSIDH. Some preliminary work has been done in [BS18; BIJ18; Jao+18], but a consensus has yet to be reached.

Concerning SIDH, it is interesting that the only recent result on generic attacks shows that SIDH is potentially *more* secure than originally thought [Adj+18]. It is known that an efficient algorithm to compute endomorphism rings of supersingular curves would also break SIDH [Gal+16], however the currently best algorithms for this problem [Cer04; Koh96] have much worse complexity than other attacks on SIDH. It would be extremely interesting to look for sub-exponential algorithms for computing endomorphism rings of supersingular curves, or alternatively produce convincing arguments for why this is not possible.

However, none of the known attacks on SIDH exploits the "auxiliary" information transmitted by the protocol in the form of torsion points. Some work in this direction has been done by Petit [Pet17], but no direct impact on SIDH has been demonstrated. In particular, there seems to be absolutely no research on using the auxiliary information in a quantum algorithm.

Whether quantum computers, Tate's theorem, Couveignes' algorithm, or a combination of all can help crack the security of SIDH is certainly one of the most fascinating topics in isogeny based cryptography.

Besides key recovery, other kinds of attacks are also interesting. The most celebrated one is Galbraith, Petit, Shani and Ti's key learning attack against the static version of SIDH [Gal+16], showing that key validation for SIDH is indeed hard. Other attack models, e.g., side channel attacks, have also been investigated [GW17; Ti17]. These efforts remain sporadic, and more work is needed in these directions.

Finally, more security proofs would help consolidate trust in isogenies. The security assumption of SIDH being so *ad hoc*, more work on security reductions, such as [Gal+16; Eis+18], could help build trust by reducing security to more "natural" problems. CSIDH, on the other hand, benefits from a much nicer hardness assumption, although key distribution is often an issue for security proofs: it is not clear how much its security is affected by more aggressive parameters deviating considerably from the uniform distribution. Given the intended use of these protocols, proofs of quantum security are another obvious target for research.

Appendix: five peer-reviewed research articles

A Explicit isogenies in quadratic time in any characteristic

Abstract

Consider two ordinary elliptic curves E, E' defined over a finite field $\mathbb{F}_q$, and suppose that there exists an isogeny ψ between E and E'. We propose an algorithm that determines ψ from the knowledge of E, E' and of its degree r, by using the structure of the ℓ-torsion of the curves (where ℓ is a prime different from the characteristic p of the base field).

Our approach is inspired by a previous algorithm due to Couveignes, that involved computations using the p-torsion on the curves. The most refined version of that algorithm, due to De Feo, has a complexity of $\tilde{O}(r^2)p^{O(1)}$ base field operations. On the other hand, the cost of our algorithm is $\tilde{O}(r^2)\log(q)^{O(1)}$, for a large class of inputs; this makes it an interesting alternative for the medium- and large-characteristic cases.

A.1 Introduction

Isogenies are non-zero morphisms of elliptic curves, that is, non-constant rational maps preserving the identity element. They are also algebraic group morphisms. Isogeny computations play a central role in the algorithmic theory of elliptic curves. They are notably used to speed up Schoof's point counting algorithm [Sch85; Atk91; Sch95; Elk98]. They are also widely applied in cryptography, where they are used to speed up point multiplication [GLV01; LS14], to perform cryptanalysis [MMT01], and to construct new cryptosystems [Tes06; CGL09; Sto10; DJP14; JS14].

The *degree* of an isogeny is its degree as a rational map. If an isogeny has degree r, we call it an r-isogeny, and we say that two elliptic curves are r-isogenous if there exists an r-isogeny relating them. Accordingly, we say that two field elements j and j' are r-isogenous if there exist r-isogenous elliptic curves E and E' such that $j(E) = j$ and $j(E') = j'$. The *explicit isogeny* problem has many incarnations. In this paper, we are interested in the variant defined below.

Problem A.1 (Explicit isogeny problem)**.** Given two j-invariants j and j', and a positive integer r, determine if they are r-isogenous. In that case, compute curves E, E' with $j(E) = j$ and $j(E') = j'$, and the rational functions defining an r-isogeny $\psi : E \to E'$.

A good measure of the computational difficulty of the problem is given by the isogeny degree r. Indeed the output is represented by $O(r)$ base field elements, hence an asymptotically optimal algorithm would solve the problem using $O(r)$ field operations. Even though the input size is logarithmic in r, by a slight abuse we say that an algorithm solves the isogeny problem in polynomial time if it does so in the size of the output. Thanks to Vélu's formulas [Vél71], in particular the version appearing in [Koh96, §2.4], we can compute ψ from the knowledge of the polynomial h vanishing on the abscissas of the points in $\ker \psi$, at the cost of a constant number of multiplications of polynomials of degree $O(r)$. Given that all known algorithms to compute h require more than a few polynomial multiplications, we often say that we have computed ψ whenever we have computed h, and conversely.

This paper focuses on the explicit isogeny problem for *ordinary* elliptic curves over finite fields. A famous theorem by Tate [Tat66] states that two curves are isogenous over a finite field if and only if they have the same cardinality over that field. The explicit isogeny problem stated here appears naturally in the Schoof-Elkies-Atkin point

counting algorithm (SEA). There, E is a curve over $\mathbb{F}_q$, whose rational points we wish to count, and E' is an r-isogenous curve, with r a prime of size approximately $\log(q)$. For this reason, the explicit isogeny problem is customarily solved without prior knowledge of the cardinality of $E(\mathbb{F}_q)$. We abide by this convention here.

Many algorithms have been suggested over the years to solve the explicit isogeny problem. Early algorithms were due to Atkin [Atk91] and Charlap, Coley and Robbins [CCR91]. Elkies' [Elk98; Bos+08] was the first algorithm targeted to finite fields (of large enough characteristic). Assuming r is prime, its complexity is dominated by the computation of the modular polynomial Φ_r, which is an object of bit size $O(r^3 \log(r))$. Later Bröker, Lauter and Sutherland [BLS12] optimized the modular polynomial computation in the context of the SEA algorithm [Sut13b]. Finally Lercier and Sirvent [LS08b; LV16] generalized Elkies' algorithm to work in any characteristic. Despite these advances, the overall cost of Elkies' algorithm and its variants is still at least cubic in r.

Another line of work to solve the explicit isogeny problem for ordinary curves was initiated by Couveignes [Cou94; Cou96; Cou00], and later improved by De Feo and Schost [De 11; DS12]. These algorithms use an interpolation approach combined with ad-hoc constructions for towers of finite fields of characteristic p. Their complexity is quasi-quadratic in r, but exponential in $\log(p)$, hence they are only practical for very small characteristic.

In this paper we present a variant of Couveignes' algorithm with complexity polynomial in $\log(p)$ and quasi-quadratic in r. Like the original algorithm, it is limited to isogenies of ordinary curves. Together with the Lercier-Sirvent algorithm, they are the only polynomial-time isogeny computation algorithms working in any characteristic, hence they are especially relevant for counting points in *medium* characteristic (i.e., counting points over $\mathbb{F}_{p^n}$, when $n \gg p/\log(p)$).

Note that, although Couveignes-type algorithms do not make use of the modular polynomial Φ_r, its computation is still necessary in the context of the SEA algorithm. Thus our new algorithm does not improve the overall complexity of point counting, though it may provide a speed-up in some cases. It gives, however, an effective algorithm for solving the explicit isogeny problem, with potential applications in other contexts, e.g., cryptography.

Notation

Throughout this paper: r is a positive integer, p an odd prime, q a power of p, and $\mathbb{F}_q$ is the finite field with q elements. E is an ordinary elliptic curve over $\mathbb{F}_q$, its group of n-torsion points is denoted by $E[n]$, its q-Frobenius endomorphism by π. The endomorphism ring of E is denoted by $\mathcal{O}$, with $K = \mathcal{O} \otimes \mathbb{Q}$ the corresponding number field, $\mathcal{O}_K$ its maximal order, and d_K the discriminant of $\mathcal{O}_K$. For a prime ℓ different from p and not dividing r, we denote by $E[\ell^k]$ the group of ℓ^k-torsion points of E, $E[\ell^\infty] = \varinjlim E[\ell^k]$ the union of all $E[\ell^k]$, and $T_\ell(E) = \varprojlim E[\ell^k]$ the ℓ-adic Tate module [Sil92, p. III.7], which is free of rank two over $\mathbb{Z}_\ell$. The factorization of the characteristic polynomial of π over $\mathbb{Z}_\ell$ is determined by the Kronecker symbol (d_K/ℓ). If $(d_K/\ell) = +1$ then we also define λ, μ as the eigenvalues of π in $\mathbb{Z}_\ell$ and write $h = v_\ell(\lambda - \mu)$, where v_ℓ is the ℓ-adic valuation.

We measure all computational complexities in terms of operations in $\mathbb{F}_q$; the boolean costs associated to the algorithms presented next are negligible compared to the algebraic costs, and will be ignored. We use the Landau notation $O(\,)$ to express asymptotic complexities, and the notation $\tilde{O}(\,)$ to neglect (poly)logarithmic factors. We let $\mathsf{M}(n)$

be a function such that polynomials in $\mathbb{F}_q[x]$ of degree less than n can be multiplied using $\mathsf{M}(n)$ operations in $\mathbb{F}_q$, under the assumptions of [GG99, Chapter 8.3]. Using FFT multiplication, one can take $\mathsf{M}(n) \in O(n\log(n)\operatorname{loglog}(n))$.

Couveignes' algorithm and our contribution

Couveignes' isogeny algorithm takes as input two *ordinary* j-invariants $j, j' \in \mathbb{F}_q$, and a positive integer r not divisible by p, and returns, if it exists, an r-isogeny $\psi : E \to E'$, with $j(E) = j$ and $j(E') = j'$. It is based on the observation that the isogeny ψ must put $E[p^k]$ in bijection with $E'[p^k]$, in a way that is compatible with their structure as cyclic groups. It proceeds in three steps

1. Compute generators P, P' of $E[p^k]$ and $E'[p^k]$ respectively, for k large enough;

2. Compute the interpolation polynomial L sending $x(P)$ to $x(P')$, and the abscissas of their scalar multiples accordingly;

3. Deduce a rational fraction $g(x)/h(x)$ that coincides with L at all points of $E[p^k]$, and verify that it defines the x-component of an isogeny of degree r. If it does, return it; otherwise, replace P' with a scalar multiple of itself and go back to Step 2.

For this algorithm to succeed, enough interpolation points are required. Given that the x-component of ψ is defined by $O(r)$ coefficients, we have $p^k \in \Theta(r)$. However, most of the time, those points are not going to be defined in the base field $\mathbb{F}_q$, so we must use efficient algorithms to construct and compute in towers of extensions of finite fields. Indeed, Couveignes and his successors go at great length in studying the arithmetic of *Artin-Schreier towers* [Cou00; DS12], and the adaptation of the fast interpolation algorithm to that setting [De 11]. Using these highly specialized constructions, Steps 1 and 2 are both executed in time $\tilde{O}(p^{k+O(1)}) = \tilde{O}(rp^{O(1)})$. However the last step only succeeds for one pair of torsion points P, P', in general, thus $O(r)$ trials are expected on average. Hence, the overall complexity of Couveignes' algorithm is $\tilde{O}(r^2 p^{O(1)})$, i.e., quadratic in r, but exponential in $\log(p)$. Although the exponent of p is relatively small, Couveignes algorithm quickly becomes impractical as p grows.

In this paper we introduce a variant of Couveignes' algorithm with the same quadratic complexity in r, and **no exponential dependency in** $\log(p)$.

The bottom line of our algorithm is elementary: replace $E[p^k]$ in the algorithm with $E[\ell^k]$, for some small prime ℓ. However a naive application of this idea fails to yield a quadratic-time algorithm. Indeed, in the worst case one has $\ell^{2k} \in \Theta(r)$, with $E[\ell^k] \simeq (\mathbb{Z}/\ell^k\mathbb{Z})^2$. Hence, two generators P, Q of $E[\ell^k]$ must be mapped onto two generators of $E'[\ell^k]$. This can be done in $O(\ell^{4k})$ possible ways, with a best possible cost of $O(\ell^{2k})$ per trial, thus yielding an algorithm of complexity $O(\ell^{6k}) = O(r^3)$ at best.

To avoid this pitfall, we carefully study in Section A.2 the structure of $E[\ell^k]$, and its relationship with the Frobenius endomorphism π. With that knowledge, we can put some restrictions on the generators P, Q, as explained in Section A.3, thus limiting the number of trials to $O(\ell^{2k})$. In Section A.4 we present an interpolation algorithm adapted to the setting of ℓ-adic towers, and in Section A.5 we put all steps together and analyze the full algorithm. Finally in Section A.6 we discuss our implementation and the performance of the algorithm.

Towers of finite fields

The algorithms presented next operate on elements defined in finite extensions of $\mathbb{F}_q$. Specifically, we will work in a *tower* of finite fields $\mathbb{F}_q = F_0 \subset F_1 \subset \cdots \subset F_n$, with ℓ dividing $\#F_1 - 1$, $d_1 = [F_1 : F_0]$ dividing $\ell - 1$, and $[F_{i+1} : F_i] = \ell$ for any $i > 0$. For $\ell = 2$, we build upon the work of Doliskani and Schost [DS15], whereas for general ℓ we use towers of Kummer extensions in a way similar to [DDS13, §2]. Both constructions represent elements of F_i as univariate polynomials with coefficients in $\mathbb{F}_q$, thus basic arithmetic operations can be performed using modular polynomial arithmetic over $\mathbb{F}_q$. While constructing the tower, we also enforce special relations between the generators of each level, so that moving elements up and down the tower, and testing membership, can be done at negligible cost.

We briefly sketch the construction for odd ℓ. We first look for a primitive polynomial $P_1 \in \mathbb{F}_q[x]$ of degree equal to $[F_1 : F_0]$. There are many probabilistic algorithms to compute P_1 in expected time polynomial in ℓ and $\log(q)$; since their cost does not depend on the height n of the tower, we neglect it (in all that follows, by *expected cost* of an algorithm, we refer to a Las Vegas algorithm, whose runtime is given in expectation). Then, the image x_1 of x in $F_1 = \mathbb{F}_q[x]/P_1(x)$ is an element of multiplicative order $\#F_1 - 1$, and in particular it is not a ℓ-th power. Hence for any $i > 1$ we define F_i as $\mathbb{F}_q[x]/P_1\left(x^{\ell^{i-1}}\right)$, the computation of the polynomials $P_1\left(x^{\ell^{i-1}}\right)$ incurring no algebraic cost. Using this representation, elements of F_i can be expressed as elements of a higher level F_{i+j}, and reciprocally, by a simple rearrangement of the coefficients. Another fundamental operation can be done much more efficiently than in generic finite fields, as the following generalization of [DS15, §2.3] shows.

Lemma A.2. *Let $F_0 \subset \cdots \subset F_n$ be a Kummer tower as defined above, and let $a \in F_i$ for some $0 \leq i \leq n$. For any integer j, we can compute the $(\#F_j)$-th power of a using $O(\ell^{i-1}\mathsf{M}(\ell))$ operations in $\mathbb{F}_q$, after a precomputation independent of a of cost $O(\ell\mathsf{M}(\ell)\log(q))$.*

Proof. Without loss of generality, we can assume that $j < i$; otherwise, the output is simply a itself. Let $s = \#F_j$, and let $d = [F_i : F_1] = \ell^{i-1}$. Let x_i be the image of x in $F_i = \mathbb{F}_q[x]/P_i(x)$, so that $x_i^d = x_1$.

The first step, independently of a, is to compute $y = x_i^s$. Writing $s = ud + r$, with $r < d$, we see that y is given by $x_1^{u \bmod \#F_1} x_i^r$. We compute $x_1^{u \bmod \#F_1}$ using $O(\ell\mathsf{M}(\ell)\log(q))$ operations in $\mathbb{F}_q$, and we keep this element as a monomial of $F_1[x_i]$. By assumption, a is represented as a polynomial in x_i of degree less than $[F_i : F_0]$. We rewrite it as $a = a_0 + a_1 x_i + \cdots + a_{d-1} x_i^{d-1}$, with $a_i \in F_1$. This is done by a simple rearrangement of the coefficients of a.

Finally, we compute $a(y)$ by a Horner scheme. All powers y^k we need are themselves monomials in $F_1[x_i]$, each computed from the previous one using $O(\mathsf{M}(\ell))$ operations in $\mathbb{F}_q$, for a total of $O(\ell^{i-1}\mathsf{M}(\ell))$. Finally the monomials $a_k y^k$ are combined together to form a polynomial in (x_1, x_i) of degree less than (d_1, d), and then brought to a canonical form in F_i via another rearrangement of coefficients. □

Summarizing, the following computations can be performed in a Kummer tower at the indicated asymptotic costs, all expressed in terms of operations in $\mathbb{F}_q$.

- basic arithmetic operations (addition, multiplication) in F_i, using $O(\mathsf{M}(\ell^i))$ operations;

- inversion in F_i using $O(\mathsf{M}(\ell^i)\log(\ell^i))$ operations (when $\ell = 2$, a factor of i can be saved here [DS15], but we will disregard this optimization for simplicity.)
- mapping elements from F_{i-1} to F_i and *vice versa* at no arithmetic cost;
- multiplication and Euclidean division of polynomials of degree at most d in $F_i[x]$ using $O(\mathsf{M}(d\ell^i))$ operations, via Kronecker's substitution, as already done in e.g. [GS92];
- computing a $(\#F_j)$-th power in F_i using $O(\ell^{i-1}\mathsf{M}(\ell))$ operations, after a precomputation that uses $O(\ell\mathsf{M}(\ell)\log(q))$ operations.

For one fundamental operation, we only have an efficient algorithm in the case $\ell = 2$, hence we introduce the following notation:

- $\mathsf{R}(i)$ is a bound on the expected cost of finding a root of a polynomial of degree ℓ in $F_i[x]$.

Note that we allow Las Vegas algorithms here, as no deterministic polynomial time algorithm is known. For $\ell = 2$, Doliskani and Schost show that $\mathsf{R}(i) = O(\mathsf{M}(\ell^i)\log(\ell^i q))$. For general ℓ, we have $\mathsf{R}(i) = O(\ell^i \mathsf{M}(\ell^{i+1})\log(\ell)\log(\ell q))$ using the variant of the Cantor-Zassenhaus algorithm described in [GG99, Chapter 14.5], or $\mathsf{R}(i) = O\big((\ell^{i(\omega+1)/2} + \mathsf{M}(\ell^{i+1}\log(q)))i\log(\ell)\big)$ using [KS97]. Here, ω is such that matrix multiplication in size m over any ring can be done in $O(m^\omega)$ base ring operations (so we can take $\omega = 2.38$ using the Coppersmith-Winograd algorithm). In any case, $\mathsf{R}(i)$ is between linear and quadratic in the degree ℓ^i.

A.2 The Frobenius and the volcano

In this section we explore some fundamental properties of ordinary elliptic curves over finite fields: the structure of their isogeny classes, its relationship with the rational ℓ^∞-torsion points, and with the Frobenius endomorphism π.

Isogeny volcanoes

For an extensive introduction to isogeny volcanoes we refer the reader to [Sut13a]. We recall here, without their proof, two results about ℓ-isogenies between ordinary elliptic curves.

Proposition A.3 ([Koh96, Proposition 21])**.** *Let $\phi : E \to E'$ be an ℓ-isogeny between ordinary elliptic curves and $\mathcal{O}, \mathcal{O}'$ be their endomorphism rings. Then one of the three following cases is true:*

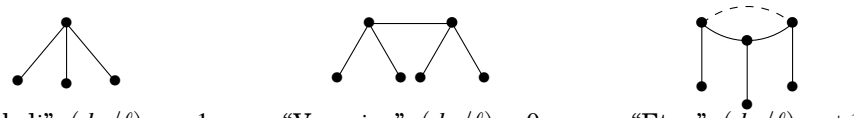
1. *$[\mathcal{O}' : \mathcal{O}] = \ell$, in which case we call ϕ* ascending*;*
2. *$[\mathcal{O} : \mathcal{O}'] = \ell$, in which case we call ϕ* descending*;*
3. *$\mathcal{O}' = \mathcal{O}$, in which case we call ϕ* horizontal*.*

Proposition A.4 ([Koh96, Proposition 23]; [Sut13a, Lemma 6])**.** *Let E be an ordinary elliptic curve with endomorphism ring $\mathcal{O}$.*

1. *If $\mathcal{O}$ is ℓ-maximal then there are $(d_K/\ell)+1$ horizontal ℓ-isogenies from E (and no ascending ℓ-isogenies).*

2. *If $\mathcal{O}$ is not ℓ-maximal then there are no horizontal ℓ-isogenies from E, and one ascending ℓ-isogeny.*

A *volcano of ℓ-isogenies* is a connected component of the graph of rational ℓ-isogenies between curves defined on $\mathbb{F}_q$. The *crater* is the subgraph corresponding to curves having an ℓ-maximal endomorphism ring. The shape of the crater is given by the Kronecker symbol (d_K/ℓ), as per Proposition A.4. For any $k \geq 0$, an ℓ^k-isogeny is *horizontal* if it is the composite of k horizontal ℓ-isogenies. The *depth* of a curve is its distance from the crater. It is also the ℓ-adic valuation of the conductor of $\mathcal{O} = \mathrm{End}(E)$. We illustrate the three possible shapes for a volcano in Figure A.1.



"Stromboli": $(d_K/\ell) = -1$ "Vesuvius": $(d_K/\ell) = 0$ "Etna": $(d_K/\ell) = +1$

Figure A.1: The three shapes of volcanoes of 2-isogenies

The ℓ-adic Frobenius

In the rest of this paper we consider only a volcano with a cyclic crater (i.e. we assume $(d_K/\ell) = +1$), so that ℓ is an Elkies prime for these curves. This implies that the Frobenius automorphism on $T_\ell(E)$, which we write $\pi|T_\ell(E)$, has two distinct eigenvalues $\lambda \neq \mu$. The depth of the volcano of $\mathbb{F}_q$-rational ℓ-isogenies is $h = v_\ell(\lambda - \mu)$ [Sut13a, Theorem 7(iv)].

Proposition A.5. *Let E be an ordinary elliptic curve with Frobenius endomorphism π. Assume that the characteristic polynomial of π has two distinct roots λ, μ in $\mathbb{Z}_\ell$, so that the ℓ-isogeny volcano has a cyclic crater. Then there exists a unique $e \in [\![0, h]\!]$ such that $\pi|T_\ell(E)$ is conjugate, over $\mathbb{Z}_\ell$, to the matrix $\left(\begin{smallmatrix} \lambda & \ell^e \\ 0 & \mu \end{smallmatrix}\right)$. Moreover $e = h$ if E lies on the crater, and else $h - e$ is the depth of E in the volcano.*

We note here that the matrix $\left(\begin{smallmatrix} \lambda & \ell^h \\ 0 & \mu \end{smallmatrix}\right)$ is conjugate over $\mathbb{Z}_\ell$ to $\left(\begin{smallmatrix} \lambda & 0 \\ 0 & \mu \end{smallmatrix}\right)$.

Proof. Since the characteristic polynomial of π splits over $\mathbb{Z}_\ell$, the matrix of $\pi|T_\ell(E)$ is trigonalizable. Conjugating the matrix $\left(\begin{smallmatrix} \lambda & a \\ 0 & \mu \end{smallmatrix}\right)$ by $\left(\begin{smallmatrix} 1 & b \\ 0 & 1 \end{smallmatrix}\right)$ replaces a by $a - b(\lambda - \mu)$, and conjugating by $\left(\begin{smallmatrix} c & 0 \\ 0 & 1 \end{smallmatrix}\right)$ replaces a by $c \cdot a$, so that the valuation $e = v_\ell(a)$ is an invariant under matrix conjugation. This proves the first part. For the second part, by Tate's theorem [Sil92, Isogeny theorem 7.7 (a)], $\mathcal{O} \otimes \mathbb{Z}_\ell$ is isomorphic to the order in $\mathbb{Q}_\ell[\pi_\ell]$ of matrices with integer coefficients, which is generated by the identity and $\ell^{-\min(h, v_\ell(a))}(\pi_\ell - \lambda)$. □

We now study the action of ℓ-isogenies on the ℓ-adic Frobenius by showing the link between two related notions of diagonalization.

Definition A.6 (Horizontal and diagonal bases)**.** Let E be a curve lying on the crater. We call a point of $E[\ell^k]$ *horizontal* if it generates the kernel of a horizontal ℓ^k-isogeny. We call a basis of $E[\ell^k]$ *diagonal* if π is diagonal in it, *horizontal* if both basis points are horizontal.

Proposition A.7. *Let E be a curve lying on the crater and P be a point of $E[\ell^k]$ such that $\ell^h P$ is an eigenvector of π. Then $\ell^h P$ is horizontal if, and only if, P is an eigenvector for π. If $\pi(P) = \lambda P$ then we say that $\ell^h P$ has direction λ.*

This proposition being trivially true for $h \geq k$, we assume that $k \geq h$ in what follows.

Let R be a point of E of order ℓ^k, let ϕ be the isogeny with kernel $\langle R \rangle$, and let E' be its image. The subgroup $\langle R \rangle$ defines a point in the projective space of $E[\ell^k]$, which is a projective line over $\mathbb{Z}/\ell^k\mathbb{Z}$. There exists a canonical bijection [Ser77, p. II.1.1] between this projective line and the set of lattices of index ℓ^k in the $\mathbb{Z}_\ell$-module $T_\ell(E)$: it maps a line $\langle R \rangle$ to the lattice $\Lambda_R = \langle R \rangle + \ell^k T_\ell(E)$. This lattice is also the preimage by ϕ of the lattice $\ell^k T_\ell(E')$.

Fix a basis (P, Q) of $E[\ell^k]$, let Π be the matrix of π in this basis, and let $R = xP + yQ$. The lattice Λ_R is generated by the columns of the matrix $L_R = \begin{pmatrix} \ell^k & 0 & x \\ 0 & \ell^k & y \end{pmatrix}$. The Hermite normal form of L_R is $M_R = \begin{pmatrix} \ell^{k-m} & x/y' \\ 0 & \ell^m \end{pmatrix}$, where we write $y = \ell^m y'$ with $\ell \nmid y'$, and the columns of M_R also generate the lattice Λ_R. We check that M_R has determinant ℓ^k. Since $\Lambda_R = \phi_R^{-1}(\ell^k T_\ell(E'))$, there exists a basis of $T_\ell(E')$ in which ϕ_R has matrix $\ell^k M_R^{-1}$. Therefore, in that basis of $T_\ell(E')$, the matrix of $\pi | T_\ell(E')$ is $M_R^{-1} \cdot \Pi \cdot M_R$.

Proof of Proposition A.7. Fix a basis (R, S) of $E[\ell^k]$ that diagonalizes π. We can write $P = xR + yS$; without loss of generality we may assume $y = 1$. Let ϕ be the isogeny determined by $\ell^h P$, and let E' be its image. Since $\ell^h P$ is an eigenvector of π, ϕ is a rational isogeny. According to the previous discussion, $\pi | T_\ell(E')$ has matrix $\begin{pmatrix} \lambda & \ell^{h-k} x(\lambda - \mu) \\ 0 & \mu \end{pmatrix}$. This matrix is diagonalizable only if $v_\ell(x) \geq k - h$. On the other hand, we can compute $(\pi - \mu)P = x(\lambda - \mu)R$, so that P is an eigenvector on the same condition $v_\ell(x) \geq k - h$. □

While horizontal bases are our main interest, diagonal bases are easier to compute in practice. Algorithms computing both kind of bases are given in Section A.3. The main tool for this is the next proposition: given a horizontal point of order ℓ^k, it allows us to compute a horizontal point of order ℓ^{k+1}.

Proposition A.8. *Let $\psi : E \to E'$ be a horizontal ℓ-isogeny with direction λ. For any point $Q \in E[\ell^\infty]$, if ℓQ is horizontal with direction μ, then $\psi(Q)$ is horizontal with direction μ.*

Proof. Let $Q' = \psi(Q)$ and $\widehat{\psi}$ be the isogeny dual to ψ. Since both $\widehat{\psi}$ and $\widehat{\psi}(Q') = \ell Q$ are horizontal with direction μ, Q' is also horizontal. □

Proposition A.9. *Let $\psi : E \to E'$ be an isogeny of degree r prime to ℓ.*

1. *The curves E and E' have the same depth in their ℓ-isogeny volcanoes.*
2. *For any point $P \in E[\ell^k]$, the isogenies with kernel $\langle P \rangle$ and $\langle \psi(P) \rangle$ have the same type (ascending, descending, or horizontal with the same direction).*
3. *If $P \in E[\ell]$ and $P' \in E'[\ell]$ are both ascending, or both horizontal with the same direction, then E/P and E'/P' are again r-isogenous.*

Proof. Points (i) and (ii) are consequences of Proposition A.5 and of the fact that ψ, being rational and of degree prime to ℓ, induces an isomorphism between the Tate modules of E and E', commuting to the Frobenius endomorphisms. For point (iii), we

just note that since there exists a unique subgroup of order ℓ which is either ascending or horizontal with a given direction, we must have $\langle P' \rangle = \langle \psi(P) \rangle$. □

Galois classes in the ℓ-torsion

Assume that E has a ℓ-maximal endomorphism ring. The following proposition summarizes the properties of $E[\ell^k]$ that we will need for our main interpolation algorithm. If ℓ is odd, let $\alpha = v_\ell(\lambda^{\ell-1} - 1)$ and $\beta = v_\ell(\mu^{\ell-1} - 1)$; if $\ell = 2$, let $\alpha = v_2(\lambda^2 - 1) - 1$ and $\beta = v_2(\mu^2 - 1) - 1$, and assume without loss of generality that $\alpha \geq \beta$. Since $\lambda \not\equiv \mu \pmod{\ell^{h+1}}$, it is impossible that $\lambda \equiv \mu \equiv 1 \pmod{\ell^h}$, so that one at least of the two valuations α, β is $\leq h$, and therefore $\beta \leq h$.

Proposition A.10. *For any k, let d_k be the degree of the smallest field extension $F/\mathbb{F}_q$ such that $E[\ell^k] \subset E(F)$. Then:*

1. *The order of q in $(\mathbb{Z}/\ell\mathbb{Z})^\times$ divides d_1, and d_1 divides $(\ell - 1)$.*
2. *If ℓ is odd then for all $k \geq 1$, $d_k = \ell^{\min(v_\ell(d_1), k-\beta)}$.*
3. *If $\ell = 2$ then $d_2 \in \{1, 2\}$ and, for all $k \geq 2$, $d_k = \ell^{\min(v_\ell(d_2), k-\beta)}$.*
4. *Let $[F : \mathbb{F}_q] = d_1 \ell^n$, the group $E[\ell^\infty](F)$ is isomorphic to $(\mathbb{Z}/\ell^{n+\alpha}\mathbb{Z}) \times (\mathbb{Z}/\ell^{n+\beta}\mathbb{Z})$.*
5. *The group $E[\ell^k]$ contains at most $k \cdot \ell^{k+\beta}$ Galois conjugacy classes over $F_1 = \mathbb{F}_{q^{d_1}}$.*

Proof. The degree d_k is exactly the order of the matrix $\pi | E[\ell^k]$. It is therefore the least common multiple of the multiplicative orders of λ, μ modulo ℓ^k. This proves (i) using the fact that $\lambda \cdot \mu = q$. For points (ii)–(v) we may assume that $d_1 = 1$. Then, for any N, $v_\ell(\lambda^{2N} - 1) = \alpha + v_\ell(2N)$. Let (P, Q) be a diagonal basis of $E[\ell^k]$. The point $(\pi^N - 1)(xP + yQ) = (\lambda^N - 1)xP + (\mu^N - 1)yQ$ is zero iff $v_\ell(x) + \alpha + v_\ell(N) \geq k$ and $v_\ell(y) + \beta + v_\ell(N) \geq k$. This shows (iv). The largest Galois classes are those for which $v_\ell(y) = 0$ and their size is $\ell^{k-\beta}$, proving (ii) and (iii). Moreover, for any $i \leq k - \beta$ the points in an orbit of size $\leq \ell^i$ are those for which $v_\ell(x) \geq k - \alpha - i$ and $v_\ell(y) \geq k - \beta - i$; there are at most $\ell^{\min(\alpha+i,k)+\min(\beta+i,k)}$ such points, and therefore $\ell^{\min(\alpha+i,k)+\min(\beta,k-i)} \leq \ell^{k-i+\beta}$ corresponding classes. Summing this over all i proves (v). □

A.3 Computing the action of the Frobenius endomorphism

We continue here our study on the action of the Frobenius π on $E[\ell^k]$. Given an ordinary elliptic curve E with ℓ-maximal endomorphism ring, we explicitly compute diagonal and horizontal bases of $E[\ell^k]$ as defined in the previous section. We will use the latter basis of $E[\ell^k]$ in Section A.5, to put restrictions on the interpolation problem of our algorithm.

We suppose that $k \geqslant h$. By Proposition A.10, there exists a Kummer tower $F_0 \subset \cdots \subset F_{k-\beta}$ such that all the points of $E[\ell^k]$ are rational over $F_{k-\beta}$. The algorithms presented next assume that the tower has already been computed.

Computation of a diagonal basis

In Algorithm 1 below, we describe how to compute eigenvalues of the Frobenius $\bmod \ell^k$ and corresponding eigenvectors in the ℓ^k-torsion subgroup. We write $Q \leftarrow \mathsf{divide}(\ell, P)$ for the computation of a preimage of P by multiplication by ℓ.

Algorithm 1 Computing a diagonal basis of $E[\ell^k]$

Input: E: an ordinary, ℓ-maximal elliptic curve; k: a positive integer;
Output: (P_k, Q_k): a basis of $E[\ell^k]$; $\lambda, \mu \in \mathbb{Z}/\ell^k\mathbb{Z}$ such that $\pi(P_k) = \lambda P_k$, $\pi(Q_k) = \mu Q_k$.

1: $\lambda \leftarrow 0$; $\mu \leftarrow 0$; $P_0, Q_0 \leftarrow$ neutral element of $E[\ell]$.
2: **for** $i = 0$ to $k-1$ **do**
3: $\quad P' \leftarrow \mathsf{divide}(\ell, P_i)$; $Q' \leftarrow \mathsf{divide}(\ell, Q_i)$.
4: $\quad$ compute $\pi|(P', Q') = \begin{pmatrix} \lambda + a\ell^i & b\ell^i \\ c\ell^i & \mu + d\ell^i \end{pmatrix} \pmod{\ell^{i+1}}$.
5: $\quad$ **if** $b = 0$ **then** $x \leftarrow 0$; solve equation $c\ell^i + ((d-a)\ell^i + \mu - \lambda)y = 0$;
6: $\quad$ **else** solve equation $c\ell^i x^2 + ((d-a)\ell^i + \mu - \lambda)x - b\ell^i = 0$; $y \leftarrow -cx/b$; **end if**.
7: $\quad P_{i+1} \leftarrow P' + yQ'$; $Q_{i+1} \leftarrow xP' + Q'$.
8: $\quad \lambda \leftarrow \lambda + \ell^i(a + bx)$; $\mu \leftarrow \mu + \ell^i(d + cy)$.
9: **end for**
10: **return** (P_k, Q_k, λ, μ).

Proposition A.11. *Algorithm 1 computes a diagonal basis of $E[\ell^k]$ using an expected $O(\mathsf{R}(k-\beta) + \ell^2\mathsf{M}(\ell^{k-\beta}) + \ell\mathsf{M}(\ell^2)\log(\ell)\log(\ell q))$ operations in $\mathbb{F}_q$.*

Proof. The equation at line 5 or 6 is first divided out by the largest power of ℓ possible, which is $\ell^{\min(h,i)}$, then solved modulo ℓ. For $i \leq h-1$, since $a = d$ and $b = c = 0$, the solutions are $x = y = 0$, and steps 5 to 7 do nothing. A straightforward calculation shows that after each loop the basis (P_{i+1}, Q_{i+1}) is diagonal.

For $i = 0$, the basis of $E[\ell](F_1)$ at step 3 is computed by factoring the ℓ-division polynomial at an expected cost of $O(\ell\mathsf{M}(\ell^2)\log(\ell)\log(\ell q))$ operations using the Cantor-Zassenhaus algorithm. Once $E[\ell]$ has been computed, we can factor the multiplication-by-ℓ map as a product of two ℓ-isogenies. Then, for any P defined in $E(F_{i-\beta})$, the computation of $\mathsf{divide}(\ell, P)$ at Step 3 costs $O(\mathsf{R}(i-\beta+1))$ operations. Evaluating $\pi(P')$ in Step 4 has a cost of $O(\ell^{i-\beta}\mathsf{M}(\ell))$. Writing $\pi(P')$ as a linear combination $\alpha P' + \beta Q'$ needs at most ℓ^2 point additions, with a cost of $\ell^2\mathsf{M}(\ell^{i-\beta+1})$. The cost of solving the equations at Steps 5 and 6 by exhaustive search is negligible, as are the remaining operations. Since the cost of each loop grows geometrically, the last loop dominates all others, and gives the stated complexity. □

Computation of a horizontal basis

Using the previous algorithm we can compute a diagonal basis of $E[\ell^{h+1}]$. By Proposition A.7, this gives us a horizontal basis of $E[\ell]$. Thanks to Proposition A.8, we can use this information to improve horizontal points of $E[\ell^i]$ into horizontal points of $E[\ell^{i+1}]$, as illustrated in Algorithm 2.

Proposition A.12. *Algorithm 2 is correct and computes its output using an expected $O(\mathsf{R}(k-\beta) + k\mathsf{R}(h-\beta+1) + k\ell^2\mathsf{M}(\ell^{h-\beta+1}))$ operations in $\mathbb{F}_q$.*

Proof. Let E_i be the image curve of ϕ_i. We check that at step i of the loop, the points (P_i, Q_i) form a diagonal basis of $E_i[\ell^{h+1}]$, and ϕ_i has direction λ. The fact that R is horizontal is then a consequence of Proposition A.8. The two most expensive operations in the loop are Steps 4 and 5, costing respectively $O(\mathsf{R}(h-\beta+1))$ and $O(\ell^2\mathsf{M}(\ell^{h-\beta+1}))$, as discussed in the proof of Proposition A.11. They are repeated k times. Finally, Step 7 is dominated by the last divide operation, which costs $O(\mathsf{R}(k-\beta))$. □

Algorithm 2 Computing a horizontal point of order ℓ^k

Input: (P_0, Q_0): a diagonal basis of $E[\ell^{h+1}]$; k: an integer, $k \geq h+1$.
Output: R: a horizontal point of $E[\ell^k]$ with direction λ.

1: **for** $i = 1$ to $k-1$ **do**
2: $\quad \phi_i \leftarrow$ isogeny with kernel $\langle \ell^h P_{i-1} \rangle$
3: $\quad Q_i \leftarrow \phi_i(Q_{i-1})$
4: $\quad P' \leftarrow \mathsf{divide}(\ell, \phi_i(P_{i-1}))$.
5: $\quad$ Write $\pi(P') = \lambda P' + bQ_i$ for $b \in \mathbb{Z}/\ell\mathbb{Z}$ and let $P_i \leftarrow P' - (b/\mu)Q_i$.
6: **end for**
7: **return** $R = \widehat{\phi}_1 \circ \ldots \circ \widehat{\phi}_{k-1}(\mathsf{divide}(\ell^{k-(h+1)}, P_{k-1}))$.

One application of Algorithm 1 (with input $k \leftarrow h+1$) and two applications of Algorithm 2 allow us to compute a horizontal basis of $E[\ell^k]$. This could be done directly with Algorithm 1 instead, but that would require computing in an extension $F_{k+h-\beta}$.

A.4 Interpolation step

After constructing bases (P, Q) of $E[\ell^k]$ and (P', Q') of $E'[\ell^k]$ using the algorithms of the previous section, our algorithm computes the polynomial with coefficients in $\mathbb{F}_q$ mapping $x(P)$ to $x(P')$, $x(Q)$ to $x(Q')$, and the other abscissas accordingly. In this section we give an efficient algorithm for this specific interpolation problem. The algorithm appeared in [De 11] in the context of the Artin-Schreier extensions used in Couveignes' isogeny algorithm; it uses original ideas from [EM03]. We recall this algorithm here, and adapt the complexity analysis to our setting of Kummer extensions.

We start by tackling a simpler problem. We suppose we have constructed a tower of Kummer extensions $\mathbb{F}_q = F_0 \subset F_1 \subset \cdots \subset F_n$, with $[F_1 : F_0] \mid (\ell - 1)$, and $[F_{i+1} : F_i] = \ell$ for any $i > 0$. Given two elements $v, w \in F_n \setminus F_{n-1}$, we want to compute polynomials T and L such that:

- $T \in \mathbb{F}_q[x]$ is the minimal polynomial of v, of degree $d = \deg T < \ell^n$;
- L is in $\mathbb{F}_q[x]$, of degree less than d, and $L(v) = w$.

Observe that, since $v, w \notin F_{n-1}$, we necessarily have $v_\ell(d) = n-1$, so that $\ell^{n-1} \leq d < \ell^n$. Using a fast interpolation algorithm [GG99, Chapter 10.2], the polynomials T and L could be computed in $O\big(n\mathsf{M}(\ell^{2n})\log(\ell)\big)$ operations in $\mathbb{F}_q$. We can do much better by exploiting the form of the Kummer tower, and the Frobenius algorithm given in Lemma A.2.

Following [De 11], we first compute T, starting from $T^{(0)} = x - v$. We let σ_i be the map that takes all the coefficients of a polynomial in $F_{n-i}[x]$ to the power $\#F_{n-i-1}$. For $i = 0, \ldots, n-1$, suppose we know a polynomial $T^{(i)}$ of degree ℓ^i in $F_{n-i}[x]$. Then, compute the polynomials $T^{(i,j)}$ given by $T^{(i,j)} = \sigma_i^j\big(T^{(i)}\big)$ for $0 \leq j \leq \ell - 1$, and define

$$T^{(i+1)} = \prod_{j=0}^{b} T^{(i,j)} \qquad \text{with} \qquad b = \begin{cases} \ell - 1 & \text{if } i < n-1, \\ d/\ell^{n-1} & \text{otherwise.} \end{cases} \tag{A.1}$$

One easily sees that $T^{(i+1)}$ is the minimal polynomial of v over F_{n-i+1}.

Lemma A.13. *The cost of computing T is $O(n\mathsf{M}(\ell^{n+1})\log(\ell))$ operations in $\mathbb{F}_q$.*

Proof. At each step i, from the knowledge of $T^{(i)}$ we compute all $T^{(i,j)}$ using Lemma A.2. The cost for a single polynomial $T^{(i,j)}$ is of $O(\ell^i \ell^{n-i-1}\mathsf{M}(\ell))$ operations, *i.e.* $O(\ell^n\mathsf{M}(\ell))$ for all $O(\ell)$ of them. From the $T^{(i,j)}$'s we compute $T^{(i+1)}$ using a subproduct tree, as in [GG99, Lemma 10.4]. The result has degree $O(\ell^{i+1})$ and coefficients in F_{n-i}, thus the overall cost is $O(\mathsf{M}(\ell^{n+1})\log(\ell))$. After $T^{(i+1)}$ is computed this way, we can convert its coefficients to F_{n-i-1} at no algebraic cost. Summing over all i, we obtain the stated complexity. □

We can finally proceed with the interpolation itself. First, compute $w' = w/T'(v)$ and let $L^{(0)} = w'$. Next, for $i = 0, \dots, n-2$, suppose we know a polynomial $L^{(i)}$ in $F_{n-i}[x]$ of degree less than ℓ^i. We compute the polynomials $L^{(i,j)}$ given by $L^{(i,j)} = \sigma_i^j\left(L^{(i)}\right)$ and

$$L^{(i+1)} = \sum_{j=0}^{b} L^{(i,j)} \frac{T^{(i+1)}}{T^{(i,j)}}, \qquad b \text{ defined as in Eq. (A.1).}$$

As shown in [De 11], $L^{(n)}$ is the polynomial L we are looking for.

Proposition A.14. *Given $v, w \in F_n \setminus F_{n-1}$, the cost of computing the minimal polynomial $T \in \mathbb{F}_q[x]$ of v and the interpolating polynomial $L \in \mathbb{F}_q[x]$ such that $L(v) = w$ is $O(n\mathsf{M}(\ell^{n+1})\log(\ell))$ operations in $\mathbb{F}_q$.*

Proof. After the polynomials $T^{(i)}$ have been computed, we need to compute $T'(v)$. This is done by means of successive Euclidean remainders, since $T'(v) = (((T' \bmod T^{(1)}) \bmod T^{(2)}) \cdots \bmod T^{(n)})$. At stage i, we have to compute the Euclidean division of a polynomial of degree $O(\ell^{n-i+1})$ by one of degree $O(\ell^{n-i})$ in $F_i[x]$. Using the complexities from Section A.1 we deduce that each division can be done in time $O(\mathsf{M}(\ell^{n+1}))$, for a total of $O(n\mathsf{M}(\ell^{n+1}))$ operations. Then, computing $w' = w/T'(v)$ takes $O(\mathsf{M}(\ell^n)\log(\ell^n))$ operations.

Finally, at each step i, the polynomials $L^{(i,j)}$ are computed at a cost of $O(\ell^n\mathsf{M}(\ell))$, as in the proof of Lemma A.13. The computation of $L^{(i+1)}$ uses the same subproduct tree as for the computation of $T^{(i)}$, requiring $O(\log \ell)$ additions, multiplications and divisions of polynomials of degree $O(\ell^{i+1})$ with coefficients in F_{n-i}, for a total of $O(\mathsf{M}(\ell^{n+1})\log(\ell))$. Summing over all i, the complexity statement follows readily. □

We end with the general problem of interpolating a polynomial in $\mathbb{F}_q[x]$ at points of F_n.

Proposition A.15. *Let $(v_1, w_1), \dots, (v_s, w_s)$ be pairs of elements of F_n, let t_i be the degree of the minimal polynomial of v_i, and let $t = \sum t_i$. The polynomials*

- *$T \in \mathbb{F}_q[x]$ of degree t such that $T(v_i) = 0$ for all i, and*
- *$L \in \mathbb{F}_q[x]$ of degree less than t such that $L(v_i) = w_i$ for all i*

can be computed using $O\big(\mathsf{M}(t)\log(s) + n\mathsf{M}(\ell^2 t)\log(\ell)\big)$ operations in $\mathbb{F}_q$.

Proof. The polynomial T is simply the product of all the minimal polynomials T_i. Let $n_i = v_\ell(t_i)$, so that $v_i, w_i \in F_{n_i+1} \setminus F_{n_i}$, and $\ell^{n_i} \le t_i < \ell^{n_i+1}$. We convert (v_i, w_i) to a pair of elements of F_{n_i+1} at no algebraic cost, then we compute T_i as explained previously at a cost of $O(n\mathsf{M}(\ell^{n_i+2})\log(\ell))$ operations. Bounding ℓ^{n_i} by t_i, summing over all i, and using the superlinearity of M, we obtain a total cost of $O(n\mathsf{M}(\ell^2 t)\log(\ell))$ operations.

Simultaneously, we compute all the polynomials L_i such that $L_i(v_i) = w_i$, at the same cost.

Then we arrange the T_i's into a binary subproduct tree and multiply them together. A balanced binary tree, though not necessarily optimal, has a depth of $O(\log(s))$, and requires $O(\mathsf{M}(t))$ operations per level. Thus we can bound the cost of computing T by $O(\mathsf{M}(t)\log(s))$.

Finally, using the same subproduct tree structure, we apply the Chinese remainder algorithm of [GG99, Chapter 10] to compute the polynomial L at the same cost $O(\mathsf{M}(t)\log(s))$. □

A.5 The complete algorithm

We finally come to the description of the full algorithm. Given two j-invariants, defining two elliptic curves E and E', and an integer r, we want to compute an isogeny $\psi : E \to E'$ of degree r. Since the algorithms of Section A.3 apply to curves on top of volcanoes with cyclic crater, we first need to determine a small Elkies prime ℓ for E and E', and then reduce to an explicit isogeny problem on the crater of the ℓ-volcanoes. These steps are discussed and analyzed next.

Finding a suitable ℓ-volcano

Our algorithm uses an Elkies prime ℓ. Since d_K is not assumed to be known yet, we need to be able to compute the height h of the volcano, the shape of its crater, as well as the shortest ℓ-isogeny chain from E to the crater.

The algorithms of Fouquet and Morain [FM02] compute the height h and find a curve $E_{\max}$ on the crater at the cost of $O(\ell h^2)$ factorizations of the ℓ-th modular polynomial Φ_ℓ. The polynomial Φ_ℓ is computed using $\tilde{O}(\ell^3\log(\ell))$ boolean operations, then each factorization costs an expected $O(\mathsf{M}(\ell)\log(\ell)\log(\ell q))$ operations using the Cantor-Zassenhaus algorithm (more efficient methods for special instances of volcanoes are presented in [Mir+05] and in [IJ13], but we do not discuss them). Working on E and E', we compute the shortest path of ℓ-isogenies $\alpha : E \to E_{\max}$, $\alpha' : E' \to E'_{\max}$ linking the curves E, E' to the craters. We still have to determine the shape of these craters. Since the height h of the volcano is known, using Algorithm 1 we can compute a matrix of $\pi|E_{\max}[\ell^{h+1}]$. If this matrix has two distinct eigenvalues then the crater is cyclic, otherwise it is not.

By Proposition A.9, the depth of E and E' below their respective craters is the same. By Proposition A.9 3, the curves $E_{\max}$ and $E'_{\max}$ are again r-isogenous; we can use our algorithm to compute such an isogeny $\psi_{\max}$. Then, since ℓ is coprime to r, $\psi = (\alpha')^{-1} \circ \psi_{\max} \circ \alpha$ is well defined and is the required r-isogeny. Its kernel can be computed in $O(h\mathsf{M}(\ell r)\log(\ell r))$ operations by evaluating the dual isogeny $\hat{\alpha}$ on the kernel of $\psi_{\max}$ via a sequence of resultants.

Interpolating the isogeny

We now assume that both curves E, E' have ℓ-maximal endomorphism rings. We fix bases of $E[\ell^k]$, $E'[\ell^k]$ and write π, π' for the matrices of the Frobenius. Since ψ is rational, its matrix satisfies the relation $\pi' \cdot \psi = \psi \cdot \pi$ in $\mathbb{Z}_\ell^{2\times 2}$ and hence in $(\mathbb{Z}/\ell^k\mathbb{Z})^{2\times 2}$.

If diagonal bases of $E[\ell^k], E'[\ell^k]$ are used, then, since π is a cyclic endomorphism of $\mathbb{Z}_\ell^2$, this condition seems to ensure that ψ is a diagonal matrix; however, $\mathbb{Z}/\ell^k\mathbb{Z}$ is not an integral domain and π is congruent, modulo ℓ^h, to the scalar matrix λ, so we

can only say that $\psi \pmod{\ell^{k-h}}$ is diagonal. If on the other hand we choose *horizontal* bases of $E[\ell^k], E'[\ell^k]$ then, by Proposition A.9 2, we know that ψ is a diagonal matrix.

We then enumerate all the ℓ^{2k-2} invertible diagonal matrices; for each matrix M, we interpolate the action of M on $E[\ell^k]$ as a rational fraction, and verify that it is an r-isogeny. The successful interpolation will be our explicit isogeny ψ. Precisely, we interpolate using the abscissas of non-zero points of $E[\ell^k]$; there are $(\ell^{2k}-1)/2$ distinct such abscissas (or $2^{2k-1}+1$ when $\ell = 2$). The isogeny ψ acts on abscissas as a rational fraction of degrees $(r, r-1)$, which is thus defined by $2r$ coefficients; knowing this rational function allows us to find the kernel of ψ, and recover ψ itself using Vélu's formulas. For this method to work, we therefore select the smallest $k \geq h+1$ such that $\ell^{2k} - 1 > 4r$.

Summarizing, our algorithm for two ℓ-maximal curves proceeds as follows:

1. Use Algorithms 1 and 2 to compute horizontal bases $(P,Q), (P',Q')$ of $E[\ell^k], E'[\ell^k]$;
2. Compute the polynomial T vanishing on the abscissas of $\langle P,Q\rangle$ as in Section A.4;
3. For each invertible diagonal matrix $\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$ in $(\mathbb{Z}/\ell^k\mathbb{Z})^{2\times 2}$:
   a) compute the interpolation polynomial $L_{a,b}$ such that $L_{a,b}(x(uP+vQ)) = x(au P' + bv Q')$ for all $u, v \in \mathbb{Z}/\ell^k\mathbb{Z}$;
   b) Use the *Cauchy interpolation algorithm* of [GG99, Chapter 5.8] to compute a rational fraction $F_{a,b} \equiv L_{a,b} \pmod{T}$ of degrees $(r, r-1)$;
   c) If $F_{a,b}$ defines an isogeny of degree r, return it and stop.

Proposition A.16. *Assuming that $\ell^h < \sqrt{r}$, the algorithm above computes an isogeny $\psi : E \to E'$ in expected time $O\Big(\big(r\ell^2\mathsf{M}(r\ell^4) + \mathsf{M}(r\ell^3)\log(\ell q)\big)\log(r)\log(\ell)\Big)$.*

Proof. By definition of k, we know that $\ell^{2k} \in O(r\ell^2)$. By Proposition A.10, there is a $\beta < h$ such that $E[\ell^k]$ is contained in $E(F_n)$ with $n = k - \beta$. We thus construct the Kummer tower $F_0 \subset \cdots \subset F_n$, and we do the precomputations required by Lemma A.2 at a cost of $O(\ell\mathsf{M}(\ell)\log(q))$.

Bounding h by $k-1$, Step 1 costs on average $O(k\mathsf{R}(k-\beta) + k\ell^2\mathsf{M}(\ell\sqrt{r}) + \ell\mathsf{M}(\ell^2)\log(\ell)\log(\ell q))$ according to Propositions A.11 and A.12. Using the most pessimistic estimates of Section A.1, we see that this cost is bounded by $O(\mathsf{M}(r\ell^3)\log(r)\log(\ell)\log(\ell q))$.

By Proposition A.10 5, there are at most $O(k\cdot\ell^{k+\beta})$ Galois classes in $E[\ell^k]$. In order to apply the algorithms of Section A.4, we need to compute a representative for each class. Each representative is computed from the basis (P,Q) using point multiplication by two scalars $\leq \ell^k$ in the field F_n, which costs $O(\mathsf{M}(\ell^n)\log(\ell^k))$ operations. We thus have a total cost of $O(k\mathsf{M}(\ell^{2k})\log(\ell^k)) \subset O(\mathsf{M}(r\ell^2)\log(r)^2)$ to compute all such representatives.

Then, using Proposition A.15, where the total degree is $t = (\ell^{2k}-1)/2 \in O(r\ell^2)$, and the number of interpolation points is $s \in O(k\cdot\ell^{k+\beta})$, we can compute the polynomials T and $L_{a,b}$ at a cost of $O(\mathsf{M}(r\ell^4)\log(r)\log(\ell))$. The cost of computing $F_{a,b}$, and identifying the isogeny, is dominated by that of computing $L_{a,b}$ [De 11, §3.3]. Finally, in general approximately $\ell^{2k} = O(r\ell^2)$ candidate matrices must be tried before finding the isogeny. □

Overall complexity

By a result of Shparlinski and Sutherland [SS14, Theorem 1], for almost all primes q and curves $E/\mathbb{F}_q$, for $L \geq \log(q)^\varepsilon$ for any $\varepsilon > 0$, asymptotically half of the primes $\ell \leq L$ are Elkies primes. Hence, we expect to have enough small Elkies primes to apply our algorithm. The following theorem states a worst case bound depending on r and q alone.

Theorem. For almost all primes q and curves E, E' over $\mathbb{F}_q$, it is possible to solve the "Explicit Isogeny Problem" in expected time $O\big(r\mathsf{M}(r\log(q)^6)\log(r)\operatorname{loglog}(q)\big)$.

Proof. Given a curve E, we search for the smallest Elkies prime satisfying the conditions of Proposition A.16. As a special case of [SS14, Theorem 1], we can take $L \in O(\log(q))$ such that the product of all Elkies primes $\ell \leq L$ exceeds $\Omega(\sqrt{q})$. On the other hand, we discard those primes $\ell \leq L$ for which the height h satisfies $\ell^h > \sqrt{r}$; since those discarded primes are divisors of $\sqrt{d_K}$, their product is at most $O(\sqrt{q})$. This shows that there remains enough "good" Elkies primes in $[\![1, L]\!]$, so that in the worst case $\ell \in O(\log(q))$.

The most expensive steps in Section A.5 are the computation and the factorization of the modular polynomials for all primes up to ℓ. This is well within $O(\log(q)^6)$. The stated complexity follows then from substituting $\ell = O(\log(q))$ in Proposition A.16. □

A.6 Conclusion and experimental results

In the previous sections we have obtained a Las Vegas algorithm with an interesting asymptotic complexity. In particular, in the favorable case where $\ell = O(1)$, the running time of the algorithm is quasi-quadratic in the isogeny degree r and quasi-linear in $\log q$. Thus we expect it to be practical, and a substantial improvement over Couveignes' original algorithm, at least when small parameters ℓ and h can be found quickly. A large ℓ or h adversely affects performance in the following ways:

- All modular polynomials up to Φ_ℓ must be computed or retrieved from tables.
- All degrees $(\ell^{2k}-1)/4 \leq r < (\ell^{2k+1}-1)/4$ require essentially the same computational effort, thus resulting in a *staircase behavior* when r increases.
- Because we must have $k > h$, all degrees r smaller than $(\ell^{2h+2}-1)/4$ require the same computational effort.

For these reasons, it is wisest in practice to set small *a priori* bounds on ℓ and h, and only run our algorithm when parameters within these bounds can be found.

To validate our findings, we implemented a simplified version of our main algorithm using SageMath v7.1 [Sage]. In our current implementation, we only handle the case $\ell = 2$ and we work only with curves on the crater of a 2-volcano. We implemented the construction of Kummer towers described in [DS15], in the favorable case where $p = 1 \bmod 4$. Source code and benchmark data are available in the GitHub project https://github.com/Hugounenq-Cyril/Two_curves_on_a_volcano/.

We ran benchmarks on an Intel Xeon E5530 CPU clocked at 2.4GHz. We fixed a base field $\mathbb{F}_q$ and an elliptic curve E with height $h = 3$ and $\beta = 2$, then ran our algorithm to compute the multiplication-by-r isogeny $E \to E$, for r increasing. The torsion levels involved in the computations varied from 2^3 to 2^8. Figure A.2 (left) shows the running times for the computation of the horizontal basis of $E[\ell^k]$, and for one execution of the

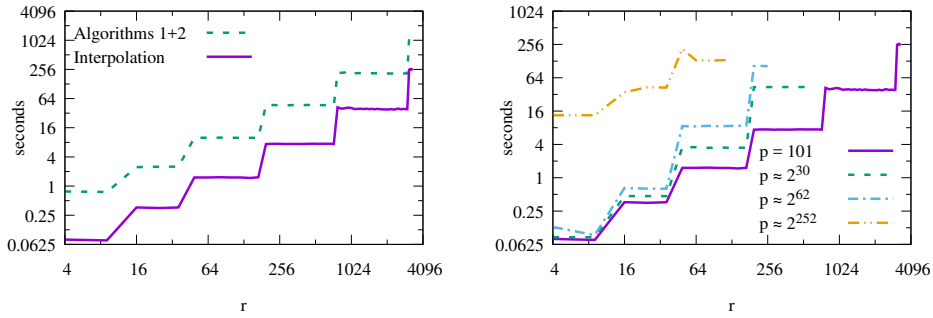


Figure A.2: Left: comparison of horizontal basis computation and interpolation phases, for a fixed curve defined over $\mathbb{F}_{101}$, and increasing r. Right: Comparison of one interpolation phase for $\mathbb{F}_{101}$, $\mathbb{F}_{2^{30}+669}$, $\mathbb{F}_{2^{62}+189}$ and $\mathbb{F}_{2^{252}+421}$, and increasing r. Plots in logarithmic scale.

interpolation step. Running times are close to linear in r, as expected. The staircase behavior of our algorithm is apparent from the plot. Since the interpolation steps must be repeated $\sim r$ times, we focus on this step to compare the running time for different base fields. In Figure A.2 (right) we observe that the dependency in q, although much better than in Couveignes' original algorithm, is higher than what the theoretical analysis would predict. This may due to low-level implementation details of SageMath, which, in the current implementation, are beyond our control.

In conclusion, our algorithm shows promise of being of practical interest within selected parameter ranges. Generalizing it to work with Atkin primes would considerably enlarge its applicability range; we hope to develop such a generalization in a future work. On the practical side, we plan to work on two improvements that seem within reach. First, the reduction from generic curves to ℓ-maximal curves seems superfluous and unduly expensive: it would be interesting to generalize the concept of horizontal bases to any curve. Second, a multi-modular approach interpolating on a torsion group of composite order is certainly possible, and could improve the running time of our algorithm by allowing it to work in smaller extension fields.

A.7 Galois classes in $E[\ell^k]$

We give here the full decomposition of $E[\ell^k]$ in Galois classes. This is a more precise form of Proposition A.10 (v).

Proposition A.17. *Let E be an elliptic curve with ℓ-maximal endomorphism ring. Assume $\ell \neq 2$, $\lambda \equiv \mu \equiv 1 \pmod{\ell}$ and let $\alpha = v_\ell(\lambda - 1), \beta = v_\ell(\mu - 1)$. Write $\nu(x,y) = \min(x+y, x+\beta-1, y+\alpha-1)$ and $\rho(x,y) = x+y-\nu(x,y) = \max(0, x-\alpha+1, y-\beta+1)$. The decomposition of the group $E[\ell^k]$ in Galois classes is as follows:*

1. *for $i,j = 1,\dots,k-1$: $(\ell-1)^2 \cdot \ell^{\nu(i,j)}$ classes of size $\ell^{\rho(i,j)}$;*
2. *for $i = 1,\dots,k-1$: $(\ell-1)\cdot\ell^{\min(i,\alpha-1)}$ classes of size $\ell^{\max(0,i-\alpha+1)}$, and $(\ell-1)\cdot\ell^{\min(i,\beta-1)}$ classes of size $\ell^{\max(0,i-\beta+1)}$;*
3. *the ℓ^2 singleton classes of $E[\ell]$.*

Proof. Fix a basis (P,Q) of $E[\ell^k]$ such that $\pi(P)=\lambda P$, $\pi(Q)=\mu Q$. Studying the Galois orbits of $E[\ell^k]$ means studying the map $\mathbb{Z}_\ell^2 \to \mathbb{Z}_\ell^2, (x,y)\mapsto(\lambda x,\mu y)$. In other words, the orbits correspond to elements of $\mathbb{Z}_\ell^2$ modulo the *multiplicative* subgroup generated by (λ,μ). An easy way to describe this is to consider a *multiplicative lattice* in $(\mathbb{Q}_\ell^\times)^2$.

Let ξ be a primitive $(\ell-1)$-th root of unity in $\mathbb{Z}_\ell$. Then by [Ser70, Théorème II.3.2], the map $f(x,y,z)=\ell^x\cdot\xi^y\cdot\exp(\ell z)$ is a group isomorphism between $\mathbb{Z}\times(\mathbb{Z}/(\ell-1)\mathbb{Z})\times\mathbb{Z}_\ell$ and $\mathbb{Q}_\ell^\times$. For $i\in[\![0,k-1]\!]$ and $c\in\mathbb{Z}/(\ell-1)\mathbb{Z}$, let $V(i,c)$ be the image in $\mathbb{Z}/\ell^k\mathbb{Z}$ of the map $f(k-1-i,c,-)$: then the multiplicative structure of $V(i,c)$ is that of a principal homogeneous space under $\mathbb{Z}/\ell^i\mathbb{Z}$. We also define $W(i,j,c,d)=V(i,c)\cdot P+V(j,d)\cdot Q\subset E[\ell^k]$.

Since $\lambda\equiv 1 \pmod{\ell}$, we may write $\lambda=f(0,0,u\ell^{\alpha-1})$ and $\mu=f(0,0,v\ell^{\beta-1})$ for some $u,v\in\mathbb{Z}_\ell^\times$. This implies that the set $W(i,j,c,d)$ is stable under Galois. Moreover, the orbits of $W(i,j,c,d)$ correspond bijectively to points of a fundamental domain of the lattice $\Lambda_{i,j}$ generated by the columns of $\begin{pmatrix}\ell^i & 0 & u\ell^{\alpha-1}\\ 0 & \ell^j & v\ell^{\beta-1}\end{pmatrix}$, whereas the size of each orbit is $[(\mathbb{Z}/\ell^i\mathbb{Z})\times(\mathbb{Z}/\ell^j\mathbb{Z}) : \Lambda_{i,j}]$. By using elementary column manipulations, we find that the covolume of $\Lambda_{i,j}$ is $\ell^{v(i,j)}$, hence the point (i) of the proposition. (The case $i=j=0$ yields singleton classes in $E[\ell]$).

The union of all the sets $W(j,i,c,d)$ is exactly the set of all $xP+yQ$ for $x,y\neq 0$. We obtain the classes of (ii) by considering the sets $V(i,c)\cdot P$ and $V(j,d)\cdot Q$. □

We now state the equivalent proposition when $\ell=2$. The proof is much the same as in the odd case.

Proposition A.18. *Let E be an elliptic curve with 2-maximal endomorphism ring. Assume $\lambda\equiv\mu\equiv 1 \pmod 4$ and let $\alpha=v_2(\lambda-1),\beta=v_2(\mu-1)$. Write $v_2(x,y)=\min(x+y,x+\beta-2,y+\alpha-2)$ and $\rho_2(x,y)=x+y-v_2(x,y)=\max(0,x-\alpha+2,y-\beta+2)$. The decomposition of the group $E[2^k]$ in Galois classes is as follows:*

1. *for $i,j=1,\dots,k-2$: $4\cdot 2^{v_2(i,j)}$ classes of size $2^{\rho_2(i,j)}$;*
2. *for $i=1,\dots,k-2$: $4\cdot 2^{\min(i,\alpha-2)}$ classes of size $2^{\max(0,i-\alpha+2)}$, and $4\cdot 2^{\min(i,\beta-2)}$ classes of size $2^{\max(0,i-\beta+2)}$.*
3. *the 16 singleton classes of $E[4]$.*

Note that if λ or $\mu\equiv -1 \pmod 4$ then by replacing the base field by a quadratic extension, we can always ensure that the condition $\lambda\equiv\mu\equiv 1 \pmod 4$ is satisfied.

A.8 References for "Explicit isogenies in any characteristic"

[Atk91] Arthur O. L. Atkin. *The number of points on an elliptic curve modulo a prime*. 1991. URL: http://www.lix.polytechnique.fr/Labo/Francois.Morain/AtkinEmails/19910614.txt.

[BLS12] Reinier Bröker, Kristin Lauter, and Andrew Sutherland. "Modular polynomials via isogeny volcanoes". In: *Mathematics of Computation* 81.278 (2012), pp. 1201–1231. DOI: 10.1090/S0025-5718-2011-02508-1.

[Bos+08] Alin Bostan, François Morain, Bruno Salvy, and Éric Schost. "Fast algorithms for computing isogenies between elliptic curves". In: *Mathematics of Computation* 77.263 (Sept. 2008), pp. 1755–1778. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-08-02066-8.

[CCR91] Leonard S Charlap, Raymond Coley, and David P Robbins. *Enumeration of rational points on elliptic curves over finite fields*. Preprint. 1991.

[CGL09] Denis X. Charles, Eyal Z. Goren, and Kristin E. Lauter. "Cryptographic Hash Functions from Expander Graphs". In: *Journal of Cryptology* 22.1 (Jan. 2009), pp. 93–113. ISSN: 0933-2790. DOI: 10.1007/s00145-007-9002-x.

[Cou00] Jean-Marc Couveignes. "Isomorphisms between Artin-Schreier towers". In: *Mathematics of Computation* 69.232 (2000), pp. 1625–1631. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-00-01193-5.

[Cou94] Jean-Marc Couveignes. "Quelques calculs en théorie des nombres". PhD thesis. Université de Bordeaux, 1994.

[Cou96] Jean-Marc Couveignes. "Computing ℓ-isogenies using the p-Torsion". In: *ANTS-II: Proceedings of the Second International Symposium on Algorithmic Number Theory*. London, UK: Springer-Verlag, 1996, pp. 59–65. ISBN: 3-540-61581-4.

[DDS13] Luca De Feo, Javad Doliskani, and Éric Schost. "Fast Algorithms for ℓ-adic Towers over Finite Fields". In: *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*. ISSAC '13. New York, NY, USA: ACM, 2013, pp. 165–172. ISBN: 978-1-4503-2059-7. DOI: 10.1145/2465506.2465956.

[De 11] Luca De Feo. "Fast algorithms for computing isogenies between ordinary elliptic curves in small characteristic". In: *Journal of Number Theory* 131.5 (May 2011), pp. 873–893. ISSN: 0022-314X. DOI: 10.1016/j.jnt.2010.07.003.

[DJP14] Luca De Feo, David Jao, and Jérôme Plût. "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies". In: *Journal of Mathematical Cryptology* 8.3 (2014), pp. 209–247.

[DS12] Luca De Feo and Éric Schost. "Fast arithmetics in Artin-Schreier towers over finite fields". In: *Journal of Symbolic Computation* 47.7 (2012), pp. 771–792. ISSN: 07477171. DOI: 10.1016/j.jsc.2011.12.008.

[DS15] Javad Doliskani and Éric Schost. "Computing in degree 2^k-extensions of finite fields of odd characteristic". In: *Designs, Codes and Cryptography* 74.3 (2015), pp. 559–569.

[Elk98] Noam D. Elkies. "Elliptic and modular curves over finite fields and related computational issues". In: *Computational perspectives on number theory (Chicago, IL, 1995)*. Vol. 7. Studies in Advanced Mathematics. Providence, RI: AMS International Press, 1998, pp. 21–76. URL: http://www.ams.org/mathscinet-getitem?mr=1486831.

[EM03] Andreas Enge and François Morain. "Fast decomposition of polynomials with known Galois group". In: *AAECC'03: Proceedings of the 15th international conference on Applied algebra, algebraic algorithms and error-correcting codes*. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 254–264. ISBN: 3-540-40111-3.

[FM02] Mireille Fouquet and François Morain. “Isogeny Volcanoes and the SEA Algorithm”. In: *Algorithmic Number Theory Symposium*. Ed. by Claus Fieker and David R. Kohel. Vol. 2369. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2002. Chap. 23, pp. 47–62. ISBN: 978-3-540-43863-2. DOI: 10.1007/3-540-45455-1_23.

[GG99] Joachim von zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. New York, NY, USA: Cambridge University Press, 1999. ISBN: 0-521-64176-4.

[GLV01] Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone. “Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms”. In: *CRYPTO ’01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 2001, pp. 190–200. ISBN: 3-540-42456-3. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.2004.

[GS92] Joachim von zur Gathen and Victor Shoup. “Computing Frobenius Maps and Factoring Polynomials”. In: *Computational Complexity* 2 (1992), pp. 187–224.

[IJ13] Sorina Ionica and Antoine Joux. “Pairing the volcano”. In: *Mathematics of Computation* 82.281 (2013), pp. 581–603.

[JS14] David Jao and Vladimir Soukharev. “Isogeny-based quantum-resistant undeniable signatures”. In: *Post-Quantum Cryptography: 6th International Workshop, PQCrypto 2014*. Waterloo, ON, Canada: Springer International Publishing, 2014, pp. 160–179. ISBN: 978-3-319-11659-4. DOI: 10.1007/978-3-319-11659-4_10.

[Koh96] David R. Kohel. “Endomorphism rings of elliptic curves over finite fields”. PhD thesis. University of California at Berkeley, 1996.

[KS97] Erich Kaltofen and Victor Shoup. “Fast polynomial factorization over high algebraic extensions of finite fields”. In: *ISSAC ’97: Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*. New York, NY, USA: ACM, 1997, pp. 184–188. ISBN: 0-89791-875-4. DOI: 10.1145/258726.258777.

[LS08b] Reynald Lercier and Thomas Sirvent. “On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field”. In: *Journal de théorie des nombres de Bordeaux* 20.3 (2008), pp. 783–797. URL: http://perso.univ-rennes1.fr/reynald.lercier/file/LS08.pdf.

[LS14] Patrick Longa and Francesco Sica. “Four-Dimensional Gallant–Lambert–Vanstone Scalar Multiplication”. In: *Journal of Cryptology* 27.2 (2014), pp. 248–283. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9144-3.

[LV16] Pierre Lairez and Tristan Vaccon. “On p-adic Differential Equations with Separation of Variables”. In: *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*. ISSAC ’16. New York, NY, USA: ACM, 2016, pp. 319–323. ISBN: 978-1-4503-4380-0. DOI: 10.1145/2930889.2930912.

[Mir+05] Josep M. Miret, Ramiro Moreno, Ana Rio, and Magda Valls. “Determining the 2-Sylow subgroup of an elliptic curve over a finite field”. In: *Mathematics of Computation* 74.249 (2005), pp. 411–427. DOI: 10.1090/S0025-5718-04-01640-0.

[MMT01] Markus Maurer, Alfred Menezes, and Edlyn Teske. “Analysis of the GHS Weil Descent Attack on the ECDLP over Characteristic Two Finite Fields of Composite Degree”. In: *INDOCRYPT ’01: Proceedings of the Second International Conference on Cryptology in India*. Berlin: Springer-Verlag, 2001, pp. 195–213. ISBN: 3-540-43010-5.

[Sage] *SageMath, the Sage Mathematics Software System (Version 8.0)*. The Sage Developers. 2018. URL: https://www.sagemath.org.

[Sch85] René Schoof. “Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p”. In: *Mathematics of Computation* 44.170 (1985), pp. 483–494. ISSN: 00255718. DOI: 10.2307/2007968.

[Sch95] René Schoof. “Counting points on elliptic curves over finite fields”. In: *Journal de Théorie des Nombres de Bordeaux* 7.1 (1995), pp. 219–254. URL: http://www.ams.org/mathscinet-getitem?mr=1413578.

[Ser70] Jean-Pierre Serre. *Cours d’arithmétique*. Paris: Presses Universitaires de France, 1970.

[Ser77] Jean-Pierre Serre. *Arbres, amalgames, SL_2*. Vol. 46. Astérisque. Paris: Société Mathématique de France, 1977.

[Sil92] Joseph H. Silverman. *The arithmetic of elliptic curves*. Vol. 106. Graduate Texts in Mathematics. Corrected reprint of the 1986 original. New York: Springer-Verlag, 1992, pp. xii+400. ISBN: 0-387-96203-4.

[SS14] Igor E. Shparlinski and Andrew V. Sutherland. “On the Distribution of Atkin and Elkies Primes”. In: *Foundations of Computational Mathematics* 14.2 (Apr. 2014), pp. 285–297. ISSN: 1615-3383. DOI: 10.1007/s10208-013-9181-9.

[Sto10] Anton Stolbunov. “Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves”. In: *Advances in Mathematics of Communications* 4.2 (2010).

[Sut13a] Andrew Sutherland. “Isogeny volcanoes”. In: *ANTS X: Proceedings of the Algorithmic Number Theory 10th International Symposium*. Vol. 1. Berkeley: Mathematical Sciences Publishers, 2013, pp. 507–530.

[Sut13b] Andrew Sutherland. “On the evaluation of modular polynomials”. In: *The Open Book Series* 1.1 (2013), pp. 531–555.

[Tat66] John Tate. “Endomorphisms of abelian varieties over finite fields”. In: *Inventiones mathematicae* 2.2 (Apr. 1966), pp. 134–144. ISSN: 1432-1297. DOI: 10.1007/BF01404549.

[Tes06] Edlyn Teske. “An Elliptic Curve Trapdoor System”. In: *Journal of Cryptology* 19.1 (Jan. 2006), pp. 115–133. ISSN: 0933-2790. DOI: 10.1007/s00145-004-0328-3.

[Vél71] Jacques Vélu. “Isogénies entre courbes elliptiques”. In: *Comptes Rendus de l’Académie des Sciences de Paris* 273 (1971), pp. 238–241.

B Fast arithmetic for the algebraic closure of finite fields

Abstract

We present algorithms to construct and do arithmetic operations in the algebraic closure of the finite field $\mathbb{F}_p$. Our approach is inspired by algorithms for constructing irreducible polynomials, which first reduce to prime power degrees, then use composita techniques. We use similar ideas to give efficient algorithms for embeddings and isomorphisms.

B.1 Introduction

Several computer algebra systems or libraries, such as Magma [BCP97], Sage [Sage], NTL [NTL], PARI [PARI] or Flint [Har10], offer built-in features to build and compute in arbitrary finite fields. At the core of these designs, one finds algorithms for building irreducible polynomials and algorithms to compute embeddings and isomorphisms. The system used in Magma (one of the most complete we know of) is described in [BCS97a].

Previous algorithms typically rely on linear algebra techniques, for instance to describe embeddings or isomorphisms (this is the case for the algorithms in [BCS97a], but also for those in [Len91; All02a]). Unfortunately, linear algebra techniques have cost at least quadratic in the degree of the extensions we consider, and (usually) quadratic memory requirements. Our goal here is to replace linear algebra by polynomial arithmetic, exploiting fast polynomial multiplication to obtain algorithms of quasi-linear complexity. As we will see, we meet this goal for several, but not all, operations.

Setup. Let p be a prime (that will be fixed throughout this paper). We are interested in describing extensions $\mathbb{F}_{p^n}$ of $\mathbb{F}_p$; such an extension has dimension n over $\mathbb{F}_p$, so representing an element in it involves n base field elements.

It is customary to use polynomial arithmetic to describe these extensions (but not necessary: Lenstra's algorithm [Len91] uses a multiplication tensor). For an extension degree n, a first step is to construct an irreducible polynomial Q_n of degree n in $\mathbb{F}_p[x]$. Identifying $\mathbb{F}_{p^n}$ with $\mathbb{F}_p[x]/\langle Q_n\rangle$, operations $(+,\times,\div)$ in $\mathbb{F}_p[x]/\langle Q_n\rangle$ all take quasi-linear time in n.

However, this is not sufficient: we also want mechanisms for e.g. field embeddings. Given irreducible polynomials Q_m and Q_n over $\mathbb{F}_p$, with $\deg(Q_m)=m$ dividing $\deg(Q_n)=n$, there exist algorithms to embed $\mathbb{F}_p[x]/\langle Q_m\rangle$ in $\mathbb{F}_p[x]/\langle Q_n\rangle$ (for the system to be consistent, these embeddings must be *compatible* [BCS97a]). However, most algorithms use linear algebra techniques.

To bypass these issues, we use an approach inspired by Shoup's algorithm for computing irreducible polynomials [Sho90; Sho94b] (see also [CL13; LS08a]): first reduce to the case of prime power degrees, then use composita techniques, in a manner that ensures compatibility of the embeddings automatically.

Background: towers. Suppose that for any prime ℓ, an *ℓ-adic tower* over $\mathbb{F}_p$ is available. By this, we mean a family of polynomials $(T_{\ell,i})_{i\geq 1}$, with $T_{\ell,i}\in\mathbb{F}_p[x_1,\ldots,x_i]$, monic of degree ℓ in x_i, such that for all i the ideal $\langle T_{\ell,1},\ldots,T_{\ell,i}\rangle$ is maximal in $\mathbb{F}_p[x_1,\ldots,x_i]$. Our model of the field with p^{ℓ^i} elements could then be $\mathbb{K}_{\ell^i}=\mathbb{F}_p[x_1,\ldots,x_i]/\langle T_{\ell,1},\ldots,T_{\ell,i}\rangle$, but we prefer to work with univariate polynomials (the cost of arithmetic operations is higher in the multivariate basis).

For $1\leq i\leq n$, let then $Q_{\ell,i}$ be the minimal polynomial of x_i in the extension $\mathbb{K}_{\ell^n}/\mathbb{F}_p$. This polynomial does not depend on n, but only on i; it is monic, irreducible of degree ℓ^i in $\mathbb{F}_p[x_i]$ and allows us to define $\mathbb{F}_{p^{\ell^i}}$ as $\mathbb{F}_p[x_i]/\langle Q_{\ell,i}\rangle$. For $1\leq i\leq j\leq n$, let further

$Q_{\ell,i,j-i}$ be the minimal polynomial of x_j in the extension $\mathbb{F}_p[x_i]/\langle Q_{\ell,i}\rangle \hookrightarrow \mathbb{K}_{\ell^n}$ (as above, it does not depend on n). This polynomial is monic, irreducible of degree ℓ^{j-i} in $\mathbb{F}_{p^{\ell^i}}[x_j] = \mathbb{F}_p[x_i]/\langle Q_{\ell,i}\rangle[x_j]$.

Thus, $\mathbb{F}_p[x_j]/\langle Q_{\ell,j}\rangle$ and $\mathbb{F}_p[x_i,x_j]/\langle Q_{\ell,i}, Q_{\ell,i,j-i}\rangle$ are two models for $\mathbb{F}_{p^{\ell^j}}$. Provided conversion algorithms between these representations are available, we can perform embeddings (that will necessarily be compatible) between different levels of the ℓ-adic tower, i.e. extensions of degrees $(\ell^i)_{i\geq 1}$.

Such towers, together with efficient conversion algorithms, were constructed in the cases $\ell = p$ in [Can89; Cou00; DS12], $\ell = 2$ in [DS15], and for other values of ℓ in [DDS13]. Thus, it remains to give algorithms to "glue" towers defined for different values of ℓ. This is the purpose of this paper.

Our contribution. The algorithms used to construct towers are inspired by those used in [Sho90; Sho94b; CL13] to build irreducible polynomials. Also used in these references is the following idea: let $Q_m(x)$ and $Q_n(y)$ be irreducible polynomials over $\mathbb{F}_p$, with coprime degrees $m,n > 1$, and having respectively $(a_i)_{1\leq i\leq m}$ and $(b_j)_{1\leq j\leq n}$ as roots in an algebraic closure of $\mathbb{F}_p$. Then their *composed product* $Q_{mn} = \prod_{1\leq i\leq m, 1\leq j\leq n}(z - a_i b_j)$ is irreducible of degree mn in $\mathbb{F}_p[z]$.

In this paper, we use an *algebraic complexity model*, where the cost of an algorithm is counted in terms of the number of operations $(+,\times,\div)$ in $\mathbb{F}_p$. If the goal is building irreducible polynomials, then computing Q_{mn} is enough: an algorithm given in [Bos+06] has quasi-linear cost in mn. Our goal here is to give algorithms for further operations: computing embeddings of the form $\varphi_x : \mathbb{F}_p[x]/\langle Q_m\rangle \to \mathbb{F}_p[z]/\langle Q_{mn}\rangle$ or $\varphi_y : \mathbb{F}_p[y]/\langle Q_n\rangle \to \mathbb{F}_p[z]/\langle Q_{mn}\rangle$, and the isomorphism $\Phi : \mathbb{F}_p[x,y]/\langle Q_m, Q_n\rangle \to \mathbb{F}_p[z]/\langle Q_{mn}\rangle$ or its inverse.

Standard solutions to these questions exist, using *modular composition* techniques: once the image $S = \Phi(x)$ is known, computing $\varphi_x(a)$ amounts to computing $a(S) \bmod Q_{mn}$; similarly, computing $\Phi(b)$, for b in $\mathbb{F}_p[x,y]/\langle Q_m, Q_n\rangle$, amounts to computing $b(S,T) \bmod Q_{mn}$, with $T = \Phi(y)$. This can be done using the Brent and Kung algorithm [BK78]: the resulting cost is $O(mn^{(\omega+1)/2}) \subset O(mn^{1.69})$ for φ_x (see the analysis in [Sho94b]) and $O((mn)^{(\omega+1)/2}) \subset O(m^{1.69}n^{1.69})$ for Φ or its inverse [PS13b]. Here, we denote by ω a constant in $(2,3]$ such that one can multiply matrices of size m over any ring A using $O(m^\omega)$ operations $(+,\times)$ in A; using the algorithms of [CW90; Vas12], we can take $\omega \leq 2.38$.

Our main result improves on these former ones. We denote by $\mathsf{M} : \mathbb{N} \to \mathbb{N}$ a function such that for any ring A, polynomials in $A[x]$ of degree at most n can be multiplied in $\mathsf{M}(n)$ operations $(+,\times)$ in A, and we make the usual super-linearity assumptions on M [GG99, Chapter 8].

Theorem B.1. *One can apply φ_x (resp. φ_y) to an element of $\mathbb{F}_p[x]/\langle Q_m\rangle$ (resp. $\mathbb{F}_p[x]/\langle Q_n\rangle$), or invert it on its image, using $O(n\mathsf{M}(m) + m\mathsf{M}(n))$ operations in $\mathbb{F}_p$.*

Suppose that $m \leq n$. Then one can apply Φ to an element of $\mathbb{F}_p[x,y]/\langle Q_m, Q_n\rangle$ or invert it using either $O(m^2\mathsf{M}(n))$ or $O(\mathsf{M}(mn)n^{1/2} + \mathsf{M}(m)n^{(\omega+1)/2})$ operations in $\mathbb{F}_p$.

Using the $O^{\sim}$ notation to neglect polylogarithmic factors, we can take $\mathsf{M}(n) \in O^{\sim}(n)$. Our algorithm for embeddings and their inverses has quasi-linear cost $O^{\sim}(mn)$. Those for Φ or Φ^{-1} have respective costs $O^{\sim}(m^2n)$ and $O^{\sim}(mn^{(\omega+1)/2})$; the minimum of the two is in $O^{\sim}((mn)^{2\omega/(\omega+1)})$; for $\omega \in (2,3]$, the resulting exponent is in $(1.333\ldots, 1.5]$.

If $S = \Phi(x)$ and $T = \Phi(y)$ are known, a result by Kedlaya and Umans [KU11] for modular composition, and its extension in [PS13a], yield an algorithm with *bit complexity* essentially linear in mn and $\log(p)$ on a RAM. Unfortunately, making these

algorithms competitive in practice is challenging; we are not aware of any implementation of them. It is also worth noting that our algorithms apply in a more general setting than finite fields (mild assumptions are required).

Outline. Section B.2 presents basic algorithms for polynomials and their transposes. Section B.3 introduces the main idea behind our algorithms: the trace induces a duality on algebras of the form $\mathbb{F}_p[x]/\langle Q\rangle$, and some conversion algorithms are straightforward in dual bases; the algorithms are detailed in Section B.4. Section B.5 explains how the results in this paper can be used in order to construct the algebraic closure of $\mathbb{F}_p$. We conclude with experimental results.

B.2 Preliminaries

We recall first previous results concerning polynomial arithmetic and transposition of algorithms. In all this section, a ground field k, not necessarily finite, is fixed. For integers m,n, we denote by $k[x]_m$ (resp. $k[x,y]_{m,n}$) the set of polynomials P in $k[x]$ with $\deg(P) < m$ (resp. P in $k[x,y]$ with $\deg(P,x) < m$ and $\deg(P,y) < n$).

Polynomial multiplication and remainder

We start with some classical algorithms and their complexity. For all the algorithms that follow, all polynomials are written on the canonical monomial basis (this is innocuous for the moment, but other bases will be discussed below).

The product of two polynomials of respective degrees at most m and n can be computed in $\mathsf{M}(\max(m,n))$ operations in k. If P is a monic polynomial of degree m in $k[x]$, for $n \geq 1$, we let $\mathsf{rem}(.,P,n)$ be the operator

$$\begin{array}{rccc} \mathsf{rem}(.,P,n): & k[x]_n & \to & k[x]_m \\ & a & \mapsto & a \bmod P. \end{array}$$

For $n \leq m$, this is free of cost. For $n > m$, this can be computed in time $O(n\mathsf{M}(m)/m)$ using the Cook-Sieveking-Kung algorithm and blocking techniques [Bos10, Ch. 5.1.3]. Defining $A = k[x]/\langle P\rangle$, and choosing a fixed $b \in A$, we can then define the mapping $\mathsf{mulmod}(.,b,P)$, which maps $a \in A$ to $ab \bmod P$; it can be computed in time $O(\mathsf{M}(m))$. Finally, given an integer m, the reversal operator in length m is

$$\begin{array}{rccc} \mathsf{rev}(.,m): & k[x]_m & \to & k[x]_m \\ & a & \mapsto & x^{m-1}a(1/x). \end{array}$$

Duality and the transposition principle

The *transposition principle* is an algorithmic result which states that, given an algorithm that performs a matrix-vector product $u \mapsto Mu$, one can deduce an algorithm with essentially the same cost which performs the transposed matrix-vector product $v \mapsto M^t v$ [BCS97b, Ch. 13].

Following [De 10], we give here a more abstract presentation of the transposition principle, using the algebraic theory of duality (see [Bou07, Ch. IX.1.8]). The added level of abstraction will pay off by greatly simplifying the proofs of the next sections.

Let E and F be k-vector spaces, with $\dim(E) = \dim(F) < \infty$, and suppose that $\langle .,. \rangle : E \times F \to k$ is a non-degenerate bilinear form. Then, to any vector space basis $\boldsymbol{\xi} = (\xi_i)_i$ of E, we can associate a unique *dual basis* $\boldsymbol{\xi}^* = (\xi_i^*)_i$ of F such that $\langle \xi_i, \xi_j^* \rangle = \delta_{i,j}$ (the

Kronecker symbol). In other words, given a in F, the coefficients (a_i) of a on the basis $\boldsymbol{\xi}^*$ are given by $a_i = \langle \xi_i, a \rangle$.

For example, denote by E^* the dual space of E, i.e. the k-linear forms on E. The bilinear form on $E \times E^*$ defined by with $\langle v, \ell \rangle = \ell(v)$ for all $v \in E$ and $\ell \in E^*$ is non-degenerate. This is indeed the canonical example, and any non-degenerate form, is isomorphic to this one. We will see in the next section another family of examples, with $E = F$.

Let E', F' be two further vector spaces, with $\dim(E') = \dim(F') < \infty$ and let $\langle ., . \rangle'$ be a bilinear form $E' \times F' \to k$. Then, to any linear mapping $u : E \to E'$, one associates its *dual* (with respect to $\langle ., . \rangle$ and $\langle ., . \rangle'$), which is a linear mapping $u^t : F' \to F$ characterized by the equality $\langle u(a), b' \rangle' = \langle a, u^t(b') \rangle$, for all $a \in E$ and $b' \in F'$.

Let as above $\boldsymbol{\xi}$ be a basis of E, and let $\boldsymbol{\xi}^*$ be the dual basis of F; consider as well a basis $\boldsymbol{\upsilon}$ of E' and its dual basis $\boldsymbol{\upsilon}^*$ of F'. If M is the matrix of u in the bases $(\boldsymbol{\xi}, \boldsymbol{\upsilon})$, the matrix of u^t in the bases $(\boldsymbol{\upsilon}^*, \boldsymbol{\xi}^*)$ is the transpose of M.

As presented in [BLS03; De 10], the transposition principle is an algorithmic technique that, given an algorithm to compute $u : E \to E'$ in the bases $(\boldsymbol{\xi}, \boldsymbol{\upsilon})$, yields an algorithm for the dual map $u^t : F' \to F$ in the bases $(\boldsymbol{\upsilon}^*, \boldsymbol{\xi}^*)$. The two algorithms have same cost, up to $O(\dim(E) + \dim(E'))$. In a nutshell, starting from an algorithm relying on a few basic operations (such as polynomial or matrix multiplication), its transpose is obtained by transposing each basic subroutine, then reversing their order.

Let us briefly review the transposes of operations described in the previous subsection. The transpose of polynomial multiplication is described in [BLS03]; it is closely related to the *middle product* [HQZ04]. Let next P be monic of degree m, and define $A = k[x]/\langle P \rangle$. As shown in [BLS03], the dual map of rem

$$\mathsf{rem}^t(., P, n) : \quad A^* \quad \to \quad k[x]_n^*$$

is equivalent to *linear sequence extension*: it takes as input the initial m values of a linear recurring sequence of minimal polynomial P, and outputs its first n values. The transposed version of the Cook-Sieveking-Kung fast Euclidean division algorithm yields an algorithm with cost $O(n\mathsf{M}(m)/m)$ operations in k [GS92; Sho99].

For a fixed $b \in A$, the transpose of mulmod is the map

$$\begin{array}{rccc} \mathsf{mulmod}^t(., b, P) : & A^* & \to & A^* \\ & \ell & \mapsto & b \cdot \ell, \end{array}$$

where $b \cdot \ell$ is defined by $(b \cdot \ell)(a) = \ell(ab)$. Algorithms for mulmod^t have been subject to much research (for instance, Berlekamp's *bit serial multiplication* [Ber82] is a popular arithmetic circuit for mulmod^t in the case $k = \mathbb{F}_2$); algorithms of cost $O(\mathsf{M}(m))$ are given in [Sho99; BLS03].

Lastly, the reversal operator on $k[x]_m$ is its own transpose.

B.3 Trace and duality

Next, we discuss some classical facts about the trace form, and give algorithms to change between monomial bases and their duals. In all this section, k is a perfect field. General references for the following are [Kun86; CLO05].

Traces in reduced algebras. Let s be a positive integer and I a zero dimensional radical ideal in $k[x_1, \ldots, x_s]$. Thus, $A = k[x_1, \ldots, x_s]/I$ is a reduced k-algebra of finite dimension d, where d is the cardinality of $V = V(I) \subset \bar{k}^s$ (in general, A is not a field).

Let a be in A. As we did in the case of one variable, we associate to a the endomorphism of multiplication-by-a $M_a : A \to A$ given by $M_a(b) = ab$. Even though A may not be a field, we still define the *minimal polynomial* of a as the minimal polynomial of M_a; since I is radical, this polynomial is squarefree, with roots $a(x)$, for x in V. Similarly, the *trace* of a is the trace of M_a, and denote it by $\tau_I(a)$. Because I is radical, the trace defines a non-degenerate bilinear form on $A \times A$, given by $\langle a, b\rangle_I = \tau_I(ab)$.

Thus, to any basis $\boldsymbol{\xi} = (\xi_i)_{0\le i<d}$ of A, one can associate a dual basis $\boldsymbol{\xi}^* = (\xi_i^*)_{0\le i<d}$, such that $\langle \xi_i, \xi_j^*\rangle_I = \delta_{i,j}$ for all i, j. It will be useful to keep in mind that for $a \in A$, its expression on the dual basis $\boldsymbol{\xi}^*$ is $a = \sum_{0\le i<d}\langle a, \xi_i\rangle_I \xi_i^*$.

We now describe algorithms for converting between the monomial basis and its dual, in two particular cases, involving respectively univariate and bivariate polynomials. In both cases, our conclusion will be that such conversions have quasi-linear complexity.

Univariate conversion. Let P be monic of degree m and squarefree in $k[x]$, and define $A = k[x]/\langle P\rangle$. We denote by P' its derivative and by τ_P the trace modulo the ideal $\langle P\rangle$.

The k-algebra A is endowed with the canonical monomial basis $\boldsymbol{\xi} = (x^i)_{0\le i<m}$. In view of what was said in the previous subsection, the coefficients of an element $a \in A$ on the dual basis $\boldsymbol{\xi}^*$ are the traces $\tau_P(ax^i)_{0\le i<m}$. The following lemma shows that the generating series of these traces is rational, with a known denominator; this will be the key to the conversion algorithm. This is a restatement of well-known results, see for instance the proof of [Rou99, Theorem 3.1].

Lemma B.2. *For a in A, the following holds in $k[[x]]$:*

$$\sum_{i\ge 0} \tau_P(ax^i)x^i = \frac{\mathsf{rev}(P'a \bmod P, m)}{\mathsf{rev}(P, m+1)}.$$

Some well-known algorithms to convert between $\boldsymbol{\xi}$ and $\boldsymbol{\xi}^*$ follow easily. In these algorithms, and all that follows, input and output are vectors (written in sans serif font).

Algorithm 1: MonomialToDual(a, P)

Input $\mathsf{a} = (a_i)_{0\le i<m} \in k^m$,
P monic squarefree in $k[x]$ of degree m
Output $(\tau_P(ax^i))_{0\le i<m}$, with $a = \sum_{0\le i<m} a_i x^i$

1. $T = 1/\mathsf{rev}(P, m+1) \bmod x^m$
2. $b = \mathsf{rev}(P' \sum_{0\le i<m} a_i x^i \bmod P, m)\, T \bmod x^m$
3. **return** $(\mathsf{coefficient}(b, x^i))_{0\le i<m}$

Algorithm 2: DualToMonomial(b, P)

Input $\mathsf{b} = (b_i)_{0\le i<m} \in k^m$,
P monic squarefree in $k[x]$ of degree m
Output $(a_i)_{0\le i<m}$ such that $\tau_P(\sum_{0\le i<m} a_i x^{i+j}) = b_j$ for all j

1. $S = 1/P' \bmod P$
2. $b = \mathsf{rev}(P, m+1) \sum_{0\le i<m} b_i x^i \bmod x^m$
3. $c = \mathsf{rev}(b, m)$
4. $d = cS \bmod P$
5. **return** $(\mathsf{coefficient}(d, x^i))_{0\le i<m}$

Lemma B.3. *Algorithms 1 and 2 are correct. The former uses $O(\mathsf{M}(m))$ operations in k, the latter $O(\mathsf{M}(m)\log(m))$. If the polynomial $S = 1/P' \bmod P$ is known, the running time of Algorithm 2 drops to $O(\mathsf{M}(m))$.*

Proof. Correctness follows from Lemma B.2. Once we point out that power series inversion modulo x^m can be done in time $O(\mathsf{M}(m))$, the running time analysis of the former is straightforward. For Algorithm 2, the dominant part is the computation of S, which takes time $O(\mathsf{M}(m)\log(m))$ by fast XGCD; all other steps take $O(\mathsf{M}(m))$ operations in k. □

Bivariate conversions. Now we consider two monic squarefree polynomials P in $k[x]$ of degree m, and Q in $k[y]$ of degree n. We define $A = k[x,y]/I$, with $I = \langle P, Q\rangle$, then A has the canonical monomial basis $(x^i y^j)_{0\le i<m, 0\le j<n}$. We denote by τ_I the trace modulo I, and by τ_P and τ_Q the traces modulo respectively $\langle P\rangle$ and $\langle Q\rangle$.

In addition to its monomial basis, A can be endowed with a total of four natural bases, which are described as follows. Let $\boldsymbol{\xi} = (x^i)_{0\le i<m}$ and $\boldsymbol{\upsilon} = (y^i)_{0\le j<n}$ be the monomial bases of respectively $k[x]/\langle P\rangle$ and $k[y]/\langle Q\rangle$; let $\boldsymbol{\xi}^*$ and $\boldsymbol{\upsilon}^*$ be their respective dual bases, with respect to τ_P and τ_Q. The monomial basis seen above is $\boldsymbol{\xi}\otimes\boldsymbol{\upsilon}$; the other combinations $\boldsymbol{\xi}^*\otimes\boldsymbol{\upsilon}$, $\boldsymbol{\xi}\otimes\boldsymbol{\upsilon}^*$ and $\boldsymbol{\xi}^*\otimes\boldsymbol{\upsilon}^*$ are bases of A as well. After a precomputation of cost $O(\mathsf{M}(m)\log(m)+\mathsf{M}(n)\log(n))$, Lemma B.3 shows that conversions between any pair of these bases can be done using $O(n\mathsf{M}(m)+m\mathsf{M}(n))$ operations in k (by applying the univariate conversion algorithms n times x-wise and / or m times y-wise). Using fast multiplication, this is quasi-linear in the dimension mn of A.

The following easy lemma will help us exhibit the duality relationships between these bases; it follows from the fact that A is the tensor product of $k[x]/\langle P\rangle$ and $k[y]/\langle Q\rangle$.

Lemma B.4. *Let b be in $k[x]/\langle P\rangle$ and c in $k[y]/\langle Q\rangle$. Then we have $\tau_I(bc) = \tau_P(b)\,\tau_Q(c)$.*

This lemma implies that $\boldsymbol{\xi}\otimes\boldsymbol{\upsilon}$ and $\boldsymbol{\xi}^*\otimes\boldsymbol{\upsilon}^*$ are dual to one another with respect to $\langle .,.\rangle_I$, as are $\boldsymbol{\xi}^*\otimes\boldsymbol{\upsilon}$ and $\boldsymbol{\xi}\otimes\boldsymbol{\upsilon}^*$.

B.4 Embedding and isomorphism

This section contains the main algorithms of this paper. We consider two squarefree polynomials $P(x)$ and $Q(y)$ of respective degrees m and n, with coefficients in a perfect field k. Let us then set $A = k[x,y]/I$, where I is the ideal $\langle P(x), Q(y)\rangle$ in $k[x,y]$. In all this section, *we assume that xy is a generator of A as a k-algebra*.

The main example we have in mind is the following: k is a finite field and both P and Q are irreducible, with $\gcd(m,n) = 1$. Then our assumption is satisfied and in addition A is a field, namely, the *compositum* of the fields $k[x]/\langle P\rangle$ and $k[y]/\langle Q\rangle$, see [BC87]. More generally, if we let $(r_i)_{i<m}$ be the roots of P in an algebraic closure of k, and let $(s_j)_{j<n}$ be the roots of Q, then as soon as the products $r_i s_j$ are pairwise distinct, xy generates A as a k-algebra.

Let $R\in k[z]$ be the minimal polynomial of xy in the extension A/k (equivalently, the roots of R are the products $r_i s_j$); this polynomial is known as the *composed product* of P and Q, and we will denote it $R = P\odot Q$. As k-algebras, we have $A\simeq k[x]/\langle R\rangle$, so there

exist embeddings φ_x, φ_y, and an isomorphism Φ of the form

$$\begin{array}{rlccc} & \varphi_x : & k[x]/\langle P\rangle & \to & k[z]/\langle R\rangle \\ & \varphi_y : & k[y]/\langle Q\rangle & \to & k[z]/\langle R\rangle \\ \text{and} & \Phi : & A = k[x,y]/\langle P,Q\rangle & \to & k[z]/\langle R\rangle \\ & & xy & \mapsto & z. \end{array}$$

In this section, we give algorithms for computing R, applying φ_x, φ_y and their sections, and finally Φ and its inverse. Except from the computation of R, these are all linear algebra problems. If R and the images $S = \Phi(x), T = \Phi(y)$ are known, then as was explained in the introduction, direct solutions are available for both φ_x (or φ_y) and Φ – modular composition – but none of these approaches have a quasi-linear running time.

We take a different path. Our algorithms have quasi-linear running time for φ_x and φ_y and improve on the Brent-Kung algorithm for Φ. Put together, Lemmas B.6 to B.10 below prove Theorem B.1. One of the key aspects of these algorithms is that some are written in the usual monomial bases, whereas others are naturally expressed in the corresponding dual bases. From the complexity point of view, this is not an issue, since we saw that all change-of-bases can be done in quasi-linear time.

In what follows, we write $\tau_P, \tau_Q, \tau_R, \tau_I$ for the traces modulo the ideals $\langle P\rangle \subset k[x]$, $\langle Q\rangle \subset k[y]$, $\langle R\rangle \subset k[z]$ and $I = \langle P,Q\rangle \subset k[x,y]$; the corresponding bilinear forms are denoted by $\langle .,.\rangle_P, \ldots$

We let $\boldsymbol{\xi} = (x^i)_{0\le i<m}$, $\boldsymbol{\upsilon} = (y^i)_{0\le j<n}$ and $\boldsymbol{\zeta} = (z^i)_{0\le i<mn}$ be the monomial bases of respectively $k[x]/\langle P\rangle$, $k[y]/\langle Q\rangle$ and $k[z]/\langle R\rangle$. We also let $\boldsymbol{\xi}^* = (\xi_i^*)_{0\le i<m}$, $\boldsymbol{\upsilon}^* = (\upsilon_i^*)_{0\le i<n}$ and $\boldsymbol{\zeta}^* = (\zeta_i^*)_{0\le i<mn}$ be the dual bases, with respect to respectively $\langle .,.\rangle_P$, $\langle .,.\rangle_Q$ and $\langle .,.\rangle_R$.

Finally, we denote by $\mathsf{u}_P \in k^m$ the vector of the coordinates of $1 \in k[x]/\langle P\rangle$ on the dual basis $\boldsymbol{\xi}^*$; the vector u_Q is defined similarly. These vectors can both be computed in quasi-linear time, since we have, for instance, $\mathsf{u}_P = \mathsf{MonomialToDual}((1,0,\ldots,0),P)$. Thus, in what follows, we assume that these vectors are known.

Embedding and computing R

We first show how to compute the embeddings φ_x and φ_y, and their inverses in quasi-linear time in mn. We actually give a slightly more general algorithm, which computes the restriction of Φ to the set

$$\Pi = \{bc \mid b \in k[x]/\langle P\rangle,\ c \in k[y]/\langle Q\rangle\} \subset k[x,y]/\langle P,Q\rangle.$$

We will use the following lemma, which results from the base independence of the trace (the second equality is Lemma B.4).

Lemma B.5. *Let b be in $k[x]/\langle P\rangle$ and c in $k[y]/\langle Q\rangle$. Then we have $\tau_R(\Phi(bc)) = \tau_I(bc) = \tau_P(b)\ \tau_Q(c)$.*

An easy consequence is that $\tau_R(z^i) = \tau_P(x^i)\tau_Q(y^i)$. From this lemma, we also immediately deduce Algorithm 3, which computes the image in $k[z]/\langle R\rangle$ of any element of Π, with inputs and outputs written on dual bases.

Algorithm 3: Embed(b, c, r)

Input $\mathsf{b} = (b_i)_{0\le i<m} \in k^m$, $\mathsf{c} = (c_i)_{0\le i<n} \in k^n$
an optional integer $r \ge mn$ set to $r = mn$ by default
Output $\mathsf{a} = (a_i)_{0\le i<r} \in k^r$

1. $(t_i)_{0\le i<r} = \mathsf{rem}^t(\mathsf{b}, P, r)$
2. $(u_i)_{0\le i<r} = \mathsf{rem}^t(\mathsf{c}, Q, r)$
3. **return** $(t_i u_i)_{0\le i<r}$

Lemma B.6. *Let $b \in k[x]/\langle P\rangle$ and $c \in k[y]/\langle Q\rangle$. Given the coefficients* b *and* c *of respectively b and c in the bases $\boldsymbol{\xi}^*$ and $\boldsymbol{\upsilon}^*$,* Embed(b, c, r) *computes $a_i = \tau_R\left(\Phi(bc)z^i\right)$ for $0 \le i < r$ in time $O(r(\mathsf{M}(m)/m + \mathsf{M}(n)/n))$. If $r = mn$, $(a_i)_{0\le i<mn}$ are the coefficients of $\Phi(bc)$ in the basis $\boldsymbol{\zeta}^*$.*

Proof. Recall that for $0 \le i < m$, $b_i = \tau_P(bx^i)$, and that for $0 \le i < n$, $c_i = \tau_Q(cy^i)$. By definition of rem^t, the sequences (t_i) and (u_i) encode the same traces, but up to index r. By Lemma B.5, the algorithm correctly computes

$$\left(\tau_P(bx^i)\tau_Q(cy^i)\right)_{i<r} = \left(\tau_R(\Phi(bc)z^i)\right)_{i<r}.$$

For $r = mn$, this is indeed the representation of $\Phi(bc)$ on the dual basis $\boldsymbol{\zeta}^*$ of $k[z]/\langle R\rangle$. The cost of the calls to rem^t is in Section B.2; the last step takes r multiplications in k. □

In particular, the map φ_x is computed as Embed$(\cdot, \mathsf{u}_Q)$, and the map φ_y as Embed$(\mathsf{u}_P, \cdot)$. Another interesting consequence is that, when A is known to be a field, Embed allows us to compute R, using the Berlekamp-Massey algorithm.

Algorithm 4: Compute$R(P, Q)$

Input P in $k[x]$, Q in $k[y]$
Output R in $k[z]$

1. $(t_i)_{0\le i<2mn} = \mathsf{Embed}(\mathsf{u}_P, \mathsf{u}_Q, 2mn)$,
2. **return** BerlekampMassey$((t_i)_{0\le i<2mn})$

Indeed, in this case, Embed$(\mathsf{u}_P, \mathsf{u}_Q, 2mn)$ computes the sequence $(\tau_R(z^i))_{0\le i<2mn}$. If we know that A is a field, R is irreducible, so the minimal polynomial of this sequence (which is computed by the Berlekamp-Massey algorithm) is precisely R; the running time is $O(\mathsf{M}(mn)\log(mn))$ operations in k. This algorithm for computing R is well-known; see for instance [Bos+06] for a variant using power series exponentials instead of Berlekamp-Massey's algorithm (that applies in large enough characteristic) and [Bos+05] for the specific case of finite fields of small characteristic.

For the inverse of say φ_x, we take a in $k[z]/\langle R\rangle$ of the form $a = \varphi_x(b)$, and compute b. Using the equality of Lemma B.5 in the form $\tau_P(bx^i) = \tau_R(az^i)/\tau_Q(y^i)$ would lead to a simple algorithm, but some traces $\tau_Q(y^i)$ may vanish.

We take a different path. Let c be a fixed element in $k[y]/\langle Q\rangle$ such that $\tau_Q(c) = 1$; we will take for c the first element υ_0^* of the dual basis of $k[y]/\langle Q\rangle$, but this is not necessary. Let us denote by $\varepsilon : k[x]/\langle P\rangle \to k[z]/\langle R\rangle$ the mapping defined by $\varepsilon(b) = \Phi(bc)$, and let

Algorithm 5: Project(a)

Input $\mathsf{a} = (a_i)_{0 \le i < mn} \in k^{mn}$
Output $\mathsf{b} = (b_i)_{0 \le i < m} \in k^m$

1. $\mathsf{c} = (1, 0, \ldots, 0)$
2. $(u_i)_{0 \le i < mn} = \mathsf{rem}^t(\mathsf{c}, Q, mn)$
3. $d = \sum_{i=0}^{mn-1} a_i u_i x^i \bmod P$
4. **return** $(\mathsf{coefficient}(d, i))_{0 \le i < m}$

$\varepsilon^t : k[z]/\langle R \rangle \to k[x]/\langle P \rangle$ be its dual map with respect to the bilinear forms $\langle ., . \rangle_P$ and $\langle ., . \rangle_R$. Then, for b and b' in $k[x]/\langle P \rangle$, we have

$$\begin{aligned} \langle b, b' \rangle_P &= \tau_P(bb') = \tau_P(bb')\tau_Q(c) = \tau_R(\Phi(bb'c)) \\ &= \langle \varepsilon(b), \Phi(b') \rangle_R = \langle b, \varepsilon^t(\Phi(b')) \rangle_P, \end{aligned}$$

where the third equality comes from Lemma B.5. Using the non-degeneracy of $\langle ., . \rangle_P$, we get $\varepsilon^t(\Phi(b')) = b'$, that is, $\varepsilon^t(\varphi_x(b')) = b'$. Thus, ε^t is an inverse of φ_x on its image.

Writing $\mathsf{c} = (1, 0, \ldots, 0)$, we remark that $\mathsf{Embed}(., \mathsf{c})$ precisely computes the mapping $b \mapsto \varepsilon(b)$. Since Embed is written in the dual bases, the discussion of Section B.2 shows that transposing this algorithm (with respect to b) yields an algorithm for ε^t written in the monomial bases.

Lemma B.7. *Let $b \in k[x]/\langle P \rangle$ and $a = \varphi_x(b)$. Given the coefficients* a *of a in the basis $\boldsymbol{\zeta} = (z^i)_{0 \le i < mn}$,* Project(a) *computes the coefficients of b in the basis $\boldsymbol{\xi} = (x^i)_{0 \le i < m}$ using $O(n\mathsf{M}(m) + n\mathsf{M}(n))$ operations in k.*

Proof. We show correctness using transposition techniques as in [BLS03]. For fixed c, Embed(b, c) is linear in b and can be written as $\pi_{\mathsf{c}} \circ \mathsf{rem}^t$, where π_{c} is the map that multiplies a vector in k^{mn} coefficient-wise by $(\tau_Q(cy^i))_{i<mn}$, for $c = \sum_{0 \le i < n} c_i \upsilon_i^*$; hence, its transpose is $\mathsf{rem} \circ \pi_{\mathsf{c}}^t$. It is evident that $\pi_{\mathsf{c}}^t = \pi_{\mathsf{c}}$ (since π_{c} is a diagonal map), whereas rem is just reduction modulo P. These correspond to steps 3 and 4. The discussion above now proves that the output is $\varepsilon^t(a)$. The cost analysis is similar to the one in Lemma B.6. □

Isomorphism

We are not able to give an algorithm for Φ that would be as efficient as those for embedding; instead, we provide two algorithms, with different domains of applicability. In what follows, without loss of generality, *we assume that $m \le n$.*

Recall that $\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}$, $\boldsymbol{\xi}^* \otimes \boldsymbol{\upsilon}$, $\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}^*$ and $\boldsymbol{\xi}^* \otimes \boldsymbol{\upsilon}^*$ are four bases of A, with $(\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}, \boldsymbol{\xi}^* \otimes \boldsymbol{\upsilon}^*)$ and $(\boldsymbol{\xi}^* \otimes \boldsymbol{\upsilon}, \boldsymbol{\xi} \otimes \boldsymbol{\upsilon}^*)$ being two pairs of dual bases with respect to $\langle ., . \rangle_I$. Our algorithms will exploit all these bases; this is harmless, since conversions between these bases have quasi-linear complexity.

Before giving the details of the algorithms, we make an observation similar to the one we did regarding the transpose of Embed. Let Φ^t be the dual map of Φ with respect to $\langle ., . \rangle_I$ and $\langle ., . \rangle_R$. Then, for any $b, b' \in k[z]/\langle R \rangle$, we have:

$$\begin{aligned} \langle b, b' \rangle_I &= \tau_I(bb') = \tau_R(\Phi(bb')) \\ &= \langle \Phi(b), \Phi(b') \rangle_R = \langle b, \Phi^t(\Phi(b')) \rangle_I; \end{aligned}$$

hence, $\Phi^t = \Phi^{-1}$. If $\mathbf{b}$ and $\mathbf{b}^*$ are two bases of $A = k[x, y]/I$, dual with respect to $\langle ., . \rangle_I$ (such as the ones seen above) and if $\mathbf{c}$ and $\mathbf{c}^*$ are two bases of $k[z]/\langle R \rangle$, dual with respect

Algorithm 6: Phi1(b)

Input $\mathsf{b} = (b_{i,j})_{0\le i<m, 0\le j<n} \in k^{m\times n}$
Output $\mathsf{a} = (a_i)_{0\le i<mn} \in k^{mn}$

1. $(u_i)_{0\le i<m(n+1)-1} = \mathsf{rem}^t(\mathsf{u}_P, P, m(n+1)-1)$
2. $(a_i)_{0\le i<mn} = (0,\dots,0)$
3. **for** $0 \le i < m$
4. $\quad (t_j)_{0\le j<mn} = \mathsf{rem}^t((b_{i,j})_{0\le j<n}, Q, mn)$
5. $\quad (a_j)_{0\le j<mn} = (a_j + t_j u_{i+j})_{0\le j<mn}$
6. **return** $(a_i)_{0\le i<mn}$

Algorithm 7: InversePhi1(a)

Input $\mathsf{a} = (a_i)_{0\le i<mn} \in k^{mn}$
Output $\mathsf{b} = (b_{i,j})_{0\le i<m, 0\le j<n} \in k^{m\times n}$

1. $(u_i)_{0\le i<m(n+1)-1} = \mathsf{rem}^t(\mathsf{u}_P, P, m(n+1)-1)$
2. **for** $i = m-1,\dots,0$
3. $\quad d = \sum_{0\le j<mn} a_j u_{i+j} y^j \bmod Q$
4. $\quad (b_{i,j})_{0\le j<n} = (\mathsf{coefficient}(d,j))_{0\le j<n}$
5. **return** $(b_{i,j})_{0\le i<m, 0\le j<n}$

to $\langle .,. \rangle_R$, the previous equality, together with the transposition principle, shows the following: if we have an algorithm for Φ, expressed in the bases $(\mathbf{b}, \mathbf{c})$, transposing it yields an algorithm for Φ^{-1}, expressed in the bases $(\mathbf{c}^*, \mathbf{b}^*)$.

First case: m is small. We start by a direct application of the results in the previous subsection, which is well-suited to situations where m is small compared to n.

Let b be in $k[x,y]/I$ and let $a = \Phi(b)$. Writing $b = \sum_{0\le i<m} b_i x^i$, with all b_i in $k[y]/\langle Q\rangle$, we obtain a straightforward algorithm to compute a: compute all $\Phi(b_i x^i)$ using Embed, then sum. Since Embed takes its inputs written on the dual bases, the algorithm requires that all b_i be written on the dual basis of $k[y]/\langle Q\rangle$ (equivalently, the input is given on the basis $\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}^*$ of A). We also use the fact that the expression of x^i on the dual basis $\boldsymbol{\xi}^*$ is u_P shifted by i positions to give a more compact algorithm, called Phi1.

Transposing this algorithm then gives an algorithm for Φ^{-1}. Its input is given on the monomial basis $(z^i)_{0\le i<mn}$ of $k[z]/\langle R\rangle$; the output is written on the basis $\boldsymbol{\xi}^* \otimes \boldsymbol{\upsilon}$ of A.

Lemma B.8. *Let $b \in k[x,y]/I$. Given the coefficients* b *of b in the basis $\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}^*$,* Phi1(b) *computes the coefficients of $\Phi(b)$ in the basis $\boldsymbol{\zeta}^*$ using $O(m^2\mathsf{M}(n))$ operations in k.*

Let $a \in k[z]/\langle R\rangle$. Given the coefficients a *of a in the basis $\boldsymbol{\zeta} = (z^i)_{0\le i<mn}$,* InversePhi1(a) *computes the coefficients of $\Phi^{-1}(a)$ in the basis $\boldsymbol{\xi} \otimes \boldsymbol{\upsilon}^*$ using $O(m^2\mathsf{M}(n))$ operations in k.*

Proof. Correctness of Phi1 follows from the previous discussion; the most expensive step is m calls to rem^t, for a cumulated cost of $O(m^2\mathsf{M}(n))$.

The correctness of the transposed algorithm is proved as in Lemma B.7, observing that it consists of the line-by-line transposition of Phi1. The running time analysis is straightforward: the dominant cost is that of m remainders, each of which costs $O(m\mathsf{M}(n))$. □

Second case: m is not small. The previous algorithms are most efficient when m is small; now, we propose an alternative solution that does better when m and n are of the same order of magnitude (with still $m \le n$).

This approach is based on baby steps / giant steps techniques, as in Brent and Kung's modular composition algorithm, but uses the fact that $z = \Phi(xy)$ to reduce the cost. Given b in $A = k[x,y]/\langle P, Q\rangle$, let us write

$$\begin{aligned} b &= \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} b_{i,j}x^i y^j = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} b_{i,j}x^i y^i y^{j-i} \\ &= \sum_{h=-m+1}^{n-1}\sum_{i=0}^{m-1} b_{i,i+h}(xy)^i y^h = \frac{1}{y^{m-1}}\sum_{h=0}^{m+n-2} c_h(xy)y^h, \end{aligned}$$

with $c_h(z) = \sum_{0\le i<m} b_{i,i+h-m+1}z^i$ for all h (undefined indices are set to zero). Hence $a = \Phi(b)$ has the form

$$a = \frac{1}{T^{m-1}}\widetilde{a} \mod R \quad \text{with} \quad \widetilde{a} = \sum_{h=0}^{m+n-2} c_h T^h,$$

where $T = \Phi(y)$. We use baby steps / giant steps techniques from [LMS13] (inspired by Brent and Kung's algorithm) to compute a, reducing the problem to polynomial matrix multiplication. Let $n' = m+n-1$, $p = \lceil\sqrt{n'}\rceil$ and $q = \lceil n'/p\rceil$, so that $n \le n' \le 2n-1$ and $p \simeq q \simeq \sqrt{n}$. For baby steps, we compute the polynomials $T_i = T^i \bmod R$, which have degree at most $mn-1$; we write $T_i = \sum_{0\le j<n} T'_{i,j} z^{jm}$, with $T'_{i,j}$ of degree less than m, and build the polynomial matrix $M_{T'}$ with entries $T'_{i,j}$. We define the matrix $M_C = [c_{iq+j}]_{0\le i<p, 0\le j<q}$ containing the polynomials c_h organized in a row-major fashion, and compute the product $M_V = M_C M_T$. We can then construct polynomials from the rows of M_V, and conclude with giant steps using Horner's scheme.

The previous discussion leads to Algorithm 8. Remark that input *and* output are written on the monomial bases.

Lemma B.9. *Let $b \in k[x,y]/I$. Given the coefficients* b *of b in the basis $\boldsymbol{\xi}\otimes\boldsymbol{\upsilon} = (x^i y^j)_{0\le i<m, 0\le j<n}$,* $\mathsf{Phi2(b)}$ *computes the coefficients of $\Phi(b)$ in the basis $\boldsymbol{\zeta} = (z^i)_{0\le i<mn}$ in $O(\mathsf{M}(mn)n^{1/2} + \mathsf{M}(m)n^{(\omega+1)/2})$ operations in k.*

Proof. Correctness follows from the discussion prior to the algorithm. As to the cost analysis, remark first that $n' = O(n)$, and that p and q are both $O(\sqrt{n})$. Steps 4 and 14 cost $O(\mathsf{M}(mn)\log(mn))$ operations. Steps 5 (the baby steps) and the loop at Step 12 (the giant steps) cost $O(\sqrt{n}\mathsf{M}(mn))$. The dominant cost is the matrix product at Step 8, which involves matrices of size $O(\sqrt{n})\times O(\sqrt{n})$ and $O(\sqrt{n})\times O(n)$, with polynomial entries of degree m: using block matrix multiplication in size $O(\sqrt{n})$, this takes $O(\mathsf{M}(m)n^{(\omega+1)/2})$ operations in k. □

As before, writing the transpose of this algorithm gives us an algorithm for Φ^{-1}, this time written in the dual bases. The process is the same for the previous transposed algorithms we saw, involving line-by-line transposition. The only point that deserves mention is Step 13, where we transpose polynomial matrix multiplication; it becomes a similar matrix product, but this time involving transposed polynomial multiplications (with degree parameters $m-1$ and m). The cost then remains the same, and leads to Lemma B.10.

Lemma B.10. *Let $a \in k[z]/\langle R\rangle$. Given the coefficients* a *of a in the basis $\boldsymbol{\zeta}^*$,* $\mathsf{InversePhi2(a)}$ *computes the coefficients of $\Phi^{-1}(a)$ in the basis $\boldsymbol{\xi}^*\otimes\boldsymbol{\upsilon}^*$ in $O(\mathsf{M}(mn)n^{1/2} + \mathsf{M}(m)n^{(\omega+1)/2})$ operations in k.*

Algorithm 8: Phi2(b)

Input $\mathsf{b} = (b_{i,j})_{0\le i<m,0\le j<n} \in k^{m\times n}$
Output $\mathsf{a} = (a_i)_{0\le i<mn} \in k^{mn}$

1. $n' = m+n-1,\ p = \lceil\sqrt{n'}\rceil,\ q = \lceil n'/p\rceil$
2. $\mathsf{y} = \mathsf{MonomialToDual}((0,1,0,\ldots,0),Q)$
3. $T = \mathsf{DualToMonomial}(\mathsf{Embed}(\mathsf{u}_P,\mathsf{y}),R)$
4. $U = 1/T \bmod R$
5. $T' = [T^i \bmod R]_{0\le i\le q}$
6. $M_{T'} = [T'_{i,j}]_{0\le i<q,0,\le j<n}$ — $T'_{i,j}$ are defined in the text
7. $M_C = [c_{iq+j}]_{0\le i<p,0\le j<q}$ — c_h are defined in the text
8. $M_V = M_C M_{T'}$
9. $V = [\sum_{0\le j<n} M_{V\,i,j} z^{jm}]_{0\le i<p}$
10. $V' = [V_i \bmod R]_{0\le i<p}$
11. $a = 0$
12. **for** $i = p-1,\ldots,0$
13. $\quad a = T'_q a + V'_i \bmod R$
14. $a = aU^{m-1} \bmod R$
15. **return** $(\mathsf{coefficient}(a,i))_{0\le i<mn}$

Algorithm 9: InversePhi2(a)

Input $\mathsf{a} = (a_i)_{0\le i<mn} \in k^{mn}$
Output $\mathsf{b} = (b_{i,j})_{0\le i<m,0\le j<n} \in k^{m\times n}$

1. $n' = m+n-1,\ p = \lceil\sqrt{n'}\rceil,\ q = \lceil n'/p\rceil$
2. $\mathsf{y} = \mathsf{MonomialToDual}((0,1,0,\ldots,0),Q)$
3. $T = \mathsf{DualToMonomial}(\mathsf{Embed}(\mathsf{u}_P,\mathsf{y}),R)$
4. $U = 1/T \bmod R$
5. $T' = [T^i \bmod R]_{0\le i\le q}$
6. $M_{T'} = [T'_{i,j}]_{0\le i<q,0,\le j<n}$ — $T'_{i,j}$ as defined above
7. $\mathsf{a} = \mathsf{mulmod}^t(\mathsf{a},U^{m-1},R)$
8. **for** $i = 0,\ldots,p-1$
9. $\quad V'_i = \mathsf{a}$
10. $\quad \mathsf{a} = \mathsf{mulmod}^t(\mathsf{a},T'_q,R)$
11. $V = [\mathsf{rem}^t(V'_i,R,mn+m-1)]_{0\le i<p}$
12. $M_V = [(V_i)_{jm,\ldots,jm+2m-2}]_{0\le i<p,0\le j<n}$
13. $M_C = \mathsf{mul}^t(M_V,M_{T'},m-1,m)$
14. $c = [M_{C0,0},\ldots,M_{C0,q-1},\ldots,M_{Cp-1,q-1}]$
15. **return** $[\mathsf{coefficient}(c_{i-j+m-1},i)]_{0\le i<m,0\le j<n}$

B.5 The algebraic closure of $\mathbb{F}_p$

In this section, we explain how the algorithms of Section B.4 can be used in order to construct and work in arbitrary extensions of $\mathbb{F}_p$, when used in conjunction with algorithms for *ℓ-adic towers* over $\mathbb{F}_p$. Space constraints prevent us from giving detailed algorithms, so we only outline the construction. We reuse definitions given in the introduction relative to ℓ-adic towers: polynomials $T_{\ell,i}$, $Q_{\ell,i}$ and $Q_{\ell,i,j-i}$ and fields $\mathbb{K}_{\ell i} = \mathbb{F}_p[x_1,\ldots,x_i]/\langle T_{\ell,1},\ldots,T_{\ell,i}\rangle$. We also assume that algorithms for embeddings or change of basis in ℓ-adic towers are available (as in [DDS13] and references therein).

Setup. For ℓ prime and $i \ge 1$, the residue class of x_i in $\mathbb{K}_{\ell i}$ will be written $x_{\ell i}$. For a positive integer $m = \ell_1^{e_1}\cdots\ell_r^{e_r}$, with ℓ_i pairwise distinct primes and e_i positive integers,

$\mathbb{K}_m$ denotes the tensor product $\mathbb{K}_{\ell_1^{e_1}} \otimes \cdots \otimes \mathbb{K}_{\ell_r^{e_r}}$; this is a field with p^m elements. If m divides n, then $\mathbb{K}_m$ embeds in $\mathbb{K}_n$. Taking the direct limit of all $\mathbb{K}_m$ under such embeddings, we get an algebraic closure $\mathbb{K}$ of $\mathbb{F}_p$. The residue classes written x_{ℓ^e} in $\mathbb{K}_{\ell^e}$ all lie in $\mathbb{K}$ and are still written x_{ℓ^e}.

For any integer m of the form $m = \ell_1^{e_1} \cdots \ell_r^{e_r}$ with ℓ_i's pairwise distinct primes, we write $x_m = x_{\ell_1^{e_1}} \cdots x_{\ell_r^{e_r}} \in \mathbb{K}$.

Minimal polynomials. We discuss first minimal polynomials of monomials in $\mathbb{K}$ over $\mathbb{F}_p$.

Take x_{ℓ^e} in $\mathbb{K}$, with ℓ prime. By construction, its minimal polynomial over $\mathbb{F}_p$ is $Q_{\ell,e}$, irreducible of degree ℓ^e in (say) $\mathbb{F}_p[z]$. Next, consider a term x_m, with $m = \ell_1^{e_1} \cdots \ell_r^{e_r}$, with ℓ_i's pairwise distinct primes. It equals $x_{\ell_1^{e_1}} \cdots x_{\ell_r^{e_r}}$, so it is a root of the composed product $Q_m = Q_{\ell_1,e_1} \odot \cdots \odot Q_{\ell_r,e_r}$. In Section B.4, we pointed out that Q_m is irreducible of degree $m = \ell_1^{e_1} \cdots \ell_r^{e_r}$ in $\mathbb{F}_p[z]$, so it must be the minimal polynomial of x_m over $\mathbb{F}_p$. In particular, this implies that $\mathbb{F}_p(x_m)$ is a field with p^m elements, and that if we consider terms x_m and x_n, with m dividing n, then x_m is in $\mathbb{F}_p(x_n)$.

Note that this process of constructing irreducible polynomials over $\mathbb{F}_p$ is already in [Sho90; Sho94b; CL13].

Embedding and change of basis. Consider a sequence $e = (e_1, \ldots, e_t)$ of positive integers, and let $n = e_1 \cdots e_t$. The set

$$B_e = \{x_{e_1}^{a_1} x_{e_1 e_2}^{a_2} \cdots x_{e_1 \cdots e_t}^{a_t} \mid 0 \le a_i < e_i \text{ for all } i\}$$

is a basis of $\mathbb{F}_p(x_n)$. Important examples are sequences of the form $e = (e_1)$, with thus $n = e_1$, for which B_e is the univariate basis $(x_n^i)_{0 \le i < n}$. Also useful for us are sequences $e = (e_1, e_2)$; letting $m = e_1$ and $n = e_1 e_2$, B_e is the bivariate basis $(x_m^i x_n^j)_{0 \le i < m, 0 \le j < n/m}$.

Consider sequences $d = (d_1, \ldots, d_s)$ and $e = (e_1, \ldots, e_t)$, with $m = d_1 \cdots d_s$ and $n = e_1 \cdots e_t$, and suppose that m divides n. The linear mapping $\mathbb{F}_p^m \to \mathbb{F}_p^n$ that describes the embedding $\mathbb{F}_{p^m} \to \mathbb{F}_{p^n}$ in the bases B_d and B_e is denoted by $\Phi_{e,d}$; when $m = n$, it is an isomorphism, with inverse $\Phi_{d,e}$. More generally, as soon as this expression makes sense, we have $\Phi_{f,d} = \Phi_{f,e} \circ \Phi_{e,d}$, so these mappings are compatible.

To conclude this section, we describe how the algorithms of this paper can be used in this framework to realize some particular cases of mappings $\Phi_{d,e}$ (more general examples can be deduced readily).

Embedding. Consider two integers m, n with m dividing n. We describe here how to embed $\mathbb{F}_p(x_m)$ in $\mathbb{F}_p(x_n)$, that is, how to compute $\Phi_{(n),(m)}$. Without loss of generality, we may assume that $n = m\ell$, with ℓ prime.

Assume first that $\gcd(m, \ell) = 1$. Since then $x_n = x_m x_\ell$, and we have access to the polynomials Q_m, Q_ℓ and Q_n (see above), we just apply the embedding algorithm of Section B.4.

Suppose now that ℓ divides m, so $m = m'\ell^k$, with m', ℓ coprime. Using one of the inverse isomorphism algorithms of Section B.4, we can rewrite an element given on the basis $(x_m^i)_{0 \le i < m}$ on the basis $(x_{m'}^i x_{\ell^k}^j)_{0 \le i < m', 0 \le j < \ell^k}$. Using an algorithm for embeddings in the ℓ-adic tower, we can then embed on the basis $(x_{m'}^i x_{\ell^{k+1}}^j)_{0 \le i < m', 0 \le j < \ell^{k+1}}$; applying our isomorphism algorithm, we end up on the basis $(x_{m\ell}^i)_{0 \le i < m\ell}$, since $x_{m\ell} = x_{m'} x_{\ell^{k+1}}$.

Further operations. Without entering into details, let us mention that further operations are feasible, in the same spirit as the embedding algorithm we just described.

For instance, for arbitrary integers m and n, it is possible to compute the relative minimal polynomial of x_{mn} over $\mathbb{F}_p(x_m)$; it is obtained as a composed product, with factors deduced from the decomposition of m and n into primes.

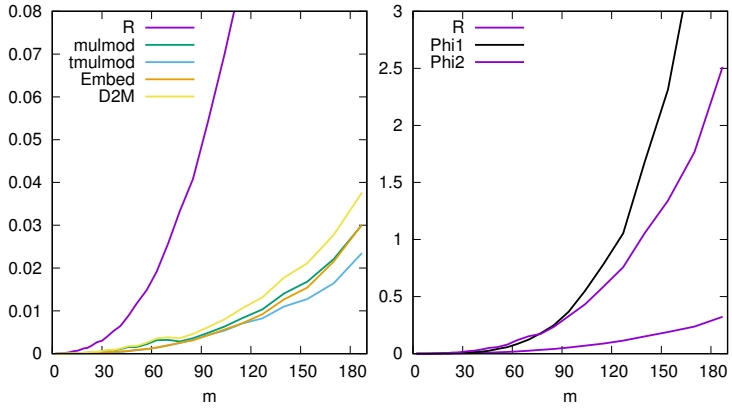


Figure B.1: Timings in seconds, $p = 5$, $n = m + 1$

As another example, we can compute $\Phi_{(m,n),(mn)}$, that is, go from the univariate basis $(x^i_{mn})_{0 \le i < mn}$ to the bivariate basis $(x^i_m x^j_{mn})_{0 \le i < m, 0 \le j < n}$. This can be used to compute for instance relative traces, norms or minimal polynomials of arbitrary elements of $\mathbb{F}_{p^{mn}}$ over $\mathbb{F}_{p^m}$.

B.6 Implementation

To demonstrate the practicality of our algorithms, we made a C implementation and compared it to various ways of constructing the same fields in Magma. All timings in this section are obtained on an Intel Xeon E5620 CPU at 2.40GHz, using Magma V2.18-12, Flint 2.4.1 and Sage 6.

Our implementation is limited to finite fields of word-sized characteristic. It is based on the C library Flint [Har10], and we make it available as a Sage module in an experimental fork at `https://github.com/defeo/sage/tree/ff_compositum`. We plan to make it available as a standard Sage module, as well as a separate C library, when the code has stabilized.

Based on the observation that algorithms Embed and Project are simpler than conversion algorithms between monomial and dual bases, we chose to implement a *lazy change of basis* strategy. By this we mean that our Sage module (rather than the C library itself) represents elements on either the monomial or the dual basis, with one representation computed from the other only when needed. For example, two elements of the same field can be summed if both have a monomial or if both have a dual representation. Similarly, two elements can be multiplied using standard multiplication if both have a monomial representation, or using transposed multiplication if one of the two has a monomial representation. In all other cases, the required representation is computed and stored when the user input prompts it. To implement this strategy efficiently, our Sage module is written in the compiled language Cython.

We focus our benchmarks on the setting of Section B.4: P and Q are two irreducible polynomials of coprime degrees m and n, and $R = P \odot Q$. We fix the base field $\mathbb{F}_p$ and make m and n grow together with $n = m + 1$. We measure the time to compute R, to apply the algorithms Embed, Phi1, etc., and to compute the changes of bases. We noticed no major difference between different characteristics, so we chose $p = 5$ for our

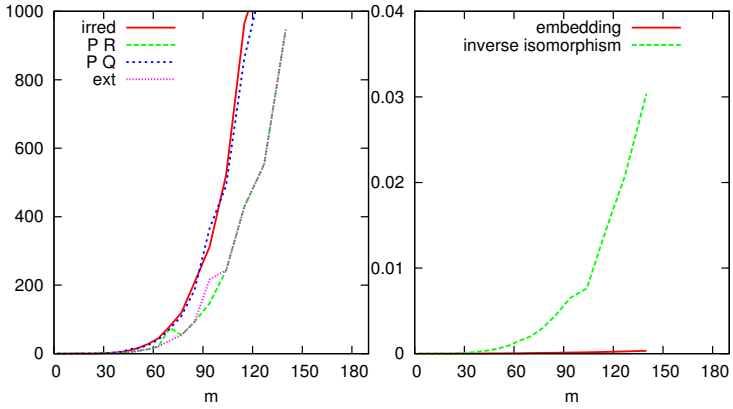


Figure B.2: Magma timings in seconds, $p = 5$, $n = m + 1$

demonstration. As shown in Figure B.1, the dominating phase is the computation of R (line labeled R). Surprisingly, transposed modular multiplication is slightly faster than ordinary modular multiplication. The cost of Embed is about the same as that of multiplication, while DualToMonomial is about 50% slower. Project and MonomialToDual have, respectively, similar performances (only slightly faster) hence they are not reported on the graph. This justifies our design choice of *lazy change of basis*.

Unsurprisingly, the isomorphism algorithms take significantly more time than the computation of R; for our choices of degrees, Phi2 is asymptotically faster than Phi1 and the crossover between them happens around $m = 70$.

We compare our implementation to four different strategies available in Magma. For each of them we measure the time to construct the finite fields and embedding data, as well as the time to do operations equivalent to Embed, resp. inverse isomorphism.

Figure B.2 reports on the following experiments. In irred, we supply directly P, Q and R to Magma's finite field constructor, then we call the `Embed` routine to compute the embedding data. In P R, we use Magma's default constructor to compute P and R (Magma chooses its own polynomials), then we call the `Embed` routine to compute the embedding. In P Q, we use Magma's default constructor to compute P and Q (Magma chooses its own polynomials), then use the `CommonOverfield` routine to compute R, then `Embed` to compute the embedding data. In ext, we use Magma's default constructor to compute P, then the `ext` operator to compute an extension of degree n of $\mathbb{F}_p[x]/\langle P\rangle$ (Magma chooses its own polynomials).

Timings for constructing the extension and the embedding vary from one method to the other; once this is done, timings for applying embeddings or (inverse) isomorphisms are the same across these methods.

The Magma implementation cannot construct the embedding data in large cases ($m = 150$) in less than 1000 seconds, while our code takes a few seconds. Once the embedding data is known, Magma can apply the embeddings or isomorphisms extremely fast; in our case, one may do the same, using our algorithms to compute the matrices of Φ and Φ^{-1}, when precomputation time and memory are not a concern.

B.7 References for "Fast arithmetic for the algebraic closure of finite fields"

[All02a] Bill Allombert. "Explicit Computation of Isomorphisms between Finite Fields". In: *Finite Fields and Their Applications* 8.3 (2002), pp. 332–342. DOI: 10.1006/ffta.2001.0344.

[BC87] Joel V. Brawley and Leonard Carlitz. "Irreducibles and the composed product for polynomials over a finite field". In: *Discrete Mathematics* 65.2 (1987), pp. 115–139. ISSN: 0012-365X. DOI: 10.1016/0012-365X(87)90135-X.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. "The MAGMA algebra system I: the user language". In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 235–265. ISSN: 0747-7171. DOI: 10.1006/jsco.1996.0125.

[BCS97a] Wieb Bosma, John Cannon, and Allan Steel. "Lattices of compatibly embedded finite fields". In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 351–369. ISSN: 0747-7171. DOI: 10.1006/jsco.1997.0138.

[BCS97b] Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic Complexity Theory*. Springer, Feb. 1997. ISBN: 3540605827.

[Ber82] Elwyn R. Berlekamp. "Bit-serial Reed–Solomon encoders". In: *IEEE Transactions on Information Theory* 28.6 (1982), pp. 869–874.

[BK78] Richard P. Brent and Hsiang Te Kung. "Fast Algorithms for Manipulating Formal Power Series". In: *Journal of the ACM* 25.4 (1978), pp. 581–595. ISSN: 0004-5411. DOI: 10.1145/322092.322099.

[BLS03] Alin Bostan, Grégoire Lecerf, and Éric Schost. "Tellegen's principle into practice". In: *ISSAC'03*. ACM, 2003, pp. 37–44. ISBN: 1-58113-641-2. DOI: 10.1145/860854.860870.

[Bos+05] Alin Bostan, Laureano González-Vega, Hervé Perdry, and Éric Schost. "From Newton sums to coefficients: complexity issues in characteristic p". In: *MEGA'05*. 2005.

[Bos+06] Alin Bostan, Philippe Flajolet, Bruno Salvy, and Éric Schost. "Fast computation of special resultants". In: *Journal of Symbolic Computation* 41.1 (2006), pp. 1–29. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2005.07.001.

[Bos10] Alin Bostan. *Algorithmes rapides pour les polynômes, séries formelles et matrices*. Vol. 1. Les cours du CIRM 2. 2010. URL: https://hal.inria.fr/hal-00780433/.

[Bou07] Nicolas Bourbaki. *Éléments de mathématique*. Algèbre. Chapitre 9. Springer, 2007.

[Can89] David G. Cantor. "On arithmetical algorithms over finite fields". In: *Journal of Combinatiorial Theory*. Series A 50.2 (1989), pp. 285–300. ISSN: 0097-3165. DOI: 10.1016/0097-3165(89)90020-4.

[CL13] Jean-Marc Couveignes and Reynald Lercier. "Fast construction of irreducible polynomials over finite fields". In: *Israel Journal of Mathematics* 194.1 (2013), pp. 77–105.

[CLO05] David A. Cox, John Little, and Donal O'Shea. *Using Algebraic Geometry*. Springer-Verlag, 2005. ISBN: 0387207066.

[Cou00] Jean-Marc Couveignes. “Isomorphisms between Artin-Schreier towers”. In: *Mathematics of Computation* 69.232 (2000), pp. 1625–1631. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-00-01193-5.

[CW90] Don Coppersmith and Shmuel Winograd. “Matrix multiplication via arithmetic progressions”. In: *Journal of Symbolic Computation* 9.3 (1990), pp. 251–280. ISSN: 07477171. DOI: 10.1016/S0747-7171(08)80013-2.

[DDS13] Luca De Feo, Javad Doliskani, and Éric Schost. “Fast Algorithms for ℓ-adic Towers over Finite Fields”. In: *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*. ISSAC ’13. New York, NY, USA: ACM, 2013, pp. 165–172. ISBN: 978-1-4503-2059-7. DOI: 10.1145/2465506.2465956.

[De 10] Luca De Feo. “Algorithmes Rapides pour les Tours de Corps Finis et les Isogénies”. PhD thesis. Ecole Polytechnique X, Dec. 2010. URL: http://tel.archives-ouvertes.fr/tel-00547034/en/.

[DS12] Luca De Feo and Éric Schost. “Fast arithmetics in Artin-Schreier towers over finite fields”. In: *Journal of Symbolic Computation* 47.7 (2012), pp. 771–792. ISSN: 07477171. DOI: 10.1016/j.jsc.2011.12.008.

[DS15] Javad Doliskani and Éric Schost. “Computing in degree 2^k-extensions of finite fields of odd characteristic”. In: *Designs, Codes and Cryptography* 74.3 (2015), pp. 559–569.

[GG99] Joachim von zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. New York, NY, USA: Cambridge University Press, 1999. ISBN: 0-521-64176-4.

[GS92] Joachim von zur Gathen and Victor Shoup. “Computing Frobenius Maps and Factoring Polynomials”. In: *Computational Complexity* 2 (1992), pp. 187–224.

[Har10] William B. Hart. “Fast Library for Number Theory: An Introduction”. In: *Proceedings of the Third International Congress on Mathematical Software*. ICMS’10. Kobe, Japan: Springer-Verlag, 2010, pp. 88–91. URL: http://flintlib.org/.

[HQZ04] Guillaume Hanrot, Michel Quercia, and Paul Zimmermann. “The Middle Product Algorithm I”. In: *Applicable Algebra in Engineering, Communication and Computing* 14.6 (2004), pp. 415–438. ISSN: 0938-1279. DOI: 10.1007/s00200-003-0144-2.

[KU11] Kiran S Kedlaya and Christopher Umans. “Fast polynomial factorization and modular composition”. In: *SIAM Journal on Computing* 40.6 (2011), pp. 1767–1802.

[Kun86] Ernst Kunz. *Kähler differentials*. Friedrich Vieweg & Sohn, 1986.

[Len91] Hendrik W. Lenstra. “Finding isomorphisms between finite fields”. In: *Mathematics of Computation* 56.193 (1991), pp. 329–347.

[LMS13] Romain Lebreton, Esmaeil Mehrabi, and Éric Schost. “On the Complexity of Solving Bivariate Systems: The Case of Non-singular Solutions”. In: *ISSAC’13*. ACM, 2013, pp. 251–258.

[LS08a] Hendrik W. Lenstra and Bart de Smit. *Standard models for finite fields: the definition*. 2008. URL: http://www.math.leidenuniv.nl/~desmit/papers/standard_models.pdf.

[NTL] Victor Shoup. *NTL: A library for doing number theory*. URL: http://www.shoup.net/ntl.

[PARI] *PARI/GP, version 2.8.0*. The PARI Group. Bordeaux, 2016. URL: https://pari.math.u-bordeaux.fr/.

[PS13a] Adrian Poteaux and Éric Schost. "Modular Composition Modulo Triangular Sets and Applications". In: *Computational Complexity* 22.3 (2013), pp. 463–516.

[PS13b] Adrien Poteaux and Éric Schost. "On the complexity of computing with zero-dimensional triangular sets". In: *Journal of Symbolic Computation* 50 (2013), pp. 110–138.

[Rou99] Fabrice Rouillier. "Solving Zero-Dimensional Systems Through the Rational Univariate Representation". In: *Applicable Algebra in Engineering, Communication and Computing* 9.5 (May 1999), pp. 433–461. ISSN: 0938-1279. DOI: 10.1007/s002000050114.

[Sage] *SageMath, the Sage Mathematics Software System (Version 8.0)*. The Sage Developers. 2018. URL: https://www.sagemath.org.

[Sho90] Victor Shoup. "New algorithms for finding irreducible polynomials over finite fields". In: *Mathematics of Computation* 54.189 (Jan. 1990), pp. 435–435. DOI: 10.1090/s0025-5718-1990-0993933-0.

[Sho94b] Victor Shoup. "Fast construction of irreducible polynomials over finite fields". In: *Journal of Symbolic Computation* 17.5 (1994), pp. 371–391. ISSN: 0747-7171. DOI: 10.1006/jsco.1994.1025.

[Sho99] Victor Shoup. "Efficient Computation of Minimal Polynomials in Algebraic Extensions of Finite Fields". In: *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*. ISSAC '99. New York, NY, USA: ACM, 1999, pp. 53–58. ISBN: 1-58113-073-2. DOI: 10.1145/309831.309859.

[Vas12] Virginia Vassilevska Williams. "Multiplying matrices faster than Coppersmith-Winograd". In: *STOC'12*. ACM, 2012, pp. 887–898.

C Computing isomorphisms and embeddings of finite fields

Abstract

Let $\mathbb{F}_q$ be a finite field. Given two irreducible polynomials f, g over $\mathbb{F}_q$, with $\deg f$ dividing $\deg g$, the finite field embedding problem asks to compute an explicit description of a field embedding of $\mathbb{F}_q[X]/f(X)$ into $\mathbb{F}_q[Y]/g(Y)$. When $\deg f = \deg g$, this is also known as the isomorphism problem.

This problem, a special instance of polynomial factorization, plays a central role in computer algebra software. We review previous algorithms, due to Lenstra, Allombert, Rains, and Narayanan, and propose improvements and generalizations. Our detailed complexity analysis shows that our newly proposed variants are at least as efficient as previously known algorithms, and in many cases significantly better.

We also implement most of the presented algorithms, compare them with the state of the art computer algebra software, and make the code available as open source. Our experiments show that our new variants consistently outperform available software.

C.1 Introduction

Let q be a prime power and let $\mathbb{F}_q$ be a field with q elements. Let f and g be irreducible polynomials over $\mathbb{F}_q$, with $\deg f$ dividing $\deg g$. Define $k = \mathbb{F}_q[X]/f(X)$ and $K = \mathbb{F}_q[Y]/g(Y)$; then, there is an embedding $\phi : k \hookrightarrow K$, unique up to $\mathbb{F}_q$-automorphisms of k. The goal of this paper is to describe algorithms to efficiently represent and evaluate one such embedding.

All the algorithms we are aware of split the embedding problem in two sub-problems:

1. Determine elements $\alpha \in k$ and $\beta \in K$ such that $k = \mathbb{F}_q(\alpha)$, and such that there exists an embedding ϕ mapping α to β. We refer to this problem as the *embedding description problem*. It is easily seen that (α, β) describes an embedding if and only if α and β share the same minimal polynomial.

2. Given elements α and β as above, given $\gamma \in k$ and $\delta \in K$, solve the following problems:

   - Compute $\phi(\gamma) \in K$.
   - Test if $\delta \in \phi(k)$.
   - If $\delta \in \phi(k)$, compute $\phi^{-1}(\delta) \in k$.

   We refer collectively to these problems as the *embedding evaluation problem*.

Motivation, previous work. The first to get interested in this problem was H. Lenstra: in his seminal paper [Len91] he shows that it can be solved in deterministic polynomial time, by using a representation for finite fields that he calls *explicit data*.[1] In practice, the embedding problem arises naturally when designing a computer algebra system: as soon as a system is capable of representing arbitrary finite fields, it is natural to ask it to compute the morphisms between them. Ultimately, by representing effectively the lattice of finite fields with inclusions, the user is given access to the

[1] Technically, Lenstra only proved his theorem in the case where k and K are isomorphic; however, the generalization to the embedding problem poses no difficulties.

algebraic closure of $\mathbb{F}_q$. The first system to implement a general embedding algorithm was Magma [BCP97]. As detailed by its developers [BCS97a], it used a much simpler approach than Lenstra's algorithm, entirely based on polynomial factorization and linear algebra. Lenstra's algorithm was later revived by Allombert [All02a; All02b] who modified some key steps in order to make it practical; his implementation has since been part of the PARI/GP system [PARI].

Meanwhile, a distinct family of algorithms for the embedding problem was started by Pinch [Pin92], and later improved by Rains [Rai96]. These algorithms, based on principles radically different from Lenstra's, are intrinsically probabilistic. Although their worst-case complexity is no better than that of Allombert's algorithm, they are potentially much more efficient on a large set of parameters. This potential was understood by Magma's developers, who implemented Rains' algorithm in Magma v2.14.[2]

With the exception of Lenstra's work, the aforementioned papers were mostly concerned with the practical aspects of the embedding problem. While it was generally understood that computing embeddings is an easier problem than general polynomial factoring, no results on its complexity more precise than Lenstra's had appeared until recently. A few months before the present paper was finalized, Narayanan published a novel generalization of Allombert's algorithm [Nar18], based on elliptic curve computations, and showed that its (randomized) complexity is at most quadratic. Narayanan's generalization relies on the fact that Artin–Schreier and Kummer theories are special cases of a more general situation: as already emphasized by Couveignes and Lercier [CL08], whereas the former theory acts on the additive group of a finite field, and the latter on its multiplicative group, they can be extended to more general commutative algebraic groups, in particular to elliptic curves.

Our contribution. This work aims to be, in large part, a complete review of all known algorithms for the embedding problem; we analyze in detail the cost of existing algorithms and introduce several new variants. After laying out the foundations in the next section, we start with algorithms for the embedding description problem.

Section C.3 describes the family of algorithms based (more or less loosely) on Lenstra's work; we call these *Kummer-type* algorithms. In doing so, we pay a particular attention to Allombert's algorithm: to our knowledge, this is the first detailed and complete complexity analysis of this algorithm and its variants. Thanks to our work on asymptotic complexity, we were able to devise improvements to the original variants of Allombert that largely outperform them both in theory and practice. One notable omission in this section is Narayanan's algorithm, which is, in our opinion, mostly of theoretical rather than practical interest. We present instead in Subsection C.3 a simpler algorithm with essentially the same complexity.

In Section C.4 we describe Rains' algorithm. Rains' original preprint [Rai96] went unpublished, thus we give here a complete description and analysis of his algorithm, for reference. We also give new variants of Rains' algorithm of lesser interest in Appendix C.8.

Then, in Section C.5 we present a generalization of Rains' algorithm using elliptic curves. The possibility of this algorithm was hinted at by Rains, but never fully developed; we show that it is indeed possible to use *elliptic periods* to solve the embedding description problem, and that the resulting algorithm behaves well both in theory and

[2] As a matter of fact, Rains' algorithm was never published; the only publicly available source for it is in Magma's source code (file `package/Ring/FldFin/embed.m`, since v2.14).

in practice. While working out the correctness proof of the elliptic variant of Rains' algorithm, we encounter an unexpected difficulty: whereas roots of unity enjoy Galois properties that guarantee the success of Rains' original algorithm, points of elliptic curves fail to provide the same. Heuristically, the failure probability of the elliptic variant is extremely small, however we are not able to prove it formally. Our experimental searches even seem to suggest that the failure probability might be, surprisingly, zero. We state this as a conjecture on elliptic periods (see Conjecture C.22).

Section C.6 does a global comparison of all the algorithms presented previously. In particular, Rains' algorithm and variants require a non-trivial search for parameters, which we discuss thoroughly. Then we present an algorithm to select the best performing embedding description algorithm from a practical point of view. This theoretical study is complemented by the experimental Section C.7, where we compare our implementations of all the algorithms; our source code is made available through the Git repository `https://github.com/defeo/ffisom` for replication and further scrutiny.

The algorithms for the embedding evaluation problem are much more classical and well understood. Due to space constraints, we do not present them here; we address instead the interested reader to the extended version of the present paper [Bri+17].

In conclusion, we hope that our review will constitute a reference guide for researchers and engineers interested in implementing embeddings of finite fields in a computer algebra system.

C.2 Preliminaries

Fundamental algorithms and complexity

We review the fundamental building blocks that constitute the algorithms presented next. We are going to measure all complexities in number of operations $+$, $\times$, $\div$ in $\mathbb{F}_q$, unless explicitly stated otherwise. Most of the algorithms we present are randomized; we use the *big-Oh* notation $O(\,)$ to express *average* asymptotic complexity, and we will make it clear when this complexity depends on heuristics. We also occasionally use the notation $\tilde{O}(\,)$ to neglect logarithmic factors in the parameters.

We let $\mathsf{M}(m)$ be a function such that polynomials in $\mathbb{F}_q[X]$ of degree less than m can be multiplied in $\mathsf{M}(m)$ operations in $\mathbb{F}_q$, under the assumptions of [GG99, Ch. 8.3], together with the slightly stronger one, that $\mathsf{M}(mn)$ is in $O(m^{1+\varepsilon}\mathsf{M}(n))$ for all $\varepsilon > 0$. Using FFT multiplication, one can take $\mathsf{M}(m) \in O(m\log(m)\operatorname{loglog}(m))$ [CK91].

We denote by ω the *exponent of linear algebra*, i.e. a constant such that $m \times m$ matrices with coefficients in any field k can be multiplied using $O(m^\omega)$ additions and multiplications in k. One can take $\omega < 2.38$, the best result to date being in [Le 14]; on the other hand, we also suppose that $\omega > 2$.

The algorithms presented in the next sections perform computations in ring extensions of finite fields. Some of these extensions also happen to be finite fields. As customary, if k is a finite field and ξ is some element of an algebraic extension of k, we will write $k[\xi]$ for the ring generated by ξ. To avoid confusion, when the extension generated by ξ is a finite field, we will write instead $k(\xi)$.

Some algorithms will operate in a polynomial ring $k[Z]$, where k is a field extension of $\mathbb{F}_q$; some other algorithms will operate in $k[Z]/h(Z)$, where h is a monic polynomial in $k[Z]$. We review the basic operations in these rings. We assume that k is represented as a quotient ring $\mathbb{F}_q[X]/f(X)$, with $m = \deg f$, and we let $s = \deg h$ in the complexity estimates.

Multiplying and dividing polynomials of degree at most s in $k[Z]$ is done in $O(\mathsf{M}(sm))$ operations in $\mathbb{F}_q$, using Kronecker's substitution [Moe76; Kal87; GG99; GS92; Har09]. Multiplication in $k[Z]/h(Z)$ is also done in $O(\mathsf{M}(sm))$ operations using the technique in [PS06]. By the same techniques, gcds of degree m polynomials in $k[Z]$ and inverses in $k[Z]/h(Z)$ are computed in $O(\mathsf{M}(sm)\log(sm))$ operations.

Given polynomials $e,g,h \in k[Z]$ of degree at most s, modular composition is the problem of computing $e(g) \bmod h$. An upper bound on the algebraic complexity of modular composition is obtained by the Brent–Kung algorithm [BK78]; under our assumptions on the respective costs of polynomial and matrix multiplication, its cost is $O(s^{(\omega+1)/2}\mathsf{M}(m))$ operations in $\mathbb{F}_q$ (so if $k=\mathbb{F}_q$, this is $O(s^{(\omega+1)/2})$). In the binary RAM complexity model, the Kedlaya–Umans algorithm [KU11] and its extension in [PS13a] yield an algorithm with essentially linear complexity in s, m and $\log(q)$. Unfortunately, making these algorithms competitive in practice is challenging; we are not aware of any implementation of them that would outperform Brent and Kung's algorithm.

Note: If we have several modular compositions of the form $e_1(g) \bmod h, \dots, e_t(g) \bmod h$ to compute, we can slightly improve the obvious bound $O(ts^{(\omega+1)/2})$ (we discuss here $k=\mathbb{F}_q$, so $m=1$). If $t=O(s)$, using [KS98, Lemma 4], this can be done in time $O(t^{(\omega-1)/2}s^{(\omega+1)/2})$. If $t=\Omega(s)$, this can be done in $O(ts^{\omega-1})$ operations, by computing $1,g,\dots,g^{s-1}$ modulo f, and doing a matrix product in size $s\times s$ by $s\times t$.

Frobenius evaluation. Consider an $\mathbb{F}_q$-algebra Q, and an element α in Q. Given integers c,d, we will have to compute expressions of the form

$$\sigma_d = \alpha^{q^d}, \quad \tau_d = \sum_{i=0}^{d-1} \alpha^{q^{ci}}, \quad \mu_d = \alpha^{\lfloor q^d/c \rfloor} \ .$$

A direct binary powering approach would yield a complexity of, e.g., $O(d\log(q))$ multiplications in Q for the first expression.

To do better, we use a recursive approach that goes back to [GS92], with further ideas borrowed from [Sho94b; KS97]. For $i \geq 1$, define integers A_i, B_i as follows

$$q^i = A_i c + B_i, \quad 0 \leq B_i < c.$$

Then, we have the relations

$$\sigma_{i+j} = \sigma_j^{q^i}, \quad \tau_{i+j} = \tau_i + \tau_j^{q^{ic}}, \quad \mu_{i+j} = \mu_j^{q^i}\mu_i^{B_j}\alpha^{\lfloor B_i B_j/c \rfloor}.$$

Since we are interested in σ_d, τ_d and μ_d, using an addition chain for d, we are left to perform $O(\log(d))$ steps as above.

To perform these operations, we will make a heavy use of a technique originating in [GS92]. In its simplest form, it amounts to the following: if $Q=\mathbb{F}_q[X]/f(X)$, for some polynomial f in $\mathbb{F}_q[X]$, and β is in Q, we can compute β^q by means of the modular composition $\beta(\xi)$, where $\xi = x^q$ and x is the image of X modulo f.

In the following proposition, we discuss versions of this idea for various kinds of algebras Q, and how they allow us to compute the expressions σ_d, τ_d, μ_d defined above.

Proposition C.1. *Let $f \in \mathbb{F}_q[X]$ be a polynomial of degree m, and define the $\mathbb{F}_q$-algebra $Q=\mathbb{F}_q[X]/f(X)$. Let $h \in Q[Z]$ be a polynomial of degree s, and define the Q-algebra $S=Q[Z]/h(Z)$. Finally, whenever $h \in \mathbb{F}_q[Z]$, define the $\mathbb{F}_q$-algebra $Q'=\mathbb{F}_q[Z]/h(Z)$.*

Denote by $T_Q, T_S, T_{Q'}$ the cost, in terms of $\mathbb{F}_q$-operations, of one modular composition in Q, S, Q' respectively. Also denote by $T_{Q,t} \leq tT_Q$ (resp. $T_{S,t}, T_{Q',t}$) the cost of t modular compositions sharing the same polynomial (see Note C.2).

Then the expressions

$$\sigma_d = \alpha^{q^d}, \quad \tau_d = \sum_{i=0}^{d-1} \alpha^{q^{ci}}, \quad \mu_d = \alpha^{\lfloor q^d/c \rfloor}$$

can be computed using the following number of operations:

Case 1. $\alpha \in Q$*:*

- σ_d*:* $O(\mathsf{M}(m)\log(q) + T_Q \log(d))$,
- τ_d*:* $O(\mathsf{M}(m)\log(q) + T_Q \log(dc))$,
- μ_d*:* $O(\mathsf{M}(m)\log(q) + (T_Q + \mathsf{M}(m)\log(c))\log(d))$;

Case 2. $\alpha \in Q$ *with* $f | X^r - 1$*:*

- σ_d*:* $O(\mathsf{M}(m)\log(q) + \mathsf{M}(r)\log(d))$,
- τ_d*:* $O(\mathsf{M}(m)\log(q) + \mathsf{M}(r)\log(dc))$,
- μ_d*:* $O(\mathsf{M}(m)\log(q) + (\mathsf{M}(r) + \mathsf{M}(m)\log(c))\log(d))$;

Case 3. $\alpha \in S$*:*

- σ_d*:* $O(\mathsf{M}(ms)\log(q) + (T_{Q,s} + T_S)\log(d))$,
- τ_d*:* $O(\mathsf{M}(ms)\log(q) + (T_{Q,s} + T_S)\log(dc))$,
- μ_d*:* $O(\mathsf{M}(ms)\log(q) + (T_{Q,s} + T_S + \mathsf{M}(ms)\log(c))\log(d))$;

Case 4. $\alpha \in S$ *with* $h \in \mathbb{F}_q[Z]$*:*

- σ_d*:* $O((\mathsf{M}(m) + \mathsf{M}(s))\log(q) + (T_{Q,s} + T_{Q',m})\log(d)$,
- τ_d*:* $O((\mathsf{M}(m) + \mathsf{M}(s))\log(q) + (T_{Q,s} + T_{Q',m})\log(dc))$,
- μ_d*:* $O((\mathsf{M}(m) + \mathsf{M}(s))\log(q) + (T_{Q,s} + T_{Q',m} + (m\mathsf{M}(s) + s\mathsf{M}(m))\log(c))\log(d))$;

Case 5. $\alpha \in S$ *with* $h | X^r - a$ *for* $a \in Q$*:*

- σ_d*:* $O(\mathsf{M}(m)\log(q) + (T_{Q,s} + \mathsf{M}(mr))\log(d))$,
- τ_d*:* $O(\mathsf{M}(m)\log(q) + (T_{Q,s} + \mathsf{M}(mr))\log(dc))$,
- μ_d*:* $O(\mathsf{M}(m)\log(q) + (T_{Q,s} + \mathsf{M}(mr) + \mathsf{M}(m)\log(c))\log(d))$.

Proof. The complexity estimates mostly rely on the complexity of modular composition.

Case 1. We let x be the image of X in Q, and we start by computing x^q, using $O(\mathsf{M}(m)\log(q))$ operations in $\mathbb{F}_q$.

For $i \geq 0$, given $\xi_i = x^{q^i}$ and β in Q, we can compute β^{q^i} as $\beta^{q^i} = \beta(\xi_i)$, using $T_Q = O(m^{(\omega+1)/2})$ operations; in particular, this allows us to compute ξ_{i+j} from the knowledge of ξ_i and ξ_j. Given an addition chain for d, we thus compute all corresponding ξ_i's, and we deduce the σ_i's similarly, since $\sigma_{i+j} = \sigma_j(\xi_j)$. Altogether, starting from $\xi_1 = x^q$, this gives us σ_d for $O(T_Q \log(d))$ further operations in $\mathbb{F}_q$.

The same holds for τ_d, with a cost in $O(T_Q \log(cd))$, since we have to compute ξ_c first; and for μ_d, with a cost in $O((T_Q + \mathsf{M}(m)\log(c))\log(d))$ operations, as the formula for μ_{i+j} shows that we can obtain it by means of a modular composition (to compute $\mu_j^{q^i} = \mu_j(\xi_i)$), together with two exponentiations of indices less than c.

The costs for computing σ_d, τ_d, μ_d follow immediately.

Case 2. When f divides $X^r - 1$, we obtain $\beta(\xi_i)$ by computing $\beta(X^{q^i \bmod r}) \bmod (X^r - 1)$ first, and then reducing modulo $f(X)$. Thus, the cost of one modular composition is $T_Q = O(\mathsf{M}(r))$, and the total cost is obtained by replacing this value in the estimates for the previous case.

Case 3. We let x and z be the respective images of X in Q and Z in S, and as a first step, we compute z^q (and x^q, unless f is as in Case 2 above), in $O(\mathsf{M}(ms)\log(q))$ operations.

In order to compute the quantities σ_d, τ_d, μ_d, we apply the same strategy as above; the key factor for complexity is thus the cost of computing β^{q^i}, for β in S, given $\zeta_i = z^{q^i}$ and $\xi_i = x^{q^i}$ (as we did in Case 1, we apply this procedure to our input element α, as well as to ζ_i itself, and ξ_i, in order to be able to continue the calculation).

To do so, we use an algorithm by Kaltofen and Shoup [KS97], which boils down to writing $\beta = \sum_{j=0}^{s-1} c_j(x) z^j$, so that $\beta^{q^i} = \sum_{j=0}^{s-1} c_j(x)^{q^i} \zeta_i^j$. The s coefficients $c_j(x)^{q^i}$ are computed by applying the previous algorithms in Q to s inputs. This takes time at most sT_Q, but as pointed out in Note C.2, improvements are possible if we base our algorithm on modular composition; we thus denote the cost $T_{Q,s}$.

Then, we do a modular composition in S to evaluate the result at ζ_i; this latter step takes $T_S = O(s^{(\omega+1)/2}\mathsf{M}(m))$ operations in $\mathbb{F}_q$.

Case 4. The cost for computing z^q is $O(\mathsf{M}(s)\log(q))$ and that for computing x^q is $O(\mathsf{M}(m)\log(q))$. In the last step, the cost T_S of modular composition in S is now that of m modular compositions in degree s (with the same argument), as detailed in Note C.2, that we denote $T_{Q',m}$. Similarly, the cost of multiplication in S can be reduced from $O(\mathsf{M}(ms))$ to $O(s\mathsf{M}(m) + m\mathsf{M}(s))$ operations.

Case 5. We start by computing x^q, using $O(\mathsf{M}(m)\log(q))$ operations in $\mathbb{F}_q$.

For β as above, suppose that we have already computed all coefficients $d_j(x) = c_j(x)^{q^i}$ in $O(T_{Q,s})$ operations; we now have to compute $\beta^{q^i} = \sum_{j=0}^{s-1} d_j(x)\zeta_i^j$.

We first do the calculation modulo $Z^r - a$ rather than modulo h; that is, we compute $\sum_{j=0}^{s-1} d_j(x) z_i^j$ where $z_i = z^{q^i}$. Because $z^r = a$, we have $z_i = a_i z^{q^i \bmod r}$, with $a_i = a^{\lfloor q^i/r \rfloor}$. If we assume that a_i is known, we can compute $\sum_{j=0}^{s-1} d_j(x) z_i^j$ using Horner's method, in time $O(s\mathsf{M}(m))$, and we reduce this result modulo h, for the cost $O(\mathsf{M}(mr))$ of a Euclidean division in degree r in $Q[Z]$.

In order to continue the calculation for all indices in our addition chain, we must thus compute the corresponding a_i's as well, just like the μ_i's; this takes $O(T_Q + \mathsf{M}(m)\log(r))$ operations.

Since the first stage of the algorithm took $O(T_{Q,s})$ operations, we can take $T_S = O(\mathsf{M}(mr))$ for computing β^{q^i}.

To initiate the procedure, the algorithm also needs to compute $a_1 = a^{\lfloor q/r \rfloor}$, using $O(\log(q))$ multiplications in Q for a cost $O(\mathsf{M}(m)\log(q))$. □

Computing subfields. With $k = \mathbb{F}_q[X]/f(X)$ and $\deg f = m$ as above, we are given a divisor r of m, and we want to construct an intermediate extension $\mathbb{F}_q \subset L \subset k$ of degree r over $\mathbb{F}_q$. More precisely, we want to compute a monic irreducible polynomial $g \in \mathbb{F}_q[X]$ of degree r, and a polynomial $h \in \mathbb{F}_q[X]$ such that $x \mapsto h(x) \bmod f$ defines an embedding $L = \mathbb{F}_q[X]/g(X) \hookrightarrow k$.

An algorithm for this will be used to reduce the embedding problem to an isomorphism problem; although this is not strictly necessary for most of our algorithms to work, it will greatly simplify the expostion. We proceed as follows.

Let $\alpha \in k$ be a random element[3]. Then α has a minimal polynomial of degree m over $\mathbb{F}_q$ with high probability. In other words, one needs $O(1)$ such random elements to find one with degree m minimal polynomial. Now, the trace

$$\mathrm{Tr}_{k/L}(\alpha) = \alpha + \alpha^{q^r} + \cdots + \alpha^{q^{m-r}} \tag{C.1}$$

has a minimal polynomial of degree r over $\mathbb{F}_q$ with high probability as well. This means we can compute, after $O(1)$ random trials, the desired polynomials $\beta = \mathrm{Tr}_{k/L}(\alpha)$, its minimal polynomial g, and h the polynomial of degree less than m representing β.

Proposition C.2. *Let $\mathbb{F}_q \subset k$ be a finite extension of degree m, and let r be a divisor of m. Computing an intermediate field $\mathbb{F}_q \subset L \subset k$ with $[L : \mathbb{F}_q] = r$ takes an expected $O(m^{(\omega+1)/2}\log(m) + \mathsf{M}(m)\log(q))$ operations in $\mathbb{F}_q$. Once L is computed, any element $\gamma \in L$ can be lifted to its image in k using $O(m^{(\omega+1)/2})$ operations.*

Proof. Computing the minimal polynomial of an element in k takes $O(m^{(\omega+1)/2})$ operations in $\mathbb{F}_q$, see [Sho99]. The trace in Eq. (C.1) is computed as the expression τ_m of the previous paragraph (with $c = r$ and $d = m/r$), at a cost of $O(m^{(\omega+1)/2}\log(m) + \mathsf{M}(m)\log(q))$ operations in $\mathbb{F}_q$.

Finally, given an element $\gamma \in L$, its image in k is computed by evaluating $h(\gamma)$, where h is the polynomial representation of $\mathrm{Tr}_{k/L}(\alpha)$. This can be done by a modular composition at cost $O(m^{(\omega+1)/2})$. □

Root finding in cyclotomic extensions. Given a field $k = \mathbb{F}_q[X]/f(X)$ of degree m as above, we will need to factor some special polynomials in $k[Z]$: we are interested in finding one factor of a polynomial that splits into factors of the same, known, degree. This problem is known as *equal degree factorization* (EDF), and the best generic algorithm for it is the Cantor–Zassenhaus method [CZ81; GS92], which runs in $O(\mathsf{M}(sm)(dm\log(q) + \log(sm)))$ operations in $\mathbb{F}_q$ [GG99, Th. 14.9], where s is the degree of the polynomial to factor, and d is the degree of the factors.

More efficient variants of the Cantor–Zassenhaus method are known for special cases. When the degree s of the polynomial is small compared to the extension degree m, Kaltofen and Shoup [KS97] give an efficient algorithm which is as follows.

[3] For efficiency reasons, it may be preferable to choose α pseudo-randomly, such as a low-degree polynomial in the generator of k

Algorithm 3 Kaltofen–Shoup EDF for extension fields

Input: A polynomial h with irreducible factors of degree d over $k = \mathbb{F}_q[X]/f(X)$.
Output: An irreducible factor of h over k.

1: If $\deg h = d$ return h.
2: Take a random polynomial $a_0 \in k[Z]$ of degree less than $\deg h$,
3: Compute $a_1 \leftarrow \sum_{i=0}^{md-1} a_0^{q^i} \mod h$,
4: **if** q is an even power $q = 2^e$ **then**
5: Compute $a_2 \leftarrow \sum_{i=0}^{e-1} a_1^{2^i} \mod h$
6: **else**
7: Compute $a_2 \leftarrow a_1^{(q-1)/2} \mod h$
8: **end if**
9: Compute $h_0 \leftarrow \gcd(a_2, h)$ and $h_1 \leftarrow \gcd(a_2 - 1, h)$ and $h_{-1} \leftarrow h/(h_0 h_1)$,
10: Apply recursively to the smallest non-constant polynomial among h_0, h_1, h_{-1}.

We refer the reader to the original paper [KS97] for the correctness of the Kaltofen–Shoup algorithm. We are mainly interested here in its application to root extraction in cyclotomic extensions. Let r be a prime power and let f be an irreducible factor of the r-th cyclotomic polynomial Φ_r, with $s = \deg f$. Denote $\mathbb{F}_q[X]/f(X)$ by $\mathbb{F}_q(\zeta)$, where ζ is the image of X in the quotient. Given an r-th power $\alpha \in \mathbb{F}_q(\zeta)$ we want to compute an r-th root $\alpha^{1/r}$, or equivalently a linear factor of $Z^r - \alpha$ over $\mathbb{F}_q(\zeta)$.

We propose two different algorithms; one of them is quadratic in r, whereas the other one has a runtime that depends on r and s, and will perform better for small values of s.

Proposition C.3. *Let r be a prime power and let ζ be a primitive r-th root of unity; let also $s = [\mathbb{F}_q(\zeta) : \mathbb{F}_q]$. One can take r-th roots in $\mathbb{F}_q(\zeta)$ using either*

$$O(\mathsf{M}(s)\log(q) + rs^{\omega-1}\log(r)\log(s) + \mathsf{M}(rs)\log(s)\log(r))$$

or

$$O(\mathsf{M}(s)\log(q) + r\mathsf{M}(r)\log(s) + \mathsf{M}(rs)\log(s)\log(r))$$

operations in $\mathbb{F}_q$.

Proof. We use Algorithm 3 with $k = \mathbb{F}_q(\zeta)$, to get a linear factor of the polynomial $Z^r - \alpha$, so that $d = 1$ (note that $Z^r - \alpha$ splits into linear factors in $k[Z]$). We discuss Step 3, which is the dominant step. Let $f \in \mathbb{F}_q[X]$ be the defining polynomial of $\mathbb{F}_q(\zeta)$ and let h be a factor of $Z^r - \alpha$ of degree n.

We are in Case 5 of our discussion on Frobenius evaluation, and we want to compute a trace-like expression of the form τ_s. As per that discussion, two algorithms are available to do Frobenius evaluation in k (one of them uses modular composition, the other the fact that f divides $X^r - 1$). Because $s \leq r$, we deduce that a_1 can be computed in either

$$O(\mathsf{M}(s)\log(q) + rs^{\omega-1}\log(s) + \mathsf{M}(rs)\log(s))$$

or

$$O(\mathsf{M}(s)\log(q) + n\mathsf{M}(r)\log(s) + \mathsf{M}(rs)\log(s))$$

operations in $\mathbb{F}_q$, where the first term accounts for computing $\alpha^{\lfloor q/r \rfloor}$ (so we need only compute it once).

Steps 7 and 5 are again an instance of Case 5, and their cost is subsumed by Step 3. The depth of the recursion in Algorithm 3 is $\log(r)$, and the degree n is halved each time, so we obtain the desired result. □

Root finding in some extensions of cyclotomic extensions. Let $r = v^d$, where $v \neq p$ is a prime and d is a positive integer and let s be the order of q in $\mathbb{Z}/v\mathbb{Z}$. We assume that $d \geq 2$, since this will be the case whenever we want to apply the following.

Consider an extension $\mathbb{F}_q \subset k = \mathbb{F}_q[X]/f(X)$ of degree r, and let $\mathbb{F}_q(\zeta)$ and $k(\zeta)$ be extensions of degree s over $\mathbb{F}_q$ and k respectively, defined by an irreducible factor of the v-th cyclotomic polynomial over $\mathbb{F}_q$. In this paragraph, we discuss the cost of computing a v-th root in $k(\zeta)$, by adapting the root extraction algorithm given in [DS14].

Following [DS14, Algorithm 3], one reduces the root extraction in $k(\zeta)$ to a root extraction in $\mathbb{F}_q(\zeta)$; note that [DS14, Algorithm 3] reduces the root extraction to the smallest possible extension of $\mathbb{F}_p$, but projecting to $\mathbb{F}_q(\zeta)$ is more convenient here. The critical computation in this algorithm is a trace-like computation performing the reduction.

Algorithm 4 v-th root in $k(\zeta)$

Input: $a \in k(\zeta)^v$
Output: a v-th root of a

1: **repeat**
2: choose a random $c \in k(\zeta)$
3: $a' \leftarrow ac^v$
4: $\lambda \leftarrow a'^{(q^s-1)/v}$
5: $b \leftarrow 1 + \lambda + \lambda^{1+q^s} + \cdots + \lambda^{1+q^s+\cdots+q^{(r-2)s}}$
6: **until** $b \neq 0$
7: $\beta \leftarrow (a'b^v)^{1/v}$ in $\mathbb{F}_q(\zeta)$
8: **return** $\beta b^{-1} c^{-1}$

One multiplication in $k(\zeta)$ amounts to doing r multiplications modulo a degree s factor of Φ_v, and s multiplications modulo f; since $s \leq r$, this takes $O(s\mathsf{M}(r))$ operations in $\mathbb{F}_q$. The computation of $\lambda = a'^{(q^s-1)/v} = a'^{\lfloor q^s/v \rfloor}$ can then be done as explained in our discussion on Frobenius evaluation (Case 4). The cost of each modular composition is $O(s^{(\omega-1)/2} r^{(\omega+1)/2})$, for a total of $O(s^{(\omega-1)/2} r^{(\omega+1)/2} \log(s) + s\mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$.

The trace-like computation of $1 + \lambda + \lambda^{1+q^s} + \cdots + \lambda^{1+q^s+\cdots+q^{(r-2)s}}$ can be done as follows. Let x be the image of X in $k = \mathbb{F}_q[X]/f(X)$. To compute x^{q^s} we first compute x^q using $O(\mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$, and then do $\log(s)$ modular compositions in k. To compute λ^{q^s}, note that an element $\lambda \in k(\zeta)$ can be written as $\lambda = \lambda_0(x) + \lambda_1(x)\zeta + \cdots + \lambda_{s-1}(x)\zeta^{s-1}$ and that $\zeta^{q^s} = \zeta$. Therefore for any i,

$$\lambda^{q^{is}} = \sum_{j=0}^{s-1} \lambda_j(x^{q^{is}}) \left(\zeta^{q^{is}}\right)^j = \sum_{j=0}^{s-1} \lambda_j(x^{q^{is}}) \zeta^j.$$

In particular, given $x^{q^{is}}$, $\lambda^{q^{is}}$ can be computed using $O(s^{(\omega-1)/2} r^{(\omega+1)/2})$ operations in $\mathbb{F}_q$, and [DS14, Algorithm 2] can be applied in a direct way, with a cost of $O(s^{(\omega-1)/2} r^{(\omega+1)/2} \log(r) + \mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$.

The root extraction in $\mathbb{F}_q(\zeta)$ is done as in the previous paragraph, and have a negligible cost, since we assumed that $s \leq v \leq \sqrt{r}$. Therefore, we arrive at the following result.

Proposition C.4. *With k, ζ and v as above, one can extract v-th roots in $k(\zeta)$ using an expected $O(s^{(\omega-1)/2} r^{(\omega+1)/2} \log(r) + s\mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$.*

The Embedding Description problem

We are finally ready to address the problem of describing the embedding of $k = \mathbb{F}_q[X]/f(X)$ in $K = \mathbb{F}_q[Y]/g(Y)$; throughout the paper we let $m = \deg f$ and $n = \deg g$, so that $m|n$. The *embedding description problem* asks to find two elements $\alpha \in k$ and $\beta \in K$ such that $\phi(\alpha) = \beta$ for some field embedding $\phi : k \to K$. This is equivalent to saying that α and β have the same minimal polynomial.

The most obvious way to solve this problem is to take the class of X in $k = \mathbb{F}_q[X]/f(X)$ for α, and a root of f in K for β. Since f splits completely in K, we can apply Algorithm 3 for the special case $d = 1$. Using our discussion on the cost of Frobenius evaluation (precisely, Case 4), we obtain an upper bound of $O\big((nm^{(\omega+1)/2} + \mathsf{M}(m)n^{(\omega+1)/2} + m\mathsf{M}(n)\log(q))\log(m)\big)$ expected operations in $\mathbb{F}_q$ for the problem. We remark that this complexity is strictly larger than $\tilde{O}(m^2)$.

For a more specialized approach, we note that it is enough to solve the following problem: let r be a prime power such that $r|m$ and $\gcd(r, m/r) = 1$, find $\alpha_r \in k$ and $\beta_r \in K$ such that α_r and β_r have the same minimal polynomial, of degree r.

Indeed, once such α_r and β_r are known for every primary factor r of m, possible solutions to the embedding problem are

$$\alpha = \prod_{\substack{r|m,\\ \gcd(r,m/r)=1}} \alpha_r, \qquad \beta = \prod_{\substack{r|m,\\ \gcd(r,m/r)=1}} \beta_r,$$

or

$$\alpha = \sum_{\substack{r|m,\\ \gcd(r,m/r)=1}} \alpha_r, \qquad \beta = \sum_{\substack{r|m,\\ \gcd(r,m/r)=1}} \beta_r.$$

Moreover, to treat the general embedding description problem, it is sufficient to treat the case where $[k : \mathbb{F}_q] = [K : \mathbb{F}_q] = r$. Indeed, we can reduce to this situation by applying Proposition C.2, at an additional cost of $O(n^{(\omega+1)/2}\log(n) + \mathsf{M}(n)\log(q))$ for each primary factor r. Therefore, to simplify the exposition, we focus on algorithms solving the following problem.

Problem C.5. Let r be a prime power and k, K two extensions of $\mathbb{F}_q$ of degree r. Describe an isomorphism between k and K.

All algorithms presented next are going to rely on one common principle: construct an element in k (and in K) such that its minimal polynomial (or, equivalently, its orbit under the absolute Galois group of $\mathbb{F}_q$) is uniquely (or *almost* uniquely) defined.

C.3 Kummer-type algorithms

In this section, we review what we call *Kummer-type* approaches to the embedding problem for prime power degree extensions. We briefly review the works of Lenstra [Len91],

and Allombert [All02a; All02b], then we give variants of these algorithms with significantly lower complexities. As stated above, we let k, K be degree r extensions of $\mathbb{F}_q$, where r is a prime power, and we let p be their characteristic. We give our fast versions of the algorithms for two separate cases: the case $p \nmid r$ is treated in Section C.3, the case $r = p^d$, where d is a positive integer, is treated in Section C.3. Finally, in Section C.3 we give a variant of the case $p \nmid r$ better suited for the case where r is a high-degree prime power.

In [Len91], Lenstra proves that given two finite fields of the same size, there exists a deterministic polynomial time algorithm that finds an isomorphism between them. The focus of the paper is on theoretical computational complexity; in particular, it avoids using randomized subroutines, such as polynomial factorization. In [All02a; All02b], Allombert gives a similar approach with more focus on practical efficiency. In contrast to Lenstra's, his algorithm relies on polynomial factorization, thus it is polynomial time Las Vegas. Even though neither of the two algorithms is given a detailed complexity analysis, both rely on solving linear systems, thus a rough analysis yields an estimate of $O(r^\omega)$ operations in $\mathbb{F}_q$ in both cases.

The idea of Lenstra's algorithm is as follows. Assume that r is prime, and let $\mathbb{F}_q[\zeta], k[\zeta]$ denote the ring extensions $\mathbb{F}_q[Z]/\Phi_r(Z), k[Z]/\Phi_r(Z)$ where Φ_r is the r-th cyclotomic polynomial. From a normal basis of k, computed using linear algebra, Lenstra constructs an element $\theta_1 \in k[\zeta]$ such that θ_1 and $\tau_1 = \theta_1^r$ are generators of the *Teichmüller subgroups* of $k[\zeta]$ and $\mathbb{F}_q[\zeta]$, respectively. He then proves that there is an $\mathbb{F}_q[\zeta]$-isomorphism of rings

$$k[\zeta] \xrightarrow{\sim} \mathbb{F}_q[\zeta][Y]/(Y^r - \tau_1)$$

sending θ_1 to the class of Y. Doing the same for $K[\zeta]$, elements $\theta_2 \in K[\zeta]$ and $\tau_2 \in \mathbb{F}_q[\zeta]$ and an $\mathbb{F}_q[\zeta]$-isomorphism

$$\mathbb{F}_q[\zeta][Y]/(Y^r - \tau_2) \xrightarrow{\sim} K[\zeta],$$

that sends the class of Y to θ_2, is obtained. Since τ_1 and τ_2 both generate the Teichmüller subgroup of $\mathbb{F}_q[\zeta]$, there exists an integer $j > 0$ such that $\tau_1 = \tau_2^j$, then the map

$$\begin{array}{rccc} \psi: & k[\zeta] & \to & K[\zeta] \\ & \theta_1 & \mapsto & \theta_2^j \end{array}$$

is an isomorphism of rings. Finally, denoting by Δ the automorphism group of $k[\zeta]$ over k, an embedding $k \hookrightarrow K$ is obtained by restricting the above isomorphism ψ to the fixed field $k[\zeta]^\Delta$. To summarize, the algorithm is made of three steps:

- Construct elements $\theta_1 \in k[\zeta]$ and $\theta_2 \in K[\zeta]$;
- Letting $\tau_i = \theta_i^r$, find the integer j such that $\tau_1 = \tau_2^j$ by a discrete logarithm computation in $\mathbb{F}_q[\zeta]$;
- Compute $\alpha \in k$ and $\beta \in K$ as some functions of θ_1, θ_2^j invariant under Δ.

The algorithm is readily generalized to prime powers r by iterating this procedure.

Allombert's algorithms differ from Lenstra's in two key steps, both resorting to polynomial factorization. First, he computes an irreducible factor h of the cyclotomic polynomial Φ_r of degree s, and so constructs a field extension $\mathbb{F}_q(\zeta)$ as $\mathbb{F}_q[Z]/h(Z)$. Then he defines $k[\zeta] = k[Z]/h(Z)$ and $K[\zeta] = K[Z]/h(Z)$ (note that these are not fields if r is

not prime), and constructs $\theta_1 \in k[\zeta]$ and $\theta_2 \in K[\zeta]$ in a way equivalent to Lenstra's using linear algebra. At this point, rather than computing a discrete logarithm, Allombert points out that there exists a $c \in \mathbb{F}_q(\zeta)$ such that $\theta_1 \mapsto c\theta_2$ defines an isomorphism, and that such value can be computed as the r-th root of θ_1^r/θ_2^r. Finally, by making the automorphism group of $k[\zeta]$ over k act on θ_1 and θ_2, he obtains an embedding $k \hookrightarrow K$.

Allombert's algorithm

In this section, we analyze the complexity of Allombert's original algorithm [All02a], that of its revised version [All02b], and we present new variants with the best known asymptotic complexities. The main difference with respect to the versions presented in [All02a; All02b] is in the way we compute θ_1, θ_2, which are solutions to Hilbert's theorem 90 as will become clear below. Whereas Allombert resorts to linear algebra, we rely instead on evaluation formulas that have a high probability of yielding a solution. Recently, Narayanan [Nar18, Sec. 3] independently described a variant which is similar to our Proposition C.8 in the special case $s = 1$.

General strategy

Let $k = \mathbb{F}_q[X]/f(X)$ where f has degree r, a prime power, and let x be the image of X in k. Let $h(Z)$ be an irreducible factor of the r-th cyclotomic polynomial over $\mathbb{F}_q$. Then h has degree s where s is the order of q in the multiplicative group $(\mathbb{Z}/r\mathbb{Z})^\times$. We form the field extension $\mathbb{F}_q(\zeta) \cong \mathbb{F}_q[Z]/h(Z)$ and the ring extension $k[\zeta] = k[Z]/h(Z) \cong k \otimes \mathbb{F}_q(\zeta)$ where ζ is the image of Z in the quotients. The action of the Galois group $\mathrm{Gal}(k/\mathbb{F}_q)$ can be extended to $k[\zeta]$ by

$$\begin{array}{rccc} \sigma: & k[\zeta] & \to & k[\zeta], \\ & x \otimes \zeta & \mapsto & x^q \otimes \zeta. \end{array}$$

Allombert shows (see [All02a, Prop. 3.2]) that σ is an automorphism of $\mathbb{F}_q(\zeta)$-algebras, and that its fixed set is isomorphic to $\mathbb{F}_q(\zeta)$. The same can be done for the ring $K[\zeta]$. Let us restate the algorithm for clarity.

Algorithm 5 Allombert's algorithm

Input: Field extensions k, K of $\mathbb{F}_q$ of degree r.
Output: The description of a field embedding $k \to K$.

1: Factor the r-th cyclotomic polynomial and make the extensions $\mathbb{F}_q(\zeta), k[\zeta], K[\zeta]$;
2: Find $\theta_1 \in k[\zeta]$ such that $\sigma(\theta_1) = \zeta\theta_1$;
3: Find $\theta_2 \in K[\zeta]$ such that $\sigma(\theta_2) = \zeta\theta_2$;
4: Compute an r-th root c of θ_1^r/θ_2^r in $\mathbb{F}_q(\zeta)$;
5: Let α, β be the constant terms of $\theta_1, c\theta_2$ respectively;
6: **return** The field embedding defined by $\alpha \mapsto \beta$.

The cyclotomic polynomial Φ_r is factored over $\mathbb{F}_q$ using [Sho94b, Theorem 9], and r-th root extraction in $\mathbb{F}_q(\zeta)$ is done using Proposition C.3, so we are left with the problem of finding θ_1 (and θ_2), that is, instances of Hilbert's theorem 90.

We now show how to do it in the extension $k[\zeta]/\mathbb{F}_q(\zeta)$, the case of $K[\zeta]$ being analogous. We review approaches due to Allombert, that rely on linear algebra, and propose new algorithms that rely on evaluation formulas and ultimately polynomial arithmetic. Note that all these variants can be directly applied to any extension degree

r as long as $p \nmid r$, and do not require r to be a prime power. Nevertheless, in practice, it is more efficient to perform computations for each primary factor independently and glue the results together in the end.

If A is a polynomial with coefficients in $\mathbb{F}_q(\zeta)$, we will denote by $\hat{A}$ the morphism $A(\sigma)$ of the algebra $k[\zeta]$; note that the usual property of *q-polynomials* holds: $\widehat{AB} = \hat{A} \circ \hat{B}$.

Algorithms relying on linear algebra

As some algorithmic details were omitted in Allombert's publications, and no precise complexity analysis was performed, we extracted the details from PARI/GP source code [PARI] and perform the complexity analysis here. We also propose another variant, using an algorithm by Paterson and Stockmeyer.

Allombert's original algorithm. A direct solution to Hilbert's theorem 90 is to find a non-zero $\theta \in k[\zeta]$ such that $\widehat{(S-\zeta)}(\theta) = 0$.

The original version of Allombert's algorithm [All02a] does precisely this, by computing the matrix of the Frobenius automorphism σ of $k/\mathbb{F}_q$ using $O(\mathsf{M}(r)\log(q) + r\mathsf{M}(r))$ operations in $\mathbb{F}_q$ and then an eigenvalue of σ for ζ over $\mathbb{F}_q(\zeta)$ using linear algebra, at a cost of $O((rs)^\omega)$ operations in $\mathbb{F}_q$. This gives a total cost of $O(s\mathsf{M}(r)\log(q) + (rs)^\omega)$ operations in $\mathbb{F}_q$.

Allombert's revised algorithm. Allombert's revision of his own algorithm [All02b] uses the factorization

$$h(S) = (S-\zeta)b(S). \tag{C.2}$$

If we set $h(S) = S^s + \sum_{i=0}^{s-1} h_i S^i$, we can explicitly write b as

$$b(S) = \sum_{i=0}^{s-1} b_i(S)\zeta^i, \quad \text{where} \quad \begin{cases} b_{s-1}(S) = 1, \\ b_{i-1}(S) = b_i(S)S + h_i. \end{cases} \tag{C.3}$$

Indeed, Horner's rule shows that $b_{-1}(S) = h(S)$, and by direct calculation we find that $(S-\zeta)\cdot b(S) = b_{-1}(S)$.

We get a solution to Hilbert's theorem 90 by evaluating $b(S) = h(S)/(S-\zeta)$ on an element in the kernel of $\hat{h}$ over k, linear algebra now taking place over $\mathbb{F}_q$ rather than $\mathbb{F}_q(\zeta)$. The details on the computation of $\hat{h}$ were extracted from PARI/GP source code and yield the following complexity.

Proposition C.6. *Using Allombert's revised algorithm, a solution θ to Hilbert's theorem 90 can be computed in $O(\mathsf{M}(r)\log(q) + sr\mathsf{M}(r) + r^\omega)$ operations in $\mathbb{F}_q$.*

Proof. As in Allombert's original algorithm, one first computes the matrix of σ over k at a cost of $O(\mathsf{M}(r)\log(q) + r\mathsf{M}(r))$ operations in $\mathbb{F}_q$.

To get the matrix of $\hat{h}$ over k, one first computes the powers x^{q^i} for $0 \le i \le s$ using the matrix of σ, at a cost of $O(sr^2)$ operations in $\mathbb{F}_q$. From them, one can iteratively compute the powers x^{jq^i} for $2 \le j \le r$ for a total cost of $O(sr\mathsf{M}(r))$ operations in $\mathbb{F}_q$, and iteratively compute the matrix of $\hat{h}$ for an additional total cost of $O(sr^2)$ operations in $\mathbb{F}_q$, accounting for the scalar multiplications by the coefficients of h. The total cost is therefore dominated by $O(sr\mathsf{M}(r))$ operations in $\mathbb{F}_q$.

Given the matrix of $\hat{h}$ over k, computing an element in its kernel costs $O(r^\omega)$ operations in $\mathbb{F}_q$. The final evaluation of $\hat{b}$ is done using Eq. (C.3) and the matrix of σ for Frobenius computations, for a cost of $O(sr^2)$ operations in $\mathbb{F}_q$. □

Using the Paterson–Stockmeyer algorithm. Given the matrix M_σ of σ, there is a natural way of evaluating $\hat{h}$ at a reduced cost: the Paterson–Stockmeyer algorithm [PS73] computes the matrix of $\hat{h}$ and $h(M_\sigma)$, using $O(\sqrt{s}r^\omega)$ operations in $\mathbb{F}_q$. The evaluations of σ that take a total of $O(sr^2)$ operations in $\mathbb{F}_q$ can be done directly using modular exponentiations, for a total of $O(s\mathsf{M}(r)\log(q))$.

Proposition C.7. *Using the Paterson–Stockmeyer algorithm and modular exponentiations, a solution θ to Hilbert's theorem 90 can be computed in $O(s\mathsf{M}(r)\log(q)+\sqrt{s}r^\omega)$ operations in $\mathbb{F}_q$.*

Although this complexity is not as good as the ones we will obtain next, this variant performs reasonably well in practice, as discussed in Section C.7.

Algorithms relying on polynomial arithmetic

It is immediate to see that the minimal polynomial of σ over $k[\zeta]$ is S^r-1; by direct calculation, we verify that it factors as

$$S^r-1=(S-\zeta)\cdot\Theta(S)=(S-\zeta)\sum_{i=0}^{r-1}\zeta^{-i-1}S^i. \tag{C.4}$$

Hence, we can set

$$\theta_a=\hat{\Theta}(a)=a\otimes\zeta^{-1}+\sigma(a)\otimes\zeta^{-2}+\cdots+\sigma^{r-1}(a)\otimes\zeta^{-r} \tag{C.5}$$

for some $a\in k$ chosen at random. Because of Eq. (C.4), θ_a is a solution as long as it is non-zero. This is reminiscent of Lenstra's algorithm [Len91, Th. 5.2].

To ensure the existence of a such that $\theta_a\neq 0$, we only need to prove that k is not entirely contained in $\ker\hat{\Theta}$. But the maps σ^i restricted to k are all distinct, thus Artin's theorem on character independence (see [Lan02, Ch VI, Theorem 4.1]) shows that they are linearly independent, and therefore $\hat{\Theta}$ is not identically zero on k. In practice, we take $a\in k$ at random until $\theta_a\neq 0$. Since the map $\hat{\Theta}$ is $\mathbb{F}_q$-linear and non-zero, it has rank at least 1, thus a random θ_a is zero with probability less than $1/q$. Therefore, we only need $O(1)$ trials to find θ_1 (and θ_2).

Using the polynomial $b(S)$ introduced in Eq. (C.2), and defining $g(S)=(S^r-1)/h(S)$, we can rewrite Eq. (C.4) as

$$\Theta(S)=b(S)\cdot g(S). \tag{C.6}$$

Then, the morphism $\hat{\Theta}$ can be evaluated as $\hat{b}\circ\hat{g}$, the advantage being that g has coefficients in $\mathbb{F}_q$, rather than in $\mathbb{F}_q(\zeta)$: we set $\tau_a=\hat{g}(a)$ for some $a\in k$ chosen at random and compute $\theta_a=\hat{b}(\tau_a)$ using Eq. (C.3), yielding a solution to Hilbert's theorem 90 as soon as $\tau_a\neq 0$. As before, $O(1)$ trials are enough to get $\theta_a\neq 0$.

We now give three variations on the above algorithm to compute a candidate solution θ_a more efficiently. Which algorithm has the best asymptotic complexity depends on the value of s with respect to r; we arrange them by increasing s.

First solution: divide-and-conquer recursion. We use a recursive algorithm similar to the computation of trace-like functions in Proposition C.1, to directly evaluate θ_a using Eq. (C.5). For $j\geq 1$, let $\xi_j=\sigma^j(x)$ and $\theta_{a,j}=a\otimes\zeta^{-1}+\sigma(a)\otimes\zeta^{-2}+\cdots+\sigma^{j-1}(a)\otimes\zeta^{-j}$, so that we want to compute $\theta_{a,r}=\theta_a$. For $j=1$, we have $(\xi_1,\theta_{a,1})=(x^q,a\zeta^{-1})$; for $i,j\geq 1$, we have the following recursive relations:

$$(\xi_{i+j},\theta_{a,i+j})=(\sigma^i(\xi_j),\theta_{a,i}+\sigma^i(\theta_{a,j})\zeta^{-i}). \tag{C.7}$$

Proposition C.8. *Given $a \in k$, the value θ_a in Eq. (C.5) can be computed using*

$$O(s^{(\omega-1)/2} r^{(\omega+1)/2} \log(r) + \mathsf{M}(r)\log(q))$$

operations in $\mathbb{F}_q$.

Proof. The value ξ_1 is computed by binary powering using $O(\mathsf{M}(r)\log(q))$ operations, while the value $\theta_{a,1}$ is deduced from the polynomial h using $O(rs)$ operations.

To compute the recursive formulas in Eq. (C.7) we use the same technique as in Proposition C.1: given $b \in k[\zeta]$, the value $\sigma^j(b)$ is computed as the modular composition of the polynomial $b(x,z)$ with the polynomial $\xi_j(x)$ in the first argument. Each modular composition in $k[\zeta]$ is done using s modular compositions in k, at a cost of $O(s^{(\omega-1)/2} r^{(\omega+1)/2})$ operations (see Note C.2). Multiplications by ζ^{-j} are done by seeing the elements of $k[\zeta]$ as polynomials in x over $\mathbb{F}_q(\zeta)$, thus performing r multiplications modulo h, at a cost of $O(r\mathsf{M}(s))$ operations. Given that the total depth of the recursion is $O(\log(r))$, we obtain the stated bound. □

Second solution: automorphism evaluation. We use Eq. (C.6) and Eq. (C.3) to compute θ_a as $\theta_a = \hat{b} \circ \hat{g}(a)$.

Proposition C.9. *Given $a \in k$, the value θ_a in Eq. (C.5) can be computed using*

$$O(r^{(\omega^2-4\omega-1)/(\omega-5)} + (s + r^{2/(5-\omega)})\mathsf{M}(r)\log(q))$$

operations in $\mathbb{F}_q$.

Proof. We proceed in two steps. We first compute $\hat{g}(a)$ using the *automorphism evaluation* algorithm of Kaltofen and Shoup [KS98, Algorithm AE], at a cost of $O(r^{(\omega+1)/2+(3-\omega)|\beta-1/2|} + r^{(\omega+1)/2+(1-\beta)(\omega-1)/2} + r^{\beta}\mathsf{M}(r)\log(q))$, for any $0 \le \beta \le 1$. Choosing $\beta = 2/(5-\omega)$ minimizes the overall runtime, giving the exponents reported above.

We then use Eq. (C.3) to compute $\theta_a = \sum_{i=0}^{s-1} a_i \otimes \zeta^i$, where $a_{s-1} = \hat{g}(a)$, and $a_{i-1} = \sigma(a_i) + h_i \hat{g}(a)$. The cost of this computation is dominated by the evaluations of σ, which take $O(\mathsf{M}(r)\log(q))$ operations each, thus contributing $O(s\mathsf{M}(r)\log(q))$ total operations. □

Third solution: multipoint evaluation. Finally, we can compute all the values $\sigma(a), \ldots, \sigma^{r-1}(a)$ directly, write θ_a as a polynomial in x and ζ of degree $r-1$ in both variables, and reduce modulo h for each power x^i.

Proposition C.10. *Given $a \in k$, the value θ_a in Eq. (C.5) can be computed using*

$$O(\mathsf{M}(r^2)\log(r) + \mathsf{M}(r)\log(q))$$

operations in $\mathbb{F}_q$.

Proof. The values $\sigma(a), \ldots, \sigma^{r-1}(a)$ can be computed by binary powering using $O(r\mathsf{M}(r)\log(q))$. We can do slightly better using the iterated Frobenius technique of von zur Gathen and Shoup [GS92, Algorithm 3.1] (see also [GG99, Ch. 14.7]), which costs of $O(\mathsf{M}(r^2)\log(r) + \mathsf{M}(r)\log(q))$ operations. The final reduction modulo h costs $O(r\mathsf{M}(r)\log(r))$ operations, which is negligible in front of the previous step. □

The following proposition summarizes our analysis. To clarify the order of magnitude of the exponents, let us assume $q = O(1)$ and neglect polylogarithmic factors; then, if $\omega = 2.38$ (best bound to date), the runtimes are $O(s^{0.69}r^{1.69})$ for $s \in O(r^{0.23})$, $O(r^{1.85} + s^{1.38}r)$ for $s \in \Omega(r^{0.23})$ and $s \in O(r^{0.72})$, and $\tilde{O}(r^2)$ otherwise. For $\omega = 3$, all costs are at best quadratic.

Proposition C.11. *Given k, K of degree r over $\mathbb{F}_q$, assuming that s is the order of q in $(\mathbb{Z}/r\mathbb{Z})^\times$, Algorithm 5 computes its output using*

- $O(s^{(\omega-1)/2}r^{(\omega+1)/2}\log(r) + \mathsf{M}(r)\log(q))$ *expected operations in $\mathbb{F}_q$ if $s \in O(r^{(\omega-3)/(\omega-5)})$, or*
- $O(r^{(\omega^2-4\omega-1)/(\omega-5)} + (s + r^{2/(5-\omega)})\mathsf{M}(r)\log(q) + s^{\omega-1}r\log(r)\log(s))$ *expected operations in $\mathbb{F}_q$ if if $s \in \Omega(r^{(\omega-3)/(\omega-5)})$ and $s \in O(r^{1/(\omega-1)})$, or*
- $O(\mathsf{M}(r^2)\log^2(r) + \mathsf{M}(r)\log(r)\log(q))$ *expected operations in $\mathbb{F}_q$ otherwise.*

Proof. The cost of factoring the r-th cyclotomic polynomial is an expected $O(\mathsf{M}(r)\log(rq))$ operations in $\mathbb{F}_q$, using [Sho94b, Theorem 9]. This is negligible compared with other steps. The solutions θ_1, θ_2 to Hilbert's theorem 90 are computed as described above, according to the size of s. The powers θ_1^r, θ_2^r are computed using Kronecker substitution in $O(\mathsf{M}(sr)\log(r))$ operations, which is also negligible. Finally, the cost of computing an r-th root in $\mathbb{F}_q(\zeta)$ is given by Proposition C.3 and can not be neglected.

Combining the costs coming from the solution to Hilbert's theorem 90 and the r-th root extraction, we obtain the following complexities according to s.

- If we use the algorithm described in our first solution, combining Proposition C.8 with the first case of Proposition C.3, we obtain an estimate of $O(s^{(\omega-1)/2}r^{(\omega+1)/2}\log(r) + \mathsf{M}(r)\log(q))$ operations.
- If we use the algorithm described in our second solution, combining Proposition C.9 with the first case of Proposition C.3, we obtain an estimate of $O(r^{(\omega^2-4\omega-1)/(\omega-5)} + (s + r^{2/(5-\omega)})\mathsf{M}(r)\log(q) + s^{\omega-1}r\log(r)\log(s) + \mathsf{M}(rs)\log(r)\log(s))$ operations.
- Otherwise, we use the algorithm described in our third solution. Combining Proposition C.10 with the second case of Proposition C.3, and replacing s with r everywhere, we obtain an estimate of $O(\mathsf{M}(r^2)\log^2(r) + \mathsf{M}(r)\log(r)\log(q))$ expected operations.

For $s \in O(r^{(\omega-3)/(\omega-5)})$, the first solution has the better runtime. Assuming $s \in \Omega(r^{(\omega-3)/(\omega-5)})$, the runtime in the second case can be written as $O(r^{(\omega^2-4\omega-1)/(\omega-5)} + (s + r^{2/(5-\omega)})\mathsf{M}(r)\log(q) + s^{\omega-1}r\log(r)\log(s))$. If in addition s is in $O(r^{1/(\omega-1)})$, this runtime is subquadratic, that is, better than that in our third solution. □

The Artin–Schreier case

This section is devoted to the case $r = p^d$ for some positive integer d. The technique we present here originates in Adleman and Lenstra's work [AL86, Lemma 5], and appears again in Lenstra's [Len91] and Allombert's [All02a]. The chief difference with previous work once again consists in replacing linear algebra with a technique to solve

the additive version of Hilbert's theorem 90 similar to the one in the previous section. Recently, Narayanan [Nar18, Sec. 4] independently described a related variant with a similar complexity.

The idea is to build a tower inside the extension $k/\mathbb{F}_q$ using polynomials of the form $X^p - X - a$ where $a \in k$. To start, let $a_1 \in \mathbb{F}_q$ be such that $\mathrm{Tr}_{\mathbb{F}_q/\mathbb{F}_p}(a_1) \neq 0$. Let $\sigma \in \mathrm{Gal}(\mathbb{F}_q/\mathbb{F}_p)$ be a generator of the Galois group. Then by the additive version of Hilbert's theorem 90 there is no element $\alpha \in \mathbb{F}_q$ such that $\sigma(\alpha) - \alpha = a_1$. Equivalently, the polynomial $f_1 = X^p - X - a_1$ has no root in $\mathbb{F}_q$. By the Artin–Schreier theorem in [Lan02, Ch VI], f_1 is irreducible over $\mathbb{F}_q$. For a root $\alpha_1 \in k$ of f_1, the extension $\mathbb{F}_q(\alpha_1)/\mathbb{F}_q$ is of degree p. Now let $a_2 = a_1\alpha_1^{p-1}$. Then, by [AL86, Lemma 5], the polynomial $f_2 = X^p - X - a_2$ is irreducible over $\mathbb{F}_q(\alpha_1)$. So, for a root $\alpha_2 \in k$ of f_2 the extension $\mathbb{F}_q(\alpha_2, \alpha_1)/\mathbb{F}_q(\alpha_1)$ is of degree p. Continuing the above process we build a tower

$$\mathbb{F}_q \subset \mathbb{F}_q(\alpha_1) \subset \cdots \subset \mathbb{F}_q(\alpha_1, \cdots, \alpha_d) = k. \tag{C.8}$$

The idea of building such tower using the Artin–Schreier polynomials f_i can also be found in [Len91; All02a; Sho93]. By construction, $\alpha_i \notin \mathbb{F}_q(\alpha_1, \cdots, \alpha_{i-1})$ for all $1 \leq i \leq d$. This means that the minimal polynomial of α_d over $\mathbb{F}_q$ is of degree $r = p^d$. Therefore, $k = \mathbb{F}_q(\alpha_d)$, and the element α_d is uniquely defined up to $\mathbb{F}_q$-isomorphism.

The above construction boils down to computing a root of the polynomial $f = X^p - X - a \in k[X]$. We now show how to efficiently compute such a root. By construction, a is always in an intermediate subfield $\mathbb{F}_q \subseteq k' \subset k$. This means

$$\mathrm{Tr}_{k/\mathbb{F}_p}(a) = \mathrm{Tr}_{k'/\mathbb{F}_p}(\mathrm{Tr}_{k/k'}(a)) = \mathrm{Tr}_{k'/\mathbb{F}_p}(p^i a) = 0$$

for some $i > 0$. By Hilbert's theorem 90 there exists $\alpha \in k$ such that $\alpha - \sigma(\alpha) = -a$ for a generator $\sigma \in \mathrm{Gal}(k/\mathbb{F}_p)$. In other words, $\alpha^p - \alpha - a = 0$. Therefore, α is a root of f. On the other hand, for a random element $\theta \in k$ with nonzero trace, α can be explicitly set as

$$\alpha = \frac{1}{\mathrm{Tr}(\theta)}[a\sigma(\theta) + (a + \sigma(a))\sigma^2(\theta) + \cdots + (a + \sigma(a) + \cdots + \sigma^{rt-2}(a))\sigma^{rt-1}(\theta)] \tag{C.9}$$

where $t = [\mathbb{F}_q : \mathbb{F}_p]$. To compute α using Eq. (C.9) efficiently, for $j \geq 1$, we define $\xi_j = \sigma^j(x)$, and, for u in k,

$$\beta_{u,j} = u + \sigma(u) + \cdots + \sigma^{j-1}(u), \quad \alpha_{u,j} = \beta_{a,1}\sigma(u) + \cdots + \beta_{a,j}\sigma^j(u).$$

Our goal is to ultimately compute $\beta_{\theta,rt} = \mathrm{Tr}(\theta)$ and $\alpha = \beta_{\theta,rt}^{-1}\alpha_{\theta,rt}$. For this, we show how to compute quadruples of the form $(\xi_j, \beta_{a,j}, \beta_{\theta,j}, \alpha_{\theta,j})$, for $j \geq 1$.

For $j = 1$, we have $(\xi_1, \beta_{a,1}, \beta_{\theta,1}, \alpha_{\theta,1}) = (x^q, a, \theta, a\theta^q)$; for $i, j \geq 1$, a calculation gives

$$\begin{aligned}(\xi_{i+j}, \beta_{a,i+j}, \beta_{\theta,i+j}, \alpha_{\theta,i+j}) = \\ \big(\sigma^j(\xi_i),\ \beta_{a,j} + \sigma^j(\beta_{a,i}),\ \beta_{\theta,j} + \sigma^j(\beta_{\theta,i}),\ \alpha_{\theta,j} + \sigma^j(\alpha_{\theta,i}) + \beta_{a,j}\sigma^{j+1}(\beta_{\theta,i})\big).\end{aligned} \tag{C.10}$$

In particular, the values $\beta_{\theta,rt}$, and $\alpha_{\theta,rt}$ can be computed recursively, in $O(\log(rt))$ steps. The initial values $(\xi_1, \beta_{a,1}, \beta_{\theta,1}, \alpha_{\theta,1})$ are computed using $O(\mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$. For higher indices, as before, the action of σ^j is the same as composing with ξ_j, so each step of the recursion is dominated by $O(1)$ modular compositions over $\mathbb{F}_q$, at the cost of $O(r^{(\omega+1)/2})$ operations in $\mathbb{F}_q$. Therefore, the cost of computing a root of f is $O(r^{(\omega+1)/2}\log(rt) + \mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$.

Now, to compute α_d in Eq. (C.8) we need to take d roots where $d \in O(\log(r)/\log(p))$ which leads to the following result. (Note that ξ_1 is computed only once and reused thereafter.)

Proposition C.12. *Let $r = p^d$ for a positive integer d, and let $t = [\mathbb{F}_q : \mathbb{F}_p]$. An isomorphism of two extensions $k/\mathbb{F}_q$, $K/\mathbb{F}_q$ of degree r can be constructed using $O(r^{(\omega+1)/2}\log(rt)\log(r) + \mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$.*

High-degree prime powers

We end this section with an algorithm that is particularly efficient when the extension degree r is a high-degree prime power. Allombert's algorithm works well in this case, however its complexity depends linearly on the order s of q modulo r. If $r = v^d$ for some prime $v \neq p$, it is natural to seek an algorithm which depends on the order of q modulo v instead. The idea we present is a variation on Lenstra's algorithm, using successive v-th root extractions. We are not aware of this algorithm appearing anywhere in the literature. We also note that Narayanan [Nar18, Sec. 5] recently published a radically different generalization of Allombert's algorithm with a very similar complexity in r (his algorithm has much worse complexity in q, though).

An overview of our construction is as follows. Let $r = v^d$ where $v \neq p$ is a prime and d is a positive integer. Suppose the extension $k/\mathbb{F}_q$ is of degree r. Let s be the order of q in $\mathbb{Z}/v\mathbb{Z}$, and write $q^s - 1 = uv^t$ where $\gcd(v,u) = 1$. Since $q^r = q^{v^d} = q \bmod v$, we see that q^r has the same order s in $\mathbb{Z}/v\mathbb{Z}$. So the cyclotomic field extensions $\mathbb{F}_q(\zeta)/\mathbb{F}_q, k(\zeta)/k, K(\zeta)/K$ are all of degree s. We move to these extensions by obtaining an irreducible factor of the v-th cyclotomic polynomial over $\mathbb{F}_q$. Next, we obtain a random non-v-th power η in $\mathbb{F}_q(\zeta)^*$.

It follows from $v^t \mid q^s - 1$ that $v^{t+d} \mid q^{rs} - 1$. So we can compute an r-th root θ_1 of η in $k(\zeta)$ using d successive v-th root extractions in $k(\zeta)$. Since η is a non-v-th power, the polynomial $Y^r - \eta$ is irreducible over $\mathbb{F}_q(\zeta)$ and there is an $\mathbb{F}_q(\zeta)$-isomorphism

$$\mathbb{F}_q(\zeta)(\theta_1) \xrightarrow{\sim} \mathbb{F}_q(\zeta)[Y]/(Y^r - \eta)$$

that sends θ_1 to Y. From this we see that $\mathbb{F}_q(\theta)$ and $k(\zeta)$ have the same degree over $\mathbb{F}_q$, and hence $\mathbb{F}_q(\theta_1) = k(\zeta)$. Similarly, we find $\theta_2 \in K(\zeta)$. So we have an $\mathbb{F}_q(\zeta)$-embedding

$$\phi : k(\zeta) \xrightarrow{\sim} \mathbb{F}_q(\zeta)[Y]/(Y^r - \eta) \xrightarrow{\sim} K(\zeta)$$

that sends θ_1 to θ_2. Let $\alpha = \mathrm{Tr}_{k(\zeta)/k}(\theta_1)$ and $\beta = \mathrm{Tr}_{K(\zeta)/K}(\theta_2)$. Then $k = \mathbb{F}_q(\alpha)$ and $K = F_q(\beta)$, see [Sho93, Algorithm 13] and the proof of [Sho90, Theorem 2.1]. Since ϕ commutes with traces, the map $\alpha \mapsto \beta$ defines an isomorphism.

The main difficulty in applying such an algorithm resides in computing efficiently v-th roots in $k(\zeta)$, for which we use Proposition C.4; this yields the main result of this section.

Theorem C.13. *Let $r = v^d$ where $v \neq p$ is a prime and d is a positive integer. Also let s be the order of q in $\mathbb{Z}/v\mathbb{Z}$. Given extensions $k/\mathbb{F}_q$, $K/\mathbb{F}_q$ of degree r, an embedding $k \hookrightarrow K$ can be constructed at the cost of an expected $O(s^{(\omega-1)/2} r^{(\omega+1)/2}\log(r)^2 + s\mathsf{M}(r)\log(r)\log(q))$ operations in $\mathbb{F}_q$.*

Proof. We can construct the embedding of Theorem C.13 as follows. We first build the extensions $k(\zeta)/\mathbb{F}_q(\zeta)$ and $K(\zeta)/\mathbb{F}_q(\zeta)$. Let η be a non v-th power in $\mathbb{F}_q(\zeta)$. Then

Algorithm 6 Kummer-type algorithm for extension towers

Input: Extensions $k/\mathbb{F}_q$, $K/\mathbb{F}_q$ of degree prime-power $r = v^d$, with $v \neq p$.
Output: The description of a field embedding $k \hookrightarrow K$.

1: Factor the v-th cyclotomic polynomial over $\mathbb{F}_q$ to build the extensions $k(\zeta)/\mathbb{F}_q(\zeta)$ and $K(\zeta)/\mathbb{F}_q(\zeta)$;
2: Find a random non v-th power $\eta \in \mathbb{F}_q(\zeta)$;
3: Compute r-th roots θ_1, θ_2 of η in $k(\zeta), K(\zeta)$;
4: Compute $\alpha = \mathrm{Tr}_{k(\zeta)/k}(\theta_1)$ and $\beta = \mathrm{Tr}_{K(\zeta)/K}(\theta_2)$;
5: **return** The field embedding defined by $\alpha \mapsto \beta$.

η is an r-power in $k(\zeta)$ and $K(\zeta)$. To obtain r-th roots $\theta_1 \in k$, $\theta_2 \in K$ of η we take d successive v-th roots.

Step 1 is done using [Sho94b, Theorem 9], which takes $O(\mathsf{M}(v)\log(vq))$ operations in $\mathbb{F}_q$. We do Step 2 by taking random elements in $\mathbb{F}_q(\zeta)$ until a non v-th power is found. Testing whether η is a v-th power amounts to computing $\eta^{(q^s-1)/v}$ in $\mathbb{F}_q(\zeta)$, which can be done in $O(s^{(\omega-1)/2}\log(s) + \mathsf{M}(s)\log(v)\log(s) + \mathsf{M}(s)\log(q))$ operations in $\mathbb{F}_q$, in view of our discussion in Section C.2.

Step 3 is done using $d = O(\log(r)/\log(v))$ successive root extractions, each of which takes an expected $O(s^{(\omega-1)/2} r^{(\omega+1)/2}\log(r) + s\mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$. Writing θ_1 and θ_2 as polynomials in ζ with coefficients in k and K, reps., we see that computing the traces of Step 4 boils down to computing the traces $\mathrm{Tr}_{\mathbb{F}_q(\zeta)/\mathbb{F}_q}(\zeta^i)$ for $0 \leq i < s$. The cost of this is dominated by those of the previous steps. Therefore, Algorithm 6 runs in an expected $O(s^{(\omega-1)/2} r^{(\omega+1)/2}\log(r)^2 + s\mathsf{M}(r)\log(r)\log(q)$ operations in $\mathbb{F}_q$. □

C.4 Rains' algorithm

We now move on to a different family of algorithms based on the theory of algebraic groups. The simplest of these is Pinch's cyclotomic algorithm [Pin92]. The idea is very simple: given r, select an integer ℓ such that $[\mathbb{F}_q(\mu_\ell) : \mathbb{F}_q] = r$, where μ_ℓ is the group of ℓ-th roots of unity. Then, any embedding $k \to K$ takes $\mu_\ell \subset k^*$ to $\mu_\ell \subset K^*$, and the minimal polynomial of any primitive ℓ-th root of unity has degree exactly r.

Pinch's algorithm is very effective when $r = \varphi(\ell)$. Indeed in this case the ℓ-th cyclotomic polynomial Φ_ℓ is irreducible over $\mathbb{F}_q$, and its roots form a unique orbit under the action of the absolute Galois group of $\mathbb{F}_q$. Thus we can take any primitive ℓ-th roots of unity $\alpha \in k$ and $\beta \in K$ to describe the embedding.

In the general case, however, the roots of Φ_ℓ are partitioned in $\varphi(\ell)/r$ orbits, thus for two randomly chosen ℓ-th roots of unity $\zeta_1 \in k$ and $\zeta_2 \in K$, we can only say that there exists an exponent e such that

$$\alpha = \zeta_1 \mapsto \zeta_2^e = \beta$$

defines a valid embedding. Pinch's algorithm tests all possible exponents e, until a suitable one is found. To test for the validity of a given e, it applies the embedding $\phi : \zeta_1 \mapsto \zeta_2$ to the class of X in k, and verifies that its image is a root of f in K.

The trial-and-error nature of Pinch's algorithm makes it impractical, except for rare favorable cases where a *small* ℓ such that $r = \varphi(\ell)$ can be found. One possible workaround, suggested by Pinch himself, is to replace the group of roots of unity with a group of torsion points of a well chosen elliptic curve. We analyze this idea in greater detail in Section C.5.

This section is devoted to a different way of improving Pinch's algorithm, imagined by Rains [Rai96], and implemented in the Magma computer algebra system [BCP97]. Rains' technical contribution is twofold: first he replaces roots of unity with Gaussian periods to avoid trial-and-error, second he moves to slightly larger extension fields to ensure the existence of a small ℓ as above.

Uniquely defined orbits from Gaussian periods

For the rest of the section, we are going to assume that q is prime. The case where q is a higher power of a prime is discussed in Note C.4.

Suppose that we have an ℓ, coprime with q, such that $[\mathbb{F}_q(\mu_\ell) : \mathbb{F}_q] = r$, then the cyclotomic polynomial Φ_ℓ factors over $\mathbb{F}_q$ into $\varphi(\ell)/r$ distinct factors of degree r. Pinch's method, by choosing random roots of Φ_ℓ in k and K, randomly selects one of these factors as minimal polynomial. By combining the roots of Φ_ℓ into Gaussian periods, Rains' method uniquely selects a minimal polynomial of degree r.

Definition C.14. Let q be a prime, and let ℓ be a squarefree integer such that $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q \rangle \times S$ for some S. For any generator ζ_ℓ of μ_ℓ in $\mathbb{F}_q(\mu_\ell)$, define the Gaussian period $\eta_q(\zeta_\ell)$ as

$$\eta_q(\zeta_\ell) = \sum_{\sigma \in S} \zeta_\ell^\sigma. \tag{C.11}$$

It is evident from the definition that the Galois orbit of $\eta_q(\zeta_\ell)$ is independent of the initial choice of ζ_ℓ. Much less evident is the fact that this orbit has maximal size and forms a normal basis of $\mathbb{F}_q(\mu_\ell)$, as stated in the following lemma.

Lemma C.15. *Let q be a prime, and let ℓ be a squarefree integer such that $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q \rangle \times S$ for some S. The periods $\eta_q(\zeta_\ell^\tau)$ for τ running through $\langle q \rangle$ form a normal basis of $\mathbb{F}_q(\mu_\ell)$ over $\mathbb{F}_q$, independent of the choice of ζ_ℓ.*

Proof. Gaussian periods were introduced by Gauss [Gau86] in 1796 and extensively studied by Kummer [Kum46; Kum47b; Kum47a; Kum47c; Kum51; Kum55; Kum57]. The existence of integral normal bases for general Galois extensions without ramification was proved by Noether [Noe32]. For a generalized treatment specific to the case of finite fields, see [FGS99, Main Theorem]: the main idea of the proof is to show that cyclotomic units are normal in characteristic zero, then that integrality conditions carry normality through reduction modulo q. □

In what follows we are going to write $\eta(\zeta_\ell)$ when q is clear from the context.

Example C.16. Consider the extension $\mathbb{F}_8/\mathbb{F}_2$ of degree 3, which is generated by the 7-th roots of unity. We have a decomposition $(\mathbb{Z}/7\mathbb{Z})^\times = \langle 2 \rangle \times \langle -1 \rangle$, and the cyclotomic polynomial factors as

$$\Phi_7(X) = (X^3 + X + 1)(X^3 + X^2 + 1). \tag{C.12}$$

For any root ζ_7, we define the period

$$\eta_2(\zeta_7) = \zeta_7 + \zeta_7^{-1}. \tag{C.13}$$

The three periods $\eta_2(\zeta_7)$, $\eta_2(\zeta_7)^2$ and $\eta_2(\zeta_7)^4$ are all roots of the polynomial $x^3 + x^2 + 1$ and form a normal basis of $\mathbb{F}_8/\mathbb{F}_2$.

Rains' cyclotomic algorithm

The bottom-line of Rains' algorithm follows immediately from the previous section: given k, K and r,

1. find a *small* ℓ satisfying the conditions of Lemma C.15 with $[\mathbb{F}_q(\mu_\ell):\mathbb{F}_q]=r$;
2. take random ℓ-th roots of unity $\zeta_\ell \in k$ and $\zeta'_\ell \in K$;
3. return the Gaussian periods $\alpha_r = \eta(\zeta_\ell)$ and $\beta_r = \eta(\zeta'_\ell)$.

The problem with this algorithm is the vaguely defined *smallness* requirement on ℓ. Indeed the conditions of Lemma C.15 imply that ℓ divides $\Phi_r(q)$, thus in the worst case ℓ can be as large as $O(q^{\varphi(r)})$, which yields an algorithm of exponential complexity in the field size.

To circumvent this problem, Rains allows the algorithm to work in small auxiliary extensions of k and K, and then descend the results to k and K via a field trace. In other words, Rains' algorithm looks for ℓ such that $[\mathbb{F}_q(\mu_\ell):\mathbb{F}_q]=rs$ for some small s. We summarize this method in Algorithm 7; we only give the procedure for the field k, the procedure for the field K being identical.

Algorithm 7 Rains' cyclotomic algorithm

Input: A field extension $k/\mathbb{F}_q$ of degree r; a squarefree integer ℓ such that

- $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q\rangle \times S$ for some S,
- $\#\langle q\rangle = rs$ for some integer s;

a polynomial h of degree s irreducible over k.

Output: A normal generator of k over $\mathbb{F}_q$, with a uniquely defined Galois orbit.

1: Construct the field extension $k' = k[Z]/h(Z)$;
2: **repeat**
3: Compute $\zeta \leftarrow \theta^{(\#k'-1)/\ell}$ for a random $\theta \in k'$
4: **until** ζ is a primitive ℓ-th root of unity;
5: Compute $\eta(\zeta) \leftarrow \sum_{\sigma\in S}\zeta^\sigma$;
6: **return** $\alpha \leftarrow \mathrm{Tr}_{k'/k}\,\eta(\zeta) = \sum_{i=0}^{s-1}\eta(\zeta)^{q^{ri}}$.

Proposition C.17. *Algorithm 7 is correct. On input q,r,ℓ,s it computes its output using $O(sr^{(\omega+1)/2}\log(sr)+\mathsf{M}(sr)(\log(q)+(\ell/r)\log(\ell)))$ operations in $\mathbb{F}_q$ on average.*

Proof. By construction k' is isomorphic to $\mathbb{F}_q(\mu_\ell)$. By Lemma C.15 $\eta(\zeta)$ is a normal generator of k', and by [MP13, Prop. 5.2.3.1] α is a normal generator of k. This proves correctness.

According to Proposition C.1, computing ζ in Step 3 costs

$$O\big(\big(s^{(\omega+1)/2}\mathsf{M}(r)+sr^{(\omega+1)/2}+\mathsf{M}(sr)\log(\ell)\big)\log(sr)+\mathsf{M}(sr)\log(q)\big),$$

and the loop is executed $O(1)$ times on average. By observing that $s^{(\omega-1)/2}\in O(\ell/r)$, this fits into the stated bound.

Steps 5 and 6 can be performed at once by observing that

$$\alpha = \sum_{i=0}^{s-1}\eta(\zeta^{q^{ri}}) = \sum_{i=0}^{s-1}\sum_{\sigma\in S}\zeta^{q^{ri}\sigma}.$$

By reducing $q^{ri}\sigma$ modulo ℓ, we can compute this sum at the cost of $\varphi(\ell)/r$ exponentiations of degree at most ℓ in k', for a total cost of $O((\mathsf{M}(sr)(\ell/r)\log(\ell))$, using the techniques of Section C.2. The final result is obtained as an element of k. □

The attentive reader will have noticed the irreducible polynomial h of degree s given as input to Rains' algorithm. Computing this polynomial may be expensive. For a start, we may ask s to be coprime with r, so that h can be taken with coefficients in $\mathbb{F}_q$. Then, for small values of s and q, one may use a table of irreducible polynomials. For larger values, the constructions [CL13; DDS13; DDS14] are reasonably efficient, and yield an irreducible polynomial in time less than quadratic in s. However negligible from an asymptotic point of view, the construction of the polynomial h and of the field k' take a serious toll on the practical performances of Rains' algorithm.[4]

This concludes the presentation of Rains' algorithm. However, we are still left with a problem: how to find ℓ satisfying the conditions of the algorithm, and what bounds can be given on it. These questions will be analyzed in Section C.6.

Note: Rains' algorithm is easily extended to a non-prime field $\mathbb{F}_q$, as long as $q = p^d$ with $\gcd(d,r) = 1$. In this case, indeed, any generator of $\mathbb{F}_{p^r}$ over $\mathbb{F}_p$ is also a generator of $\mathbb{F}_{q^r}$ over $\mathbb{F}_q$. The algorithm is unchanged, except for the additional requirement that $\gcd(\varphi(\ell),d) = 1$, which ensures that the Gaussian periods indeed generate $\mathbb{F}_{p^r}$.

However, when $\gcd(d,r) \neq 1$, it is impossible to have $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q \rangle \times S$, so Rains' algorithm simply cannot be applied to this case. In the next section we are going to present a variant that does not suffer from this problem.

C.5 Elliptic Rains' algorithm

The Pinch/Rains' algorithm presented in the previous section relies on the use of the multiplicative group of finite fields. It is natural to try to extend it to other types of algebraic groups in order to cover a wider range of parameters. And indeed Pinch [Pin92] showed how to use torsion points of elliptic curves in place of roots of unity. Rains also considered this possibility, but did not investigate it thoroughly as no theoretical gain was to be expected. However, the situation in practice is quite different. In particular, the need for auxiliary extensions in the cyclotomic method is very costly, whereas the elliptic variant has naturally more chances to work in the base fields, and to be therefore very competitive.

In the next sections, we first introduce *elliptic periods*, a straightforward generalization of Gaussian periods for torsion points of elliptic curves, then analyze the cost of their computation. The main issue with this generalization is that, contrary to Gaussian periods, elliptic periods do not yield normal bases of finite fields. We still provide experimental data and heuristic arguments to support the benefit of using them. Whether they always yield an element generating the right field extension, a weak counterpart to Lemma C.15, is left as an open problem.

[4] A straightforward way to avoid these constructions consists in computing a factor h of the cyclotomic polynomial Φ_ℓ over the extension k following case 5 from Section C.2. Then, using Newton's identities, the period can be recovered from the logarithmic derivative of the reciprocal of h. Nevertheless, the cost of factoring Φ_ℓ renders this approach unpractical.

Uniquely defined orbits from elliptic periods

An elliptic curve E/L defined over a field L is given by an equation of the form

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \qquad \text{with } a_1, a_2, a_3, a_4, a_6 \in L.$$

For any field extension M/L the group of M-rational points of E is the set

$$E(M) = \{(x,y) \in M^2 \mid E(x,y) = 0\} \cup \{\mathcal{O}\}$$

endowed with the usual group law, where $\mathcal{O}$ is the point at infinity.

For an integer ℓ, we denote by $E[\ell]$ the ℓ-torsion subgroup of $E(\bar{L})$, where $\bar{L}$ denotes the algebraic closure of L. In this section we are going to consider integers ℓ coprime with the characteristic of L, then $E[\ell]$ is a group of rank 2.

For an elliptic curve $E/\mathbb{F}_q$ defined over a finite field, we denote by π its *Frobenius endomorphism*. It is well known that π satisfies a quadratic equation $\pi^2 - t\pi + q = 0$, where t is called the *trace of* E, and that this equation determines the cardinality of E as $\#E(\mathbb{F}_q) = q + 1 - t$.

Like in the cyclotomic case, the Frobenius endomorphism partitions $E[\ell]$ into orbits. Our goal is to take traces of points in $E[\ell]$ so that a uniquely defined orbit arises. This task is made more complex by the fact that $E[\ell]$ has rank 2, hence we are going to restrict to a family of primes ℓ named *Elkies primes*.

Definition C.18 (Elkies prime)**.** Let $E/\mathbb{F}_q$ be an elliptic curve, let ℓ be a prime number not dividing q. We say that ℓ is an Elkies prime for E if the characteristic polynomial of the Frobenius endomorphism π splits into two distinct factors over $\mathbb{Z}/\ell\mathbb{Z}$:

$$\pi^2 - t\pi + q = (\pi - \lambda)(\pi - \mu) \bmod \ell \qquad \text{with } \lambda \neq \mu. \tag{C.14}$$

Note that if ℓ is an Elkies prime for E, then $E[\ell]$ splits into two eigenspaces for π which are defined on extensions of $\mathbb{F}_q$ of degrees $\mathrm{ord}_\ell(\lambda)$ and $\mathrm{ord}_\ell(\mu)$. We are now ready to define the elliptic curve analogue of Gaussian periods.

Definition C.19. Let $E/\mathbb{F}_q$ be an elliptic curve of j-invariant not 0 or 1728.[5] Let $\ell > 3$ be an Elkies prime for E, λ an eigenvalue of π, and P a point of order ℓ in the eigenspace corresponding to λ (i.e., such that $\pi(P) = \lambda P$). Suppose that there is a subgroup S of $(\mathbb{Z}/\ell\mathbb{Z})^\times$ such that

$$(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle\lambda\rangle \times S. \tag{C.15}$$

Then we define an elliptic period as

$$\eta_{\lambda,S}(P) = \begin{cases} \sum_{\sigma \in S/\{\pm 1\}} x([\sigma]P) & \text{if } -1 \in S, \\ \sum_{\sigma \in S} x([\sigma]P) & \text{otherwise,} \end{cases} \tag{C.16}$$

where $x(P)$ denotes the abscissa of P.

Lemma C.20. *With the same notation as in Definition C.19, let*

$$\#\langle\lambda\rangle = \begin{cases} r & \text{if } -1 \notin \langle\lambda\rangle, \\ 2r & \text{otherwise.} \end{cases}$$

Then, for any point P in the eigenspace of λ, the period $\eta_{\lambda,S}(P)$ is in $\mathbb{F}_{q^r}$, and its minimal polynomial does not depend on the choice of P.

[5] The definition is easily extended to include $j = 0, 1728$: one must quotient S by $\mathrm{Aut}(E) \cap S$ and raise summands to an appropriate power. See [Bri+17] for more details.

Proof. By construction, the Frobenius endomorphism π acts on $\langle P \rangle$ as multiplication by the scalar λ. It is well known that two points have the same abscissa if and only if they are opposite, hence the Galois orbit of $x(P)$ has size r, and we conclude that both $x(P)$ and $\eta_{\lambda,S}(P)$ are in $\mathbb{F}_{q^r}$.

Now let $P' = [a]P$ be another point in the eigenspace of λ. By construction, $a = \pm\lambda^i\sigma$, for some $0 \leq i < r$ and some $\sigma \in S$. Hence $\eta_{\lambda,S}(P') = \eta_{\lambda,S}([\lambda^i]P)$, implying that $\eta_{\lambda,S}(P)$ and $\eta_{\lambda,S}(P')$ are conjugates in $\mathbb{F}_{q^r}$. □

We remark that the previous lemma only states that the elliptic periods $\eta_{\lambda,S}([\lambda^i]P)$ uniquely define an orbit inside $\mathbb{F}_{q^r}$, but gives no guarantee that they generate the whole $\mathbb{F}_{q^r}$. At this point, one would like to have an equivalent of Lemma C.15 for elliptic periods, i.e. that the elliptic period $\eta_{\lambda,S}(P)$ is a normal generator of $\mathbb{F}_q(x(P))$. However, it is easy to find non-normal elliptic periods, as the following example shows.

Example C.21. Let $E/\mathbb{F}_7$ be defined by $y^2 = x^3 + 5x + 4$, and consider the degree 3 extension of $\mathbb{F}_7$ defined by $k = \mathbb{F}_7[X]/(X^3 + 6X^2 + 4)$. Then

- $\ell = 31$ is an Elkies prime for E;
- the eigenvalues of the Frobenius modulo ℓ are $\lambda = 25$ of multiplicative order 3 and $\mu = 4$ of multiplicative order 5;
- $P = (5a^2 + 2a, 4)$ is a point of order 31 of E/k;
- $\eta = \eta_{\lambda,S}(P) = 5a^2 + 5a + 4$ is not a normal element, indeed $\eta + 4\eta^7 + 2\eta^{49} = 0$.

All known proofs of Lemma C.15 rely on the fact that the ℓ-th cyclotomic polynomial is irreducible over $\mathbb{Q}$, and its roots form a normal basis of $\mathbb{Q}(\zeta_\ell)$. This fails in the elliptic case: there is indeed no guarantee that the eigenspace of λ can be lifted to a normal basis over some number field.

Note however that, even if the elliptic period is not normal, it is enough for our purpose that it generates $\mathbb{F}_q(x(P))$ as a field, like in the example above. Experimental evidence suggests that this might always be the case. Thus, we state this as a conjecture.

Conjecture C.22. With the above notation, the elliptic period $\eta_{\lambda,S}(P)$ generates $\mathbb{F}_q(x(P))$ over $\mathbb{F}_q$.

If the conjecture is false, the only arguments we can give are of a heuristic nature. First and most simply, we can assume that the elliptic period behaves like a random element of $\mathbb{F}_q(x(P))$. In this case the chance of it not being a generator is approximately $1/q^r$. Based on this observation, numerous experiments were conducted for small values of q and r: for all primes ℓ up to 1000, and all primes q up to a bound depending on ℓ, we tested all curves defined over $\mathbb{F}_q$ such that ℓ is an Elkies prime and a corresponding eigenvalue r is an odd prime. Overall, more than 43 million curves were tested, and no counterexample was found. We also tested the conjecture through more involved methods using modular curves, more details are given in [Bri+17, Appendix C.2].[6]

Secondly, based on the polynomially cyclic algebras setting of [MV10], one can give a sufficient condition for the period to be a normal generator of $\mathbb{F}_q(x(P))$, that is a weak counterpart to Lemma C.15 (see [Bri+17, Appendix C.1]). Heuristically, this suggests that the chance of the period not being normal is approximately $1/q$.

We are now ready to present the generalization of Rains' algorithm, with the warning that the algorithm may fail, with low probability, if Conjecture C.22 is false.

[6] The source code for our tests is available at https://github.com/defeo/ffisom/blob/master/misc/conjdata/.

Elliptic variant of Rains' algorithm

Rain's cyclotomic algorithm needs auxiliary extensions to accommodate for sufficiently small subgroups μ_ℓ of the unit group. By replacing unit groups with torsion groups of elliptic curves, we gain more freedom on the choice of the size of the group, thus we are able to work with smaller fields.

The algorithm is very similar to Algorithm 7, and follows immediately from the previous section. For simplicity, we are going to state it only for r odd. Given k, K and r,

1. find a prime ℓ, an elliptic curve E, and an eigenvalue λ of the Frobenius endomorphism, satisfying the conditions of Definition C.19, and such that $\mathrm{ord}_\ell(\lambda) = r$;
2. take random points $P \in E(k)[\ell]$ and $P' \in E(K)[\ell]$ in the eigenspace of λ;
3. return the elliptic periods $\alpha := \eta_{\lambda,S}(P)$ and $\beta := \eta_{\lambda,S}(P')$.

Here we are faced with a difficulty: given E and λ it is easy to pick a random point in $E[\ell]$, but it is potentially much more expensive to compute a point in the eigenspace of λ. We will circumvent the problem by forcing $E(\mathbb{F}_{q^r})[\ell]$ to be of rank 1, and to coincide exactly with the eigenspace of λ. If we write $\mu = q/\lambda$ for the other eigenvalue of π, this is easily ensured by further asking that $\mathrm{ord}_\ell(\mu) \nmid r$.

We defer the discussion on the search for the elliptic curve E to Section C.6. Here we suppose that we are already given suitable parameters ℓ, E and λ, and analyze the last two steps of the algorithm, summarized below. We only give the procedure for k, the procedure for the field K being identical.

Algorithm 8 Elliptic Rain's algorithm

Input: A field extension $k/\mathbb{F}_q$ of odd degree r, an elliptic curve $E/\mathbb{F}_q$, its trace t, a prime ℓ not dividing q, an integer λ such that:

- $X^2 - tX + q = (X-\lambda)(X - q/\lambda) \mod \ell$,
- $\mathrm{ord}_\ell(\lambda) = r$, $\mathrm{ord}_\ell(q/\lambda) \nmid r$,
- $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle\lambda\rangle \times S$ for some S.

Output: A generator of k over $\mathbb{F}_q$, with a uniquely defined Galois orbit, or FAIL.

1: **repeat**
2: Compute $P \leftarrow [\#E(k)/\ell]Q$ for a random $Q \in E(k)$;
3: **until** $P \neq \mathcal{O}$;
4: Compute $\alpha \leftarrow \eta_{\lambda,S}(P)$;
5: **return** α if $k = \mathbb{F}_q(\alpha)$, FAIL otherwise.

Proposition C.23. *Algorithm 8 is correct. Assuming the heuristics about elliptic periods are correct, it fails with probability $\leq 1/q^r$. On input $r, q, E, t, \ell, \lambda$ it computes its output using $O(\mathsf{M}(r)(r\log(q) + (\ell/r)\log(\ell))$ operations in $\mathbb{F}_q$ on average, or $\tilde{O}(r^2\log(q))$ assuming $\ell \in o(r^2)$.*

Proof. Correctness follows immediately from Lemma C.20. Success probability comes from the assumption that $\eta_{\lambda,S}(P)$ behaves like a random element of $\mathbb{F}_q(x(P))$.

From the knowledge of the trace t, we immediately determine the zeta function of E, and hence the cardinality $\#E(k)$, at no algebraic cost.

To select the random point $Q \in E(k)$ we take a random element $x \in k$, then we verify that it is the abscissa of a point using a squareness test, at a costs of $O(r\mathsf{M}(r)\log(q))$ operations. Then, using Montgomery's formulas for scalar multiplication [Mon87], we can compute the points P and $[\ell]P$ without the knowledge of the ordinate of Q, at a cost of $O(r\mathsf{M}(r)\log(q))$ operations. A valid point is obtained after $O(1)$ tries on average.

The computation of the elliptic period α requires $O(\ell/r)$ scalar multiplications by an integer less than ℓ, for a total cost of $O((\mathsf{M}(r)(\ell/r)\log(\ell))$.

Finally, testing that α generates k is done by computing its minimal polynomial, at a cost of $O(r^{(\omega+1)/2})$ operations in $\mathbb{F}_q$ using [Sho93]. □

C.6 Algorithm selection

The algorithms presented in the previous sections have very similar complexities, and no one stands out as absolute winner. The complexity of all algorithms depends in a non-trivial way on the parameters q and r, and, for Rains' algorithms, on the search for a parameter ℓ and an associated elliptic curve.

This section studies the complexity of the embedding description problem from a global perspective: we explain how to find parameters for Rains' algorithms and criteria to choose the best among the embedding algorithms.

Given parameters $q = p^d$ and r, Rains' cyclotomic algorithm asks for a *small* parameter ℓ such that:

1. $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q \rangle \times S$ for some S,
2. $\langle q \rangle = rs$ for some integer s,
3. $\gcd(\varphi(\ell), d) = 1$ (see Note C.4).

Since r is a prime power, the second condition lets us take a prime power for ℓ too. Indeed if $\mathbb{Z}/\ell\mathbb{Z} \simeq \mathbb{Z}/\ell_1\mathbb{Z} \times \mathbb{Z}/\ell_2\mathbb{Z}$, then either $q \bmod \ell_1$ or $q \bmod \ell_2$ has order a multiple of r. Furthermore, if $\gcd(\ell, r) = 1$, then we can take ℓ prime, since higher powers would not help satisfy the conditions. On the other hand if $\gcd(\ell, r) \neq 1$, then the algorithms of Section C.3 have much better complexity. Hence we shall take ℓ prime.

Given the above constraints, we can rewrite the conditions as:

1. $\ell = rsv + 1$ for some s, v such that $\gcd(rs, v) = 1$,
2. $\mathrm{ord}_\ell(q) = rs$,
3. $\gcd(rsv, d) = 1$.

Remark C.24. Rains remarked that, when $q = 2$ and r is a power of 2 greater than 4, no ℓ can satisfy these constraints because 2 is a quadratic residue modulo any prime of the form $8u + 1$. This case, however, is covered by the Artin–Schreier technique in Section C.3, we thus ignore it.

In the elliptic algorithm we look for an integer ℓ and a curve $E/\mathbb{F}_q$ that satisfy the preconditions of Algorithm 8, i.e., such that

1. the Frobenius endomorphism π satisfies a characteristic equation

$$(\pi - \lambda)(\pi - \mu) = 0 \mod \ell,$$

2. $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle\lambda\rangle \times S$ for some S,
3. $\#\langle\lambda\rangle = r$, and
4. $\mu^r \neq 1 \mod \ell$.

As before, we only need to look at prime ℓ. Because $\mu = q/\lambda$, the last condition is equivalent to $q^r \neq 1 \bmod \ell$. Hence, we can restate the conditions on ℓ as

1. $\ell = ru+1$ for some u such that $\gcd(r,u) = 1$,
2. $q^r \neq 1 \mod \ell$.

Once ℓ is found, we compile a list of all integers of order r in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, and look for a curve of trace $t = \lambda + q/\lambda \bmod \ell$ for any λ in the list. Note, however, that for there to be such a curve, t must have a representative in the interval $[-2\sqrt{q}, 2\sqrt{q}]$.

We thus have a procedure to produce parameters for Rains' algorithms: test integers of the form $\ell = ur+1$ for increasing u, until a suitable one is found. We summarize the elliptic case in Algorithm 9.

Algorithm 9 Parameter selection for Rains' elliptic algorithm

Input: Prime powers q and r, a bound $\bar{u} \in o(\sqrt[4]{q}/r)$;
Output: An elliptic curve suitable for Rains' elliptic algorithm, or FAIL.

```
for u = 1 to ū do
  if gcd(u, r) = 1, and ℓ = ur + 1 is prime, and q^r ≠ 1 mod ℓ then
    Compute 𝒯 ← {λ + q/λ mod ℓ | ord_ℓ(λ) = r};
    for each curve E defined over 𝔽_q do
      Compute t = #E − q + 1;
      if (t mod ℓ) ∈ 𝒯 return (E, t);
    end for
  end if
end for
return FAIL.
```

Nevertheless, we are left with a question: when does the search for a suitable prime ℓ stop? It is not easy to give a precise answer: already the condition that $\ell = ur+1$ is prime poses some difficulties. Heuristically, we expect that about $u/\log(u)$ of those numbers are prime. However the best lower bound on primes of the form $\ell = ur+1$, even under GRH, is $\ell \in O(r^{2.4+\varepsilon})$ [Hea92]. Empirical data show that the reality is much closer to the heuristic bound: in Figure C.1 we plot for all prime powers $r < 10^8$ the smallest u such that $ur+1$ is prime. It appears that u is effectively bounded by $O(\log(r))$ for any practical purpose.

For the cyclotomic algorithm we also require that $\mathrm{ord}_\ell(q)$ is a multiple of r. Assuming that q is uniformly distributed in $(\mathbb{Z}/\ell\mathbb{Z})^\times$, this happens with probability at least $\varphi(r)/r \geq 1/2$, hence we can assume that asymptotically $\ell \in O(r\log(r))$.

The elliptic algorithm has a similar condition on ℓ, however it also requires that the set of curves with trace in $\mathscr{T}$ is not empty. In principle, it is enough to ask that $\ell < 4\sqrt{q}$; however, in order to have a good chance of finding such curves, we set an even more stringent bound $\ell \in o(\sqrt[4]{q})$. Indeed, although it is well known that traces are not evenly distributed modulo prime numbers [Len87], it is shown in [CH13, Th. 1] that the probability that the trace of a random curve is in $\mathscr{T}$ approaches $|\mathscr{T}|/\ell \sim r/\ell$, as ℓ

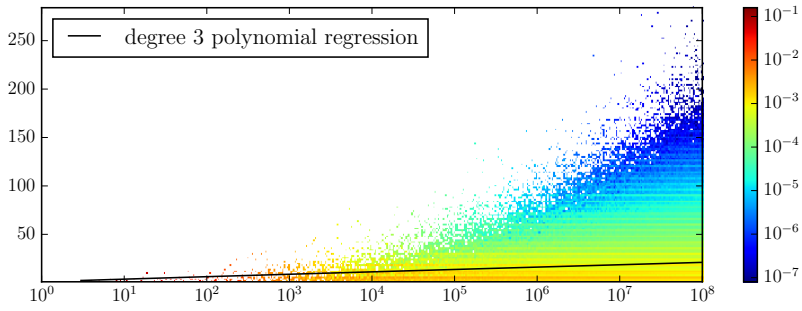


Figure C.1: Prime powers r (abscissa) versus smallest integer u (ordinate) such that $ur+1$ is prime. Abscissa in logarithmic scale, density normalized by $\log(x)/x$ and colored in logarithmic scale.

and q go to infinity, subject to $\ell \in o(\sqrt[4]{q})$. Thus, also in the elliptic case, we can assume that $\ell \in O(r\log(r))$.

Finally, we must also take into account the possibility that the elliptic algorithm fails on the curve output by Algorithm 9. Under the heuristics about the random distribution of elliptic periods, this possibility only discards one in $O(q^r)$ curves, and is thus negligible.

The cost of the search for ℓ is negligible compared to the cost of computing the embedding: for each candidate $\ell = ur+1$ we need to test its primality, and do some computations modulo ℓ. All this is well within $O(\sqrt{r})$ binary operations, using naive algorithms. In the elliptic case we also need to count the number of points of $O(\log(r))$ elliptic curves defined over $\mathbb{F}_q$, which can be done in $\tilde{O}\left(\log^5(q)\right)$ binary operations using the Schoof–Elkies–Atkin algorithm [Sch95; LS08b]. This highlights the fact that the elliptic algorithm is only practical for relatively small q.

Summarizing, we can expect heuristically to find a $\ell \in O(r\log(r))$ that satisfies all the constraints for the cyclotomic algorithm, leading to an expected running time of $\tilde{O}(r^{(\omega+1)/2} + \mathsf{M}(r)\log(q))$ operations in $\mathbb{F}_q$. Similarly, if we assume that $r\log(r) \in o(\sqrt[4]{q})$, we can expect to find suitable parameters for the elliptic algorithm, leading to an expected running time of $\tilde{O}(r^2\log(q))$ operations in $\mathbb{F}_q$, plus $\tilde{O}(\log^5(q))$ binary operations.

Although the complexity of the cyclotomic algorithm looks better, it must not be neglected that the $\tilde{O}$ notation hides the cost of taking an auxiliary extension of degree $O(\log(r))$; whereas the elliptic algorithm, when it applies, does not incur such overhead. The impact of the hidden terms in the complexity can be extremely important, as we will show in the next section.

The same considerations also apply when comparing Rains' algorithms to Allombert's. Indeed, the latter performs extremely well when the degree s of the auxiliary extension is small, but becomes slower as this degree increases.

In practice, it is hopeless to try and determine the appropriate bounds for each algorithm from a purely theoretical point of view. The best approach we can suggest, is to determine parameters at runtime, and set bounds and thresholds experimentally. To summarize, given parameters q and r, we suggest the following approach:

1. If $\gcd(q,r) \neq 1$, run the Artin–Schreier algorithm of Section C.3.
2. If r is a power of a small prime v, run the algorithm of Section C.3.

3. Determine the order s of q in $(\mathbb{Z}/r\mathbb{Z})^\times$. If it is small enough, run one of the variants of Allombert's algorithm presented in Section C.3.

4. Search for suitable parameters for Rains' algorithms. Depending on the best parameters found, run the best option among Rains' cyclotomic algorithm, Rains' elliptic algorithm, and Allombert's algorithm.

In the next section we shall focus on the last two steps, by comparing our implementations of the algorithms involved, thus giving an estimate of the various thresholds between them. However we stress that these thresholds are bound to vary depending on the implementation and the target platform, thus it is the implementer responsibility to determine them at the moment of configuring the system.

C.7 Experimental Results

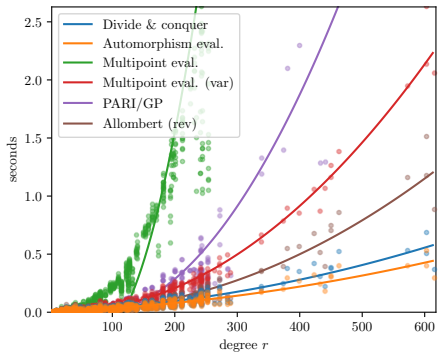
To validate our results, we implemented the algorithms described in the previous sections, and compared them to the implementation of Allombert's algorithm available in PARI/GP 2.8.0 [PARI], and to that of Rains' algorithm available in Magma 2.22-7 [BCP97]. The variants of Allombert's algorithm described in Section C.3 were implemented in C on top of the Flint library, version 2.5.2 [Har10]. Rains' cyclotomic and elliptic algorithms were implemented in SageMath, version 7.5.rc0 [Sage] (which itself uses PARI and Flint to implement finite fields), with critical code rewritten in C/Cython. Our code is limited to q a prime fitting in a machine-word, and m, n odd.

We make our code freely available under the open source MIT license, at `https://github.com/defeo/ffisom`. Instructions for compiling and running it are available online, as well as a ready-made cloud environment for testing it, hosted by the Binder service. We stress the fact that our code is experimental and only meant for research purposes; a subset of the authors is currently working on integrating part of it in a future release of Flint.

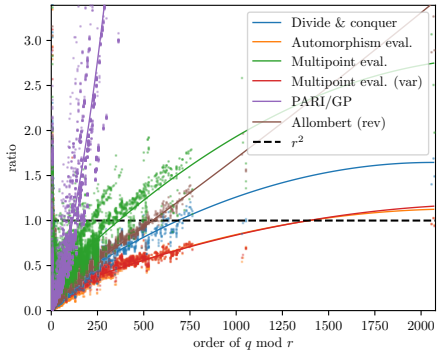
We ran tests for a wide range of primes q between 3 and $2^{60}+253$, and prime powers r between 3 and 2069. All tests were run on an Intel(R) Xeon(R) CPU E5-4650 v2 clocked at 2.40GHz. We report in Figure C.2 some statistics, exclusively on the runs for $100 < q < 2^{20}$; other ranges show very similar trends. The complete datasets, together with more plots and interactive visualizations are also hosted at `https://github.com/defeo/ffisom`.

We start by comparing our implementation of the three variants of Allombert's algorithm presented in Section C.3 with the original one in PARI. In Figure C.2a we plot running times against the extension degree r, only for cases where the auxiliary degree $s = \mathrm{ord}_q(r)$ is at most 10: dots represent individual runs, continuous lines represent degree 2 linear regressions. Analyzing the behavior for arbitrary auxiliary degree s is more challenging. Based on the observation that all variants have essentially quadratic cost in r, in Figure C.2b we take running times, we scale them down by r^2, and we plot them against the auxiliary degree s.

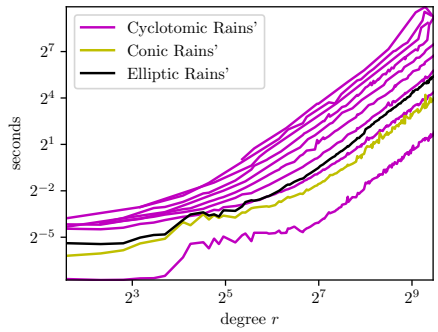
The first striking observation is the extremely poor performance of PARI, especially as s grows. To provide a fairer comparison, we re-implemented Allombert's revised algorithm [All02b], as faithfully as possible, as described in Section C.3; this is the curve labeled *"Allombert (rev)"* in the graphs. For completeness we also implemented the Paterson-Stockmeyer variant described previously; we do not plot it here, because it overlaps almost perfectly with our *"Divide & conquer"* curve. Although our re-



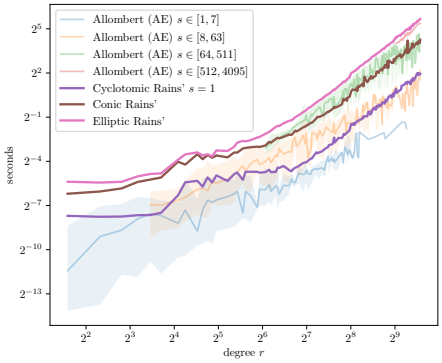
(a) Comparison of various implementations of Allombert's algorithm, in the case where the auxiliary degree $s = \mathrm{ord}_q(r) \leq 10$. Dots represent individual runs, lines represent degree 2 linear regressions.



(b) Comparison of various implementations of Allombert's algorithm, as a function of the auxiliary degree $s = \mathrm{ord}_q(r)$. Individual running times are scaled by down by r^2. Dots represent individual runs, lines represent degree 2 linear regressions.



(c) Cyclotomic, conic and elliptic variants of Rains' algorithm. Auxiliary extension degrees s for cyclotomic Rains' range between 1 and 9. Lines represent median times.



(d) Comparison of Allombert's (Automorphism evaluation variant) and Rains' algorithms at some fixed auxiliary extension degrees s. Lines represent median times, shaded areas minimum and maximum times.

Figure C.2: Benchmarks for Rains' and Allombert's algorithms. q is a prime between 100 and 2^{20}, r is an odd prime power varying between 3 and 2069. Plots c and d are in doubly logarithmic scale.

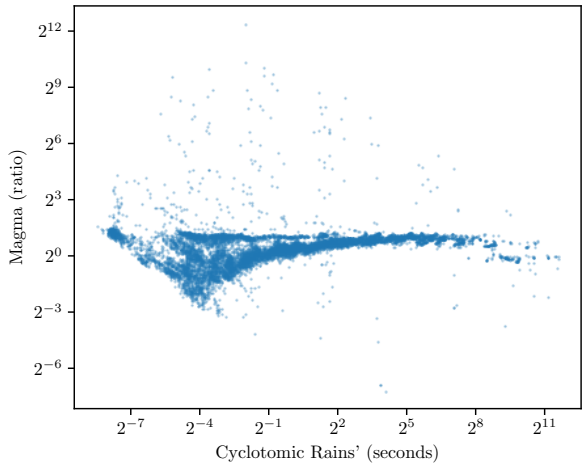


Figure C.3: Comparison of our implementation of Rains' algorithm and Magma's. Running time of our implementation in seconds vs ratio of Magma running time over ours. Plot in doubly logarithmic scale.

implementations are considerably faster than PARI, it is apparent that Allombert's original algorithm does not behave as well as our new variants.

Focusing now on our three new variants presented in Section C.3, one can't fail to notice that the second one, named *"Automorphism evaluation"*, beats the other two by a great margin, both for small and large auxiliary degree. Although the *"Multipoint evaluation"* approach is expected to eventually beat the other variants as s grows, the cross point seems to be extremely far from the parameters we explored. However, we notice that the naive variant of *"Multipoint evaluation"* not using the iterated Frobenius technique (labeled *"Multipoint evaluation (var)"* in the graphs), starts poorly, then quickly catches *"Automorphism evaluation"* as s grows.

Finally, comparing the variants as q grows shows[7] that the hierarchy between them is essentially unchanged as q approaches 2^{64}. We could not test larger values of q, as our code is limited to machine-word size.

Now we shift to Rains' algorithm and its variants. In comparing our implementation with Magma's, discarding outliers, we obtain a fairly consistent speed-up of about 30% (see Figure C.3); hence we will compare these algorithms only based on our timings. In Figure C.2c we group runs of the cyclotomic algorithm by the degree s of the auxiliary extension, and we plot median times against the degree r; only the graphs for $s < 10$ are shown in the figure. We observe a very large gap between $s = 1$ and larger s ($s = 2$ is $8 - 16$ times slower). This is partly due to the fact that we use generic Python code to construct auxiliary extensions, rather than dedicated C; however, a large gap is unavoidable, due to the added cost of computing in extension fields. We also plot median times for the elliptic variant and for the conic variant (see Appendix C.8). It is apparent that the elliptic algorithm outperforms the cyclotomic one as soon as $s \geq 3$, and that the conic algorithm conveniently replaces the case $s = 2$. Thus, at least for the parameter ranges we have tested, the cyclotomic algorithm with auxiliary extensions

[7] For lack of space we do not show these graphs here, but see https://github.com/defeo/ffisom/blob/master/notebooks/benchmarks.ipynb.

seems of limited interest. Plotting against the prime q instead of the degree r shows essentially the same behavior[8], however it should be reminded that the elliptic variant becomes very poor when q grows larger than a machine-word, because of the cost of point counting.

Finally, in Figure C.2d we compare Rains' algorithms against Allombert's. In light of the excellent performances of the *"Automorphism evaluation"* variant of Allombert's algorithm, we only plot the performances for this variant. We plot, against the degree r, runs of Allombert's algorithm grouped by ranges of the auxiliary degree $\mathrm{ord}_r(q)$: we shade the area between minimum and maximum running times, and trace the median time. We also take from Figure C.2c the graphs for the cyclotomic (only $s=1$), the conic and the elliptic variants of Rains' algorithm. We notice that Allombert's algorithm, even with relatively large auxiliary degrees, is extremely fast; the cyclotomic algorithm only beats it when $\mathrm{ord}_r(q)$ goes beyond 10 to 50, the conic algorithm only beats extremely large $\mathrm{ord}_r(q)$, and the elliptic algorithm is never better. We also observe that Allombert's algorithm has a better asymptotic behavior as the degree r grows.

In light of these comparisons, it seems that the absolute winner is our *Automorphism evaluation* variant of Allombert's algorithm, with Rains' cyclotomic algorithm being only occasionally more interesting. Obviously, the comparisons are only relevant to our own code and test conditions. Other implementations and benchmarks will likely find slightly different cross-points for the algorithms.

C.8 Rain's conic algorithm

We have seen that Rains' cyclotomic algorithm suffers in practice from the need to build a field extension k' of k. The conic variant we are going to present reduces the degree of the field extension from $s=[k':k]$ to $s/2$ whenever s is even. This is especially useful when $s=2$, as highlighted in Section C.7. The algorithm is similar in spirit to Williams' $p+1$ factoring method [Wil82], where the arithmetic of the norm 1 subgroup of k'^* is performed using Lucas sequences on a subfield of index 2 of k'.

Let $\mathbb{F}$ be a finite field of odd characteristic, let $\Delta\in\mathbb{F}$ be a quadratic non-residue, let δ be an element of the algebraic closure of $\mathbb{F}$ such that $\delta^2=\Delta$, and define the norm 1 subgroup of $\mathbb{F}[\delta]^*$ as

$$T_2(\mathbb{F})=\{(x+\delta y)/2 \mid x,y\in\mathbb{F} \text{ and } x^2-\Delta y^2=4\};$$

it is easy to verify that $T_2(\mathbb{F})$ forms a group under multiplication. If we see the elements $(x+\delta y)/2$ as points (x,y) on a conic $x^2-\Delta y^2=4$, the group law of $T_2(\mathbb{F})$ induces a group law on the conic. By projecting onto the x-coordinate, a straightforward calculation shows that, for any point $(\theta,*)$ on the conic, its n-th power has coordinates $(\theta_n,*)$, where θ_n is defined by the Lucas sequence

$$\theta_0=2,\quad \theta_1=\theta,\quad \theta_{i+1}=\theta\theta_i-\theta_{i-1}.$$

We shall denote by $[n]$ the map $\theta\mapsto\theta_n$; notice how it does not depend on the choice of Δ.

The generalization of Rains' algorithm is now obvious: by projecting on the x-coordinate, we work in a field extension twice as small compared to the original algorithm. This is summarized in Algorithm 10.

[8] Again, see https://github.com/defeo/ffisom/blob/master/notebooks/benchmarks.ipynb for a plot.

Algorithm 10 Rains' conic algorithm

Input: A field extension $k/\mathbb{F}_q$ of degree r; a prime ℓ such that

- $(\mathbb{Z}/\ell\mathbb{Z})^\times = \langle q \rangle \times S$ for some S,
- $\#\langle q \rangle = 2rs$ for some integer s;

a polynomial h of degree s irreducible over k.

Output: A normal generator of k over $\mathbb{F}_q$, with a uniquely defined Galois orbit.

1: Construct the field extension $k' = k[Z]/h(Z)$;
2: **repeat**
3: **repeat**
4: Take a random element $\theta \in k'$,
5: **until** $\theta^2 - 4$ is a quadratic non-residue;
6: Compute $\zeta = [(\#k' + 1)/\ell]\theta$,
7: **until** $\zeta \neq 2$;
8: Compute $\eta(\zeta) \leftarrow \sum_{\sigma \in S} [\sigma]\zeta$;
9: **return** $\alpha \leftarrow \mathrm{Tr}_{k'/k}\, \eta(\zeta) = \sum_{i=0}^{s-1} [q^{ri}]\eta(\zeta)$.

Proposition C.25. *Algorithm 10 is correct: on input q, r, ℓ, s it returns an element in the same Galois orbit as Algorithm 7 on input $q, r, \ell, 2s$. It computes its output using $O(\mathsf{M}(sr)(sr\log(q) + (\ell/r)\log(\ell)))$ operations in $\mathbb{F}_q$ on average, or $\tilde{O}((sr)^2\log(q))$ assuming $\ell \in o(sr^2)$.*

Proof. By construction, all the ℓ-th roots of unity are in $T_2(k')$. Observe that if $(x + \delta y)/2$ is in $T_2(k')$, then its trace over k' is equal to x. Hence, the value ζ computed in Step 6 is the trace over k' of a primitive ℓ-th root of unity. We conclude by comparing this algorithm with Algorithm 7.

The non-residuosity test in Step 5 is done by verifying that the $(\#k' - 1)/2$-th power of θ is equal to -1. We do this in $O(sr\log(q))$ operations in k', or $O(sr\mathsf{M}(sr)\log(q))$ operations in $\mathbb{F}_q$.

To implement the other steps, we need to evaluate the map $[n]$ efficiently. We have the following classical relationships for the Lucas sequence of θ:

$$\theta_{2i} = \theta_i^2 - 2, \quad \theta_{2i+1} = \theta_i\theta_{i+1} - \theta, \quad \theta_{2i+2} = \theta_{i+1}^2 - 2.$$

Starting with $\theta_0 = 2$ and $\theta_1 = \theta$, we use a binary scheme to deduce θ_i, θ_{i+1} from $\theta_{\lfloor i/2 \rfloor}, \theta_{\lfloor i/2 \rfloor + 1}$. We reach θ_n after $O(\log(n))$ steps, each requiring a constant number of operations in k'.

Hence, Step 6 costs $O(sr\mathsf{M}(sr)\log(q))$ operations in $\mathbb{F}_q$, while Steps 8 and 9 together cost $O((\mathsf{M}(sr)(\ell/r)\log(\ell))$. □

Although this variant does not exploit the asymptotic improvement offered by Proposition C.1, the fact that its auxiliary degree s is half the one of the original algorithm usually gives an interesting practical improvement. Step 6 can be modified so as to avoid the premature projection on the x-axis, so that the algorithms of Proposition C.1 apply. We leave the details of this variant to the reader.

C.9 References for "Computing isomorphisms and embeddings of finite fields"

[AL86] Leonard M. Adleman and Hendrik W. Lenstra. "Finding Irreducible Polynomials over Finite Fields". In: *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*. STOC '86. New York, NY, USA: ACM, 1986, pp. 350–355. ISBN: 0-89791-193-8. DOI: 10.1145/12130.12166.

[All02a] Bill Allombert. "Explicit Computation of Isomorphisms between Finite Fields". In: *Finite Fields and Their Applications* 8.3 (2002), pp. 332–342. DOI: 10.1006/ffta.2001.0344.

[All02b] Bill Allombert. *Explicit Computation of Isomorphisms between Finite Fields*. Revised version. 2002. URL: https://www.math.u-bordeaux.fr/~ballombe/fpisom.ps.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. "The MAGMA algebra system I: the user language". In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 235–265. ISSN: 0747-7171. DOI: 10.1006/jsco.1996.0125.

[BCS97a] Wieb Bosma, John Cannon, and Allan Steel. "Lattices of compatibly embedded finite fields". In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 351–369. ISSN: 0747-7171. DOI: 10.1006/jsco.1997.0138.

[BK78] Richard P. Brent and Hsiang Te Kung. "Fast Algorithms for Manipulating Formal Power Series". In: *Journal of the ACM* 25.4 (1978), pp. 581–595. ISSN: 0004-5411. DOI: 10.1145/322092.322099.

[Bri+17] Ludovic Brieulle, Luca De Feo, Javad Doliskani, Jean-Pierre Flori, and Éric Schost. "Computing isomorphisms and embeddings of finite fields (extended version)". In: *arXiv preprint arXiv:1705.01221* (2017). URL: https://arxiv.org/abs/1705.01221.

[CH13] Wouter Castryck and Hendrik Hubrechts. "The distribution of the number of points modulo an integer on elliptic curves over finite fields". In: *The Ramanujan Journal* 30.2 (2013), pp. 223–242.

[CK91] David G. Cantor and Erich Kaltofen. "On fast multiplication of polynomials over arbitrary algebras". In: *Acta Informatica* 28.7 (July 1991), pp. 693–701. ISSN: 0001-5903. DOI: 10.1007/BF01178683.

[CL08] Jean-Marc Couveignes and Reynald Lercier. "Galois invariant smoothness basis". In: *Series on Number Theory and Its Applications* 5 (May 2008). World Scientific, pp. 142–167.

[CL13] Jean-Marc Couveignes and Reynald Lercier. "Fast construction of irreducible polynomials over finite fields". In: *Israel Journal of Mathematics* 194.1 (2013), pp. 77–105.

[CZ81] David G Cantor and Hans Zassenhaus. "A New Algorithm for Factoring Polynomials over Finite Fields". In: *Mathematics of Computation* (1981), pp. 587–592.

[DDS13] Luca De Feo, Javad Doliskani, and Éric Schost. "Fast Algorithms for ℓ-adic Towers over Finite Fields". In: *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*. ISSAC '13. New York, NY, USA: ACM, 2013, pp. 165–172. ISBN: 978-1-4503-2059-7. DOI: 10.1145/2465506.2465956.

[DDS14] Luca De Feo, Javad Doliskani, and Éric Schost. “Fast Arithmetic for the Algebraic Closure of Finite Fields”. In: *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*. ISSAC ’14. New York, NY, USA: ACM, 2014, pp. 122–129. ISBN: 978-1-4503-2501-1. DOI: 10.1145/2608628.2608672.

[DS14] Javad Doliskani and Éric Schost. “Taking roots over high extensions of finite fields”. In: *Mathematics of Computation* 83.285 (2014), pp. 435–446.

[FGS99] Sandra Feisel, Joachim von zur Gathen, and M. Amin Shokrollahi. “Normal bases via general Gauss periods”. In: *Mathematics of Computation* 68.225 (1999), pp. 271–290.

[Gau86] Carl Friedrich Gauss. *Disquisitiones Arithmeticae*. Ed. by William C. Waterhouse. Springer-Verlag, 1986. ISBN: 9783540962540. URL: https://books.google.fr/books?id=Y-49PgAACAAJ.

[GG99] Joachim von zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. New York, NY, USA: Cambridge University Press, 1999. ISBN: 0-521-64176-4.

[GS92] Joachim von zur Gathen and Victor Shoup. “Computing Frobenius Maps and Factoring Polynomials”. In: *Computational Complexity* 2 (1992), pp. 187–224.

[Har09] David Harvey. “Faster polynomial multiplication via multipoint Kronecker substitution”. In: *Journal of Symbolic Computation* 44.10 (Oct. 2009), pp. 1502–1510. ISSN: 07477171. DOI: 10.1016/j.jsc.2009.05.004.

[Har10] William B. Hart. “Fast Library for Number Theory: An Introduction”. In: *Proceedings of the Third International Congress on Mathematical Software*. ICMS’10. Kobe, Japan: Springer-Verlag, 2010, pp. 88–91. URL: http://flintlib.org/.

[Hea92] David R. Heath-Brown. “Zero-free regions for Dirichlet L-functions, and the least prime in an arithmetic progression”. In: *Proceedings of the London Mathematical Society*. Vol. 64. 2. 1992, pp. 265–338.

[Kal87] Erich Kaltofen. “Computer algebra algorithms”. In: *Annual Review in Computer Science* 2 (1987), pp. 91–118. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/87/Ka87_annrev.pdf.

[KS97] Erich Kaltofen and Victor Shoup. “Fast polynomial factorization over high algebraic extensions of finite fields”. In: *ISSAC ’97: Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*. New York, NY, USA: ACM, 1997, pp. 184–188. ISBN: 0-89791-875-4. DOI: 10.1145/258726.258777.

[KS98] Erich Kaltofen and Victor Shoup. “Subquadratic-time factoring of polynomials over finite fields”. In: *Mathematics of Computation* 67.223 (1998), pp. 1179–1197. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-98-00944-2.

[KU11] Kiran S Kedlaya and Christopher Umans. “Fast polynomial factorization and modular composition”. In: *SIAM Journal on Computing* 40.6 (2011), pp. 1767–1802.

[Kum46] Ernst Eduard Kummer. "Über die Divisoren gewisser Formen der Zahlen, welche aus der Theorie der Kreistheilung entstehen". German. In: *Journal für die reine und angewandte Mathematik* 30 (1846), pp. 107–116. URL: http://eudml.org/doc/147278.

[Kum47a] Ernst Eduard Kummer. "Sur les nombres complexes qui sont formés avec les nombres entiers réels et les racines de l'unité". French. In: *Journal de Mathématiques Pures et Appliquées* (1847), pp. 185–212. URL: http://eudml.org/doc/235075.

[Kum47b] Ernst Eduard Kummer. "Über die Zerlegung der aus Wurzeln der Einheit gebildeten complexen Zahlen in ihre Primfactoren". German. In: *Journal für die reine und angewandte Mathematik* 35 (1847), pp. 327–367. URL: http://eudml.org/doc/147394.

[Kum47c] Ernst Eduard Kummer. "Zur Theorie der complexen Zahlen". German. In: *Journal für die reine und angewandte Mathematik* 35 (1847), pp. 319–326. URL: http://eudml.org/doc/147393.

[Kum51] Ernst Eduard Kummer. "Mémoire sur la théorie des nombres complexes composés de racines de l'unité et de nombres entiers". French. In: *Journal de Mathématiques Pures et Appliquées* (1851), pp. 377–498. URL: http://eudml.org/doc/235621.

[Kum55] Ernst Eduard Kummer. "Über eine besondere Art, aus complexen Einheiten gebildeter Ausdrücke". German. In: *Journal für die reine und angewandte Mathematik* 50 (1855), pp. 212–232. URL: http://eudml.org/doc/147605.

[Kum57] Ernst Eduard Kummer. "Über die den Gaußschen Perioden der Kreistheilung entsprechenden Congruenzwurzeln". German. In: *Journal für die reine und angewandte Mathematik* 53 (1857), pp. 142–148. URL: http://eudml.org/doc/147659.

[Lan02] Serge Lang. *Algebra*. 3rd edition. Springer, Jan. 2002. ISBN: 038795385X.

[Le 14] François Le Gall. "Powers of Tensors and Fast Matrix Multiplication". In: *ISSAC'14*. ACM, 2014, pp. 296–303.

[Len87] Hendrik W. Lenstra. "Factoring integers with elliptic curves". In: *Annals of Mathematics* 126 (1987), pp. 649–673.

[Len91] Hendrik W. Lenstra. "Finding isomorphisms between finite fields". In: *Mathematics of Computation* 56.193 (1991), pp. 329–347.

[LS08b] Reynald Lercier and Thomas Sirvent. "On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field". In: *Journal de théorie des nombres de Bordeaux* 20.3 (2008), pp. 783–797. URL: http://perso.univ-rennes1.fr/reynald.lercier/file/LS08.pdf.

[Moe76] Robert T. Moenck. "Another polynomial homomorphism". In: *Acta Informatica* 6.2 (June 1976), pp. 153–169. ISSN: 0001-5903. DOI: 10.1007/BF00268498.

[Mon87] Peter L. Montgomery. "Speeding the Pollard and Elliptic Curve Methods of Factorization". In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. ISSN: 00255718. DOI: 10.2307/2007888.

[MP13] Gary L. Mullen and Daniel Panario. *Handbook of finite fields*. CRC Press, 2013.

[MV10] Preda Mihailescu and Victor Vuletescu. “Elliptic Gauss sums and applications to point counting”. In: *Journal of Symbolic Computation* 45.8 (2010), pp. 825–836. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2010.01.004.

[Nar18] Anand Kumar Narayanan. “Fast Computation of Isomorphisms Between Finite Fields Using Elliptic Curves”. In: *International Workshop on the Arithmetic of Finite Fields, WAIFI 2018*. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2018.

[Noe32] Emmy Noether. “Normalbasis bei Körpern ohne höhere Verzweigung”. In: *Journal für die reine und angewandte Mathematik* 167 (1932), pp. 147–152. URL: http://eudml.org/doc/149800.

[PARI] *PARI/GP, version 2.8.0*. The PARI Group. Bordeaux, 2016. URL: https://pari.math.u-bordeaux.fr/.

[Pin92] Richard G. E. Pinch. “Recognising Elements Of Finite Fields”. In: *Cryptography and Coding II*. Oxford University Press, 1992, pp. 193–197.

[PS06] Cyril Pascal and Éric Schost. “Change of order for bivariate triangular sets”. In: *ISSAC ’06: Proceedings of the 2006 international symposium on Symbolic and algebraic computation*. New York, NY, USA: ACM, 2006, pp. 277–284. ISBN: 1-59593-276-3. DOI: 10.1145/1145768.1145814.

[PS13a] Adrian Poteaux and Éric Schost. “Modular Composition Modulo Triangular Sets and Applications”. In: *Computational Complexity* 22.3 (2013), pp. 463–516.

[PS73] Michael S. Paterson and Larry J. Stockmeyer. “On the Number of Nonscalar Multiplications Necessary to Evaluate Polynomials”. In: *SIAM Journal on Computing* 2.1 (1973), pp. 60–66. DOI: 10.1137/0202007.

[Rai96] Eric M. Rains. “Efficient Computation of Isomorphisms Between Finite Fields”. Personal communication. 1996.

[Sage] *SageMath, the Sage Mathematics Software System (Version 8.0)*. The Sage Developers. 2018. URL: https://www.sagemath.org.

[Sch95] René Schoof. “Counting points on elliptic curves over finite fields”. In: *Journal de Théorie des Nombres de Bordeaux* 7.1 (1995), pp. 219–254. URL: http://www.ams.org/mathscinet-getitem?mr=1413578.

[Sho90] Victor Shoup. “New algorithms for finding irreducible polynomials over finite fields”. In: *Mathematics of Computation* 54.189 (Jan. 1990), pp. 435–435. DOI: 10.1090/s0025-5718-1990-0993933-0.

[Sho93] Victor Shoup. “Fast construction of irreducible polynomials over finite fields”. In: *SODA ’93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1993, pp. 484–492. ISBN: 0-89871-313-7.

[Sho94b] Victor Shoup. “Fast construction of irreducible polynomials over finite fields”. In: *Journal of Symbolic Computation* 17.5 (1994), pp. 371–391. ISSN: 0747-7171. DOI: 10.1006/jsco.1994.1025.

[Sho99] Victor Shoup. “Efficient Computation of Minimal Polynomials in Algebraic Extensions of Finite Fields”. In: *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*. ISSAC ’99. New York, NY, USA: ACM, 1999, pp. 53–58. ISBN: 1-58113-073-2. DOI: 10.1145/309831.309859.

[Wil82] Hugh C. Williams. “A $p+1$ method of factoring”. In: *Mathematics of Computation* 39.159 (1982), pp. 225–234.

D Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies

Abstract

We present new candidates for quantum-resistant public-key cryptosystems based on the conjectured difficulty of finding isogenies between supersingular elliptic curves. The main technical idea in our scheme is that we transmit the images of torsion bases under the isogeny in order to allow the parties to construct a shared commutative square despite the noncommutativity of the endomorphism ring. We give a precise formulation of the necessary computational assumptions along with a discussion of their validity, and prove the security of our protocols under these assumptions. In addition, we present implementation results showing that our protocols are multiple orders of magnitude faster than previous isogeny-based cryptosystems over ordinary curves.

This paper is an extended version of [JD11]. We add a new zero-knowledge identification scheme, and detailed security proofs for the protocols. We also present a new, asymptotically faster, algorithm for key generation, a thorough study of its optimization, and new experimental data.

D.1 Introduction

The Diffie-Hellman scheme is a fundamental protocol for public-key exchange between two parties. Its original definition over finite fields is based on the hardness of computing the map $g, g^a, g^b \mapsto g^{ab}$ for $g \in \mathbb{F}_p^*$. Recently, Stolbunov [Sto10] proposed a Diffie-Hellman type system based on the difficulty of computing isogenies between ordinary elliptic curves, with the stated aim of obtaining quantum-resistant cryptographic protocols. The fastest known (classical) probabilistic algorithm for solving this problem is the algorithm of Galbraith and Stolbunov [GS13], based on the algorithm of Galbraith, Hess, and Smart [GHS02]. This algorithm is exponential, with a worst-case running time of $O(\sqrt[4]{q})$. However, on a quantum computer, recent work of Childs et al. [CJS14] showed that the private keys in Stolbunov's system can be recovered in subexponential time. Moreover, even if we only use classical attacks in assessing security levels, Stolbunov's scheme requires 229 seconds (even with precomputation) to perform one key exchange operation at the 128-bit security level on a desktop PC [Sto10, Table 1].

In this work we present isogeny-based key-exchange, encryption, and identification schemes that address both the performance and security drawbacks of Stolbunov's system. Our primitive achieves performance on the order of 60 milliseconds (cf. Section D.7) at the 128-bit security level (as measured against the fastest known quantum attacks) using desktop PCs, making our schemes far faster than Stolbunov's. In terms of security, our schemes are not vulnerable to the algorithm of Childs et al. [CJS14], nor to any algorithm of this type, since they are not based on a group action. The fastest known attacks against our schemes, even on quantum computers, require fully exponential time. Our schemes involve new computational assumptions upon which their quantum resistance is based, and like all new computational assumptions, further study and the passage of time is needed for validation. Nevertheless, we believe our proposal represents a promising candidate for quantum-resistant isogeny-based public-key cryptography.

Our proposal, presented in Section D.3, uses isogenies between *supersingular* elliptic curves rather than ordinary elliptic curves. The main technical difficulty is

that, in the supersingular case, the endomorphism ring is noncommutative, whereas Diffie-Hellman type protocols require commutativity. We show how to overcome this obstacle by providing the outputs of the isogeny on certain points as auxiliary input to the protocols. To the best of our knowledge, nothing similar to this idea has ever previously appeared in the literature. Providing this auxiliary input does not seem to make the problem of finding isogenies any easier, and the added difficulty arising from noncommutativity seems to make our protocols stronger; see Section D.5 for a full discussion. The multiple orders of magnitude of performance gains in our scheme arise from the fact that supersingular isogeny graphs are much faster to navigate than ordinary graphs, as described in Section D.4. In Sections D.5 and D.6 we provide formal statements of the hardness assumptions and security reductions for our system. Finally, in Section D.7 we present implementation results confirming the correctness and performance of our protocol.

D.2 Isogenies

Let E_1 and E_2 be elliptic curves defined over a finite field $\mathbb{F}_q$. An isogeny $\phi : E_1 \to E_2$ defined over $\mathbb{F}_q$ is a non-constant rational map defined over $\mathbb{F}_q$ which is also a group homomorphism from $E_1(\mathbb{F}_q)$ to $E_2(\mathbb{F}_q)$ [Sil92, p. III.4]. The degree of an isogeny is its degree as a rational map. For separable isogenies, having degree ℓ implies that the kernel of the isogeny has cardinality ℓ. Every isogeny of degree greater than 1 can be factored into a composition of isogenies of prime degree over $\bar{\mathbb{F}}_q$ [Cou06].

An endomorphism of an elliptic curve E defined over $\mathbb{F}_q$ is an isogeny $E \to E$ defined over $\mathbb{F}_{q^m}$ for some m. The set of endomorphisms of E together with the zero map forms a ring under the operations of pointwise addition and composition; this ring is called the endomorphism ring of E and denoted $\mathrm{End}(E)$. The ring $\mathrm{End}(E)$ is isomorphic either to an order in a quaternion algebra or to an order in an imaginary quadratic field [Sil92, p. V.3.1]; in the first case we say E is supersingular and in the second case we say E is ordinary.

Two elliptic curves E_1 and E_2 defined over $\mathbb{F}_q$ are said to be isogenous over $\mathbb{F}_q$ if there exists an isogeny $\phi : E_1 \to E_2$ defined over $\mathbb{F}_q$. A theorem of Tate states that two curves E_1 and E_2 are isogenous over $\mathbb{F}_q$ if and only if $\#E_1(\mathbb{F}_q) = \#E_2(\mathbb{F}_q)$ [Tat66, §3]. Since every isogeny has a dual isogeny [Sil92, p. III.6.1], the property of being isogenous over $\mathbb{F}_q$ is an equivalence relation on the finite set of $\bar{\mathbb{F}}_q$-isomorphism classes of elliptic curves defined over $\mathbb{F}_q$. Accordingly, we define an isogeny class to be an equivalence class of elliptic curves, taken up to $\bar{\mathbb{F}}_q$-isomorphism, under this equivalence relation.

The ℓ-torsion group of E, denoted $E[\ell]$, is the set of all points $P \in E(\bar{\mathbb{F}}_q)$ such that ℓP is the identity. For ℓ such that $p \nmid \ell$, we have $E[\ell] \cong \mathbb{Z}/\ell\mathbb{Z} \oplus \mathbb{Z}/\ell\mathbb{Z}$.

Curves in the same isogeny class are either all supersingular or all ordinary. We assume for the remainder of this paper that we are in the supersingular case. Because we only care about curves up to isomorphism, and every supersingular curve is isomorphic to a curve defined over the field $\mathbb{F}_{p^2}$, we limit ourselves to this base field.

For every prime $\ell \nmid p$, there exist exactly $\ell + 1$ isogenies (counting multiplicities) of degree ℓ originating from any given such supersingular curve. Given an elliptic curve E and a finite subgroup Φ of E, there is up to isomorphism a unique isogeny $E \to E'$ having kernel Φ [Sil92, p. III.4.12]. Hence we can identify an isogeny by specifying its kernel, and conversely given a kernel subgroup the corresponding isogeny can be computed using Vélu's formulas [Vél71]. Typically, this correspondence is of little use, since the kernel, or any representation thereof, is usually as unwieldy as the isogeny itself.

However, in the special case of kernels generated by $\mathbb{F}_{p^2}$-rational points of smooth order, specifying a generator of the kernel allows for compact representation and efficient computation of the corresponding isogeny, as we demonstrate in Section D.4.

Ramanujan graphs

Let $G = (\mathscr{V}, \mathscr{E})$ be a finite graph on h vertices $\mathscr{V}$ with undirected edges $\mathscr{E}$. Suppose G is a regular graph of degree k, i.e., exactly k edges meet at each vertex. Given a labeling of the vertices $\mathscr{V} = \{v_1, \ldots, v_h\}$, the adjacency matrix of G is the symmetric $h \times h$ matrix A whose ij-th entry $A_{i,j} = 1$ if an edge exists between v_i and v_j and 0 otherwise.

It is convenient to identify functions on $\mathscr{V}$ with vectors in $\mathbb{R}^h$ via this labeling, and therefore also think of A as a self-adjoint operator on $L^2(\mathscr{V})$. All of the eigenvalues of A satisfy the bound $|\lambda| \leq k$. Constant vectors are eigenfunctions of A with eigenvalue k, which for obvious reasons is called the trivial eigenvalue λ_{triv}. A family of such graphs G with $h \to \infty$ is said to be a sequence of *expander graphs* if all other eigenvalues of their adjacency matrices are bounded away from $\lambda_{\text{triv}} = k$ by a fixed amount.[1] In particular, no other eigenvalue is equal to k; this implies the graph is connected. A Ramanujan graph is a special type of expander which has $|\lambda| \leq 2\sqrt{k-1}$ for any nontrivial eigenvalue which is not equal to $-k$ (this last possibility happens if and only if the graph is bipartite). The Ramanujan property was first defined in [LPS88]. It characterizes the optimal separation between the two largest eigenvalues of the graph adjacency matrix, and implies the expansion property.

A fundamental use of expanders is to prove the rapid mixing of the random walk on $\mathscr{V}$ along the edges $\mathscr{E}$. The following rapid mixing result is standard but we present it below for completeness. For the proof, see [JMV09] or [DSV03; Lub94; Sar90].

Proposition D.1. *Let G be a regular graph of degree k on h vertices. Suppose that the eigenvalue λ of any nonconstant eigenvector satisfies the bound $|\lambda| \leq c$ for some $c < k$. Let S be any subset of the vertices of G, and x be any vertex in G. Then a random walk of length at least $\frac{\log 2h/|S|^{1/2}}{\log k/c}$ starting from x will land in S with probability at least $\frac{|S|}{2h} = \frac{|S|}{2|G|}$.*

Isogeny graphs

An isogeny graph is a graph whose nodes consist of all elliptic curves in $\mathbb{F}_q$ belonging to a fixed isogeny class, up to $\bar{\mathbb{F}}_q$-isomorphism (so that two elliptic curves which are isomorphic over $\bar{\mathbb{F}}_q$ represent the same node in the graph). In practice, the nodes are represented using j-invariants, which are invariant up to isomorphism. Isogeny graphs for supersingular elliptic curves were first considered by Mestre [Mes86], and were shown by Pizer [Piz90; Piz98] to have the Ramanujan property.

Every supersingular elliptic curve in characteristic p is defined over either $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$ [Sil92], so it suffices to fix $\mathbb{F}_q = \mathbb{F}_{p^2}$ as the field of definition for this discussion. Thus, in contrast to ordinary curves, there are a finite number of isomorphism classes of supersingular curves in any given isogeny class; this number is in fact $g+1$, where g is the genus of the modular curve $X_0(p)$, which is roughly $p/12$. It turns out that all supersingular curves defined over $\mathbb{F}_{p^2}$ belong to the same isogeny class [Mes86]. For a fixed prime value of $\ell \neq p$, we define the vertices of the supersingular isogeny

[1] Expansion is usually phrased in terms of the number of neighbors of subsets of G, but the spectral condition here is equivalent for k-regular graphs and also more useful for our purposes.

graph $\mathscr{G}$ to consist of these g isomorphism classes of curves, with edges given by isomorphism classes of degree-ℓ isogenies, defined as follows: two isogenies $\phi_1, \phi_2 : E_i \to E_j$ are isomorphic if there exists an automorphism $\alpha \in \mathrm{Aut}(E_j)$ (i.e., an invertible endomorphism) such that $\phi_2 = \alpha\phi_1$. Pizer [Piz90; Piz98] has shown that $\mathscr{G}$ is a connected $k = \ell + 1$-regular multigraph satisfying the Ramanujan bound of $|\lambda| \leq 2\sqrt{\ell} = 2\sqrt{k-1}$ for the nontrivial eigenvalues of its adjacency matrix.

D.3 Public-key cryptosystems based on supersingular curves

In this section we present a key-exchange protocol and a public-key cryptosystem analogous to those of [RS06; Sto10], and a zero-knowledge identification scheme, all using supersingular elliptic curves.

Our protocols require supersingular curves of smooth order. Such curves are normally unsuitable for cryptography since discrete logarithms on them are easy. However, since the discrete logarithm problem is unimportant in our setting, this issue does not affect us. In the supersingular setting, it is easy to construct curves of smooth order, and using a smooth order curve will give a large number of isogenies that are fast to compute. Specifically, we fix $\mathbb{F}_q = \mathbb{F}_{p^2}$ as the field of definition, where p is a prime of the form $\ell_A^{e_A}\ell_B^{e_B} \cdot f \pm 1$. Here ℓ_A and ℓ_B are small primes, and f is a cofactor such that p is prime. Then we construct a supersingular curve E defined over $\mathbb{F}_q$ of cardinality $(\ell_A^{e_A}\ell_B^{e_B} f)^2$. By construction, $E[\ell_A^{e_A}]$ is $\mathbb{F}_q$-rational and contains $\ell_A^{e_A-1}(\ell_A+1)$ cyclic subgroups of order $\ell_A^{e_A}$, each defining a different isogeny; the analogous statement holds for $E[\ell_B^{e_B}]$.

Our protocols revolve around the following commutative diagram

$$\begin{array}{ccc} E & \xrightarrow{\phi} & E/\langle P\rangle \\ {\scriptstyle\psi}\downarrow & & \downarrow \\ E/\langle Q\rangle & \longrightarrow & E/\langle P,Q\rangle \end{array} \tag{D.1}$$

where ϕ and ψ are random walks in the graphs of isogenies of degrees ℓ_A and ℓ_B respectively. Their security is based on the difficulty of finding a path connecting two given vertices in a graph of supersingular isogenies. We refer to Section D.4 for low-level algorithmic details, and Section D.5 for a full discussion of security.

Zero-knowledge proof of identity

We begin with the protocol which is easiest to understand. Peggy knows a cyclic degree $\ell_A^{e_A}$ isogeny $\phi : E \to E/\langle S\rangle$, with the curves E and $E/\langle S\rangle$ publicly known, and wants to prove to Vic that she knows a generator for $\langle S\rangle$, without revealing it.

Our protocol is loosely inspired by the zero-knowledge proof of membership for Graph Isomorphism [GMW91]. In that protocol, Peggy shows that she knows a graph isomorphism $G \simeq G'$ by first publishing a random H such that the following diagram commutes

$$\begin{array}{ccc} G & \longleftrightarrow & G' \\ & {\scriptstyle\phi}\nwarrow \quad \nearrow{\scriptstyle\psi} & \\ & H & \end{array} \tag{D.2}$$

and then revealing only one among ϕ and ψ. Intuitively, this protocol is *perfectly* zero-knowledge because the information that Peggy reveals (i.e., a random permutation of G or G') could be easily computed by anyone without her help.

In an analogous way, our protocol consists in publishing the vertices of diagram (D.1), and then revealing some, but not all, of its arrows. Unlike the case of Graph Isomorphism, in our protocol Peggy needs to use her secret knowledge to create the diagram, thus we cannot achieve a *perfect* zero-knowledge. Nevertheless, we will show in Section D.6 that, under suitable assumptions, our protocol is *computationally* zero-knowledge.

We show below the diagram used in our protocol. $\langle S \rangle$ is the kernel of the secret isogeny ϕ of degree $\ell_A^{e_A}$, while $\langle R \rangle$ is a cyclic group of order $\ell_B^{e_B}$.

$$\begin{array}{ccc} E & \xrightarrow{\phi} & E/\langle S \rangle \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle\psi'} \\ E/\langle R \rangle & \xrightarrow{\phi'} & E/\langle S, R \rangle \end{array} \tag{D.3}$$

Peggy can compute the diagram as follows:

- She uses Vélu's formulas to compute the isogeny $\psi : E \to E/\langle R \rangle$;
- She computes $R' = \phi(R)$ and the isogeny $\psi' : E/\langle S \rangle \to E/\langle S, R \rangle$;
- She computes $S' = \psi(S)$ and the isogeny $\phi' : E/\langle R \rangle \to E/\langle S, R \rangle$.

Now, the natural question is: which arrows of the diagram can Peggy reveal without compromising her secret ϕ? It is not hard to see, and we will show it in Theorem D.15, that the knowledge of (ψ, ϕ') or (ψ', ϕ') allows anyone to compute the kernel of ϕ. However, we will argue that there is no obvious way to compute ϕ from the sole knowledge of ϕ'. Revealing one of ψ or ψ' is no problem either, however revealing (ψ, ψ') altogether is more subtle. Indeed, revealing the points R and $\phi(R)$ uncovers some information on the action of ϕ on $E[\ell_B^{e_B}]$: it is to be expected that after a few iterations Peggy will reveal a basis (P, Q) of $E[\ell_B^{e_B}]$ and the respective images $\phi(P)$, $\phi(Q)$, thus allowing anyone to evaluate ϕ on the whole $E[\ell_B^{e_B}]$. Nevertheless, we conjecture that this leakage does not compromise Peggy's secret either, and we make these data part of the public parameters.[2]

Finally, we present our protocol.

Secret parameters A supersingular curve E defined over $\mathbb{F}_q$ and a primitive $\ell_A^{e_A}$-torsion point S defining an isogeny $\phi : E \to E/\langle S \rangle$.

Public parameters The curves E and $E/\langle S \rangle$. Generators P, Q of $E[\ell_B^{e_B}]$ and their images $\phi(P), \phi(Q)$.

Identification Repeat m times:

1. Peggy chooses a random primitive $\ell_B^{e_B}$-torsion point R and computes diagram (D.3).

[2] An alternative solution, that intuitively leaks less information on ϕ, would be to publish random generators of $\langle R \rangle$ and $\langle \phi(R) \rangle$. However, it is not clear that this idea would considerably improve the security of the protocol, and we will not pursue it further for coherence with the protocols that will follow.

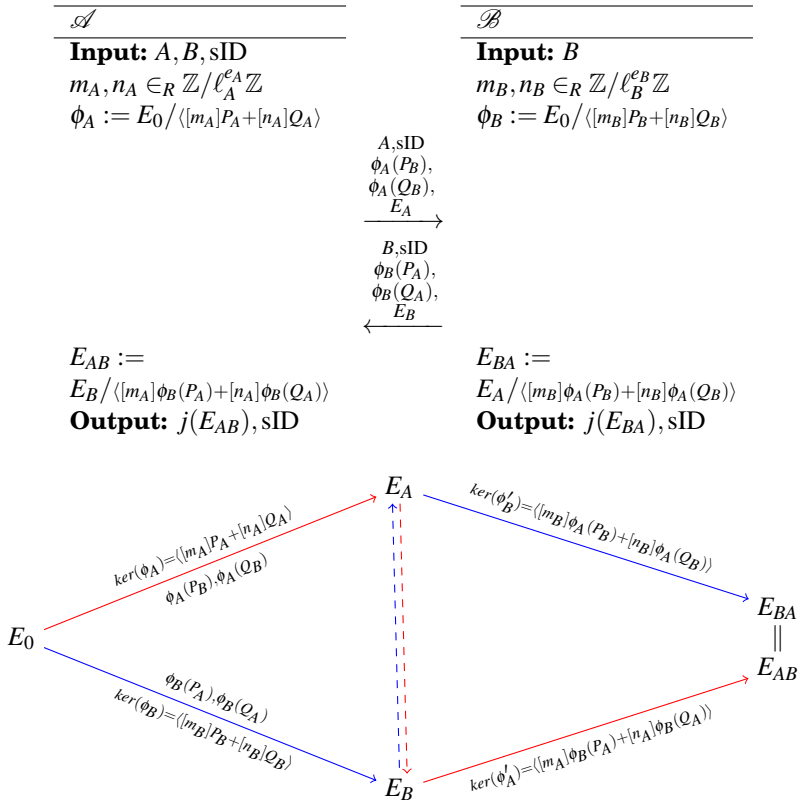


Figure D.1: Key-exchange protocol using isogenies on supersingular curves.

2. Peggy sends the curves $E_1 = E/\langle R \rangle$ and $E_2 = E/\langle S, R \rangle$ to Vic.
3. Vic selects a random bit b and sends it to Peggy.
4. If $b = 0$, Peggy reveals the points R and $\phi(R')$. Vic accepts if they have order $\ell_b^{e_B}$ and generate the kernels of isogenies $E \to E_1$ and $E/\langle S \rangle \to E_2$, respectively.
5. If $b = 1$, Peggy reveals the point $\psi(S)$. Vic accepts if it has order $\ell_A^{e_A}$ and generates the kernel of an isogeny $E_1 \to E_2$.

Key exchange

The key exchange protocol is a variation *à la* Diffie-Hellman over diagram (D.1). The idea is to let Alice choose ϕ, while Bob chooses ψ. Although similar in spirit to the protocol based on the action of the class group on ordinary elliptic curves of [Sto10], a main technical difference is that, since ideal classes no longer commute (or indeed even multiply together) in the supersingular case, extra information must be communicated as part of the protocol in order to ensure that both parties arrive at the same common value.

We fix as public parameters a supersingular curve E_0 defined over $\mathbb{F}_{p^2}$, and bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ which generate $E_0[\ell_A^{e_A}]$ and $E_0[\ell_B^{e_B}]$ respectively, so that

$\langle P_A, Q_A\rangle = E_0[\ell_A^{e_A}]$ and $\langle P_B, Q_B\rangle = E_0[\ell_B^{e_B}]$. Alice chooses two random elements $m_A, n_A \in_R \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$, not both divisible by ℓ_A, and computes an isogeny $\phi_A : E_0 \to E_A$ with kernel $K_A := \langle [m_A]P_A + [n_A]Q_A\rangle$. Alice also computes the image $\{\phi_A(P_B), \phi_A(Q_B)\} \subset E_A$ of the basis $\{P_B, Q_B\}$ for $E_0[\ell_B^{e_B}]$ under her secret isogeny ϕ_A, and sends these points to Bob together with E_A. Similarly, Bob selects random elements $m_B, n_B \in_R \mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$ and computes an isogeny $\phi_B : E_0 \to E_B$ having kernel $K_B := \langle [m_B]P_B + [n_B]Q_B\rangle$, along with the points $\{\phi_B(P_A), \phi_B(Q_A)\}$. Upon receipt of E_B and $\phi_B(P_A), \phi_B(Q_A) \in E_B$ from Bob, Alice computes an isogeny $\phi'_A : E_B \to E_{AB}$ having kernel equal to $\langle [m_A]\phi_B(P_A) + [n_A]\phi_B(Q_A)\rangle$; Bob proceeds *mutatis mutandis*. Alice and Bob can then use the common j-invariant of

$$E_{AB} = \phi'_B(\phi_A(E_0)) = \phi'_A(\phi_B(E_0)) = E_0/\langle [m_A]P_A + [n_A]Q_A, [m_B]P_B + [n_B]Q_B\rangle$$

to form a secret shared key.

The full protocol is given in Figure D.1. We denote by A and B the identifiers of Alice and Bob, and use sID to denote the unique session identifier.

Public-key encryption

The key-exchange protocol of Section D.3 can easily be adapted to yield a public-key cryptosystem, in much the same way that Elgamal encryption follows from Diffie-Hellman. We briefly give the details here. All notation is the same as above. Stolbunov [Sto10] uses a similar construction, upon which ours is based.

Setup Choose $p = \ell_A^{e_A}\ell_B^{e_B} \cdot f \pm 1$, E_0, $\{P_A, Q_A\}$, $\{P_B, Q_B\}$ as above. Let $\mathscr{H} = \{H_k : k \in K\}$ be a hash function family indexed by a finite set K, where each H_k is a function from $\mathbb{F}_{p^2}$ to the message space $\{0,1\}^w$.

Key generation Choose two random elements $m_A, n_A \in_R \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$, not both divisible by ℓ_A. Compute $E_A, \phi_A(P_B), \phi_A(Q_B)$ as above, and choose a random element $k \in_R K$. The public key is the tuple $(E_A, \phi_A(P_B), \phi_A(Q_B), k)$ and the private key is (m_A, n_A, k).

Encryption Given a public key $(E_A, \phi_A(P_B), \phi_A(Q_B), k)$ and a message $m \in \{0,1\}^w$, choose two random elements $m_B, n_B \in_R \mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$, not both divisible by ℓ_B, and compute

$$h = H_k(j(E_{AB})),$$
$$c = h \oplus m.$$

The ciphertext is $(E_B, \phi_B(P_A), \phi_B(Q_A), c)$.

Decryption Given a ciphertext $(E_B, \phi_B(P_A), \phi_B(Q_A), c)$ together with a private key (m_A, n_A, k), compute the j-invariant $j(E_{AB})$ and set

$$h = H_k(j(E_{AB})),$$
$$m = h \oplus c.$$

The plaintext is m.

D.4 Algorithmic aspects

We now give specific algorithms to implement the aforementioned steps efficiently.

Parameter generation

For any fixed choice of $\ell_A^{e_A}$ and $\ell_B^{e_B}$, one can easily test random values of f (of any desired cryptographic size) until a value is found for which $p = \ell_A^{e_A}\ell_B^{e_B} \cdot f - 1$ or $p = \ell_A^{e_A}\ell_B^{e_B} \cdot f + 1$ is prime. The prime number theorem in arithmetic progressions (specifically, the effective version of Lagarias and Odlyzko [LO77]) provides a sufficient lower bound for the density of such primes.

Once the prime $p = \ell_A^{e_A}\ell_B^{e_B} \cdot f \pm 1$ is known, Bröker [Brö09] has shown that it is easy to find a supersingular curve E over $\mathbb{F}_{p^2}$ having cardinality $(p \mp 1)^2 = (\ell_A^{e_A}\ell_B^{e_B} \cdot f)^2$. Starting from E, one can select a random supersingular curve E_0 over $\mathbb{F}_{p^2}$ by means of random walks on the isogeny graph (cf. Proposition D.1); alternatively, one can simply take $E_0 = E$. In either case, E_0 has group structure $(\mathbb{Z}/(p \mp 1)\mathbb{Z})^2$. To find a basis for $E_0[\ell_A^{e_A}]$, choose a random point $P \in_R E_0(\mathbb{F}_{p^2})$ and multiply it by $(\ell_B^{e_B} \cdot f)^2$ to obtain a point P' of order dividing $\ell_A^{e_A}$. With high probability, P' will have order exactly $\ell_A^{e_A}$; one can of course check this by multiplying P' by powers of ℓ_A. If the check succeeds, then set $P_A = P'$; otherwise try again with another P. A second point Q_A of order $\ell_A^{e_A}$ can be obtained in the same way. To check whether Q_A is independent of P_A, simply compute the Weil pairing $e(P_A, Q_A)$ in $E[\ell_A^{e_A}]$ and check that the result has order $\ell_A^{e_A}$; as before, this happens with high probability, and if not, just choose another point Q_A. Note that the choice of basis has no effect on the security of the scheme, since one can convert from one basis to another using extended discrete logarithms, which are easy to compute in $E_0[\ell_A^{e_A}]$ by [Tes99].

Key exchange and other protocols

The key exchange is performed in two rounds. In each round Alice and Bob do the following operations on each side:

1. Compute $\langle R \rangle = \langle [m]P + [n]Q \rangle$ for some points P, Q;
2. Compute the isogeny $\phi : E \to E/\langle R \rangle$ for a curve E;
3. In the first round (only), compute $\phi(R)$ and $\phi(S)$ for some points R, S;

where the curve E and the points P, Q, R, S depend on the round and the player, as shown in Figure D.1. Similar operations are needed by the other protocols of Section D.3. We demonstrate how to implement efficiently each of these steps.

Computing $\langle [m]P + [n]Q \rangle$

We first observe that it suffices to compute any generator of $\langle [m]P + [n]Q \rangle$. Without loss of generality we can assume that m is invertible modulo the order of the group, in which case $R' = P + [m^{-1}n]Q$ is one such generator. Computing R' by a standard double-and-add approach requires half the effort of computing $[m]P + [n]Q$ naively (see [ElG85; Sol01; Ant+06], though, for better ways of computing the latter).

However, computing $P + [m^{-1}n]Q$ by double-and-add has one major drawback: it is trivially vulnerable to simple power analysis (SPA) [KJJ99]. To avoid SPA, one can use a Montgomery ladder [Mon87] to compute $[m^{-1}n]Q$, and then add P, but this is significantly slower.

Instead, we propose in Algorithm 11 a much more efficient ladder, computing $P + [m^{-1}n]Q$ directly. The idea behind it is simple: at each iteration the registers A,B and C contain respectively the values $[x]Q$, $[x+1]Q$ and $P + [x]Q$, for x equal to the

Algorithm 11 Three-point ladder to compute $P+[t]Q$.

Input: t, P, Q;
1: Set $A=0, B=Q, C=P$;
2: Compute $Q-P$;
3: **for** i decreasing **from** $|t|$ **to** 1 **do**
4: Let t_i be the i-th bit of t;
5: **if** $t_i = 0$ **then**
6: $A = 2A, \quad B = \mathrm{dadd}(A,B,Q), \quad C = \mathrm{dadd}(A,C,P)$;
7: **else**
8: $A = \mathrm{dadd}(A,B,Q), \quad B = 2B, \quad C = \mathrm{dadd}(B,C,Q-P)$;
9: **end if**
10: **end for**
Output: $C = P+[t]Q$.

leftmost bits of $m^{-1}n$. The function $\mathrm{dadd}(A,B,C)$ is a *differential addition*: it computes the sum $A+B$ knowing $C=A-B$. Montgomery curves have a very efficient differential addition [Mon87], making the implementation of our ladder as efficient as the naive double and add on twisted Edwards curves (see Section D.4).

Computing smooth degree isogenies

It remains to describe how Alice and Bob can compute and evaluate the isogenies. Let E be an elliptic curve, and let R be a point of order ℓ^e. Our goal is to compute the image curve $E/\langle R\rangle$ and to evaluate the isogeny $\phi : E \to E/\langle R\rangle$ at some points of E.

Since the degree of ϕ is smooth, it is best to decompose it as a chain of ℓ-isogenies. Set $E_0 = E$, $R_0 = R$ and, for $0 \le i < e$, let

$$E_{i+1} = E_i/\langle \ell^{e-i-1} R_i\rangle, \qquad \phi_i : E_i \to E_{i+1}, \qquad R_{i+1} = \phi_i(R_i).$$

Then $E/\langle R\rangle = E_e$ and $\phi = \phi_{e-1} \circ \cdots \circ \phi_0$.

The curve E_{i+1} and the isogeny ϕ_i can be computed using Vélu's formulas [Vél71], once the ℓ-torsion subgroup $\langle R_i\rangle$ of E_i is known. This immediately suggests two strategies having quadratic complexity in e, as described in [JD11].

However, we can do much better. Figure D.2 summarizes the computational structure of the problem for $e=6$. Bullets represent points, with points on the same horizontal level having the same order, and points on the same left diagonal belonging to the same curve. Dashed edges are directed and go from top to bottom; leftward edges represent multiplication by ℓ, and rightward edges represent evaluation of an ℓ-isogeny.

At the beginning of the algorithm, only the point R_0 is known. Our goal is to compute all the points on the bottom line. Indeed, from the knowledge of the point $[\ell^{e-i-1}]R_i$ we can compute the kernel of ϕ_i using $O(\ell)$ point additions. We then apply Vélu's formulas to compute ϕ_i and E_{i+1}. From this point on, we can forget about the number theoretic nature of the problem and purely focus on its combinatorial structure. Lemma D.3 guarantees that any solution to the combinatorial problem yields a valid strategy to compute ϕ. Before stating it, we first formally rephrase the picture in Figure D.2.

Definition D.2. Let T_n be the portion of the unit triangular equilateral lattice contained between the x-axis, the line $y=\sqrt{3}x$ and the line $y=-\sqrt{3}(x-n+1)$. We call T_n the *discrete equilateral triangle* (DET) of side n.

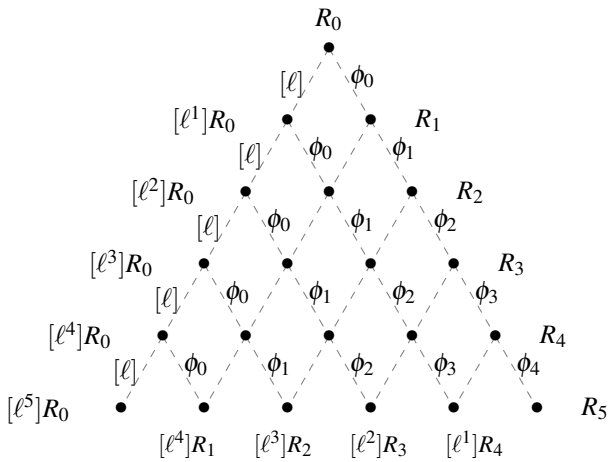


Figure D.2: Computational structure of the construction of $\phi = \phi_5 \circ \cdots \circ \phi_0$.

An *edge* is any segment of unit length directed downwards to the x-axis connecting two points of T_n. We say an edge is a *left edge* if it has positive slope, a *right edge* otherwise. This definition yields a directed acyclic graph (DAG) structure on T_n.

We equip the vertices of any directed graph G with the ordering $\rightarrow_G$ defined by: $x \rightarrow_G y$ if and only if there exists a path in G from x to y. We use $\rightarrow$ as a shorthand for $\rightarrow_{T_n}$. The *leaves* of G are the final vertices, and the *roots* of G are the initial vertices. The graph T_n has the n vertices on the x-axis as leaves and the top-most vertex as its unique root. We write $|G|$ for the number of leaves of G.

For any two vertices y, y' of T_n, there exists a most final vertex x with the property that $x \rightarrow y$ and $x \rightarrow y'$. We write $x = y \wedge y'$.

A *strategy* S is a sub-graph of T_n having a unique root. It is *full* if its leaves contain those of T_n; in this case, the root of S is the same as that of T_n. The *fork* of S is the final-most vertex x of S that is comparable (for the ordering $\rightarrow_S$) to all vertices of S.

Lemma D.3. *Any full strategy yields a valid algorithm to compute the isogeny* $\phi = \phi_{n-1} \circ \cdots \circ \phi_0$ *by decorating the DET as shown in Figure D.2.*

Proof. Travel the graph in depth-first left-first order. Upon reaching the bottom, apply Vélu's formulas before going right. □

At this point it should be clear that a strategy that passes twice through an interior vertex does unnecessary computations. Another waste of resources is a strategy having a leaf distinct from the leaves of T_n.

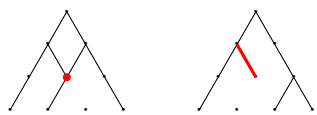


Figure D.3: Two ill-formed strategies

We define a *well-formed* strategy to be one that has no such useless edge[3]. Figure D.4 shows all the seven well-formed full strategies for $n = 4$, the leftmost and the rightmost ones corresponding respectively to the *isogeny-based* and the *multiplication-based* algorithms from [JD11].

Figure D.4: The seven well-formed full strategies for $n = 4$.

It is clear that some strategies are computationally better than others. From the figures, we see that the multiplication-based and isogeny-based strategies have a number of edges quadratic in n, whereas the middle strategy in Figure D.4 extends to a family of *balanced strategies* with asymptotically $\frac{1}{2\log 2} n \log n$ left and right edges.

We are thus interested in computing the "best" full strategy, according to some measure of computational effort. We first observe that any well-formed strategy has a binary tree topology, obtained by forgetting the internal nodes of out-degree less than two and by keeping the same connectivity structure. Conversely, to any binary tree A with n leaves one can canonically associate a strategy S on T_n as follows: if $n = 1$, then $S = T_1$; else, let S' be the strategy associated to the left branch of A, S'' be the translated to the right by $|S'|$ of the strategy associated to the right branches of A, and r', r'' be their roots. Then $r' \wedge r''$ is the root of T_n and we define $S = rr' \cup rr'' \cup S' \cup S''$, where rr' and rr'' are the (unique) paths from r to r' and r'' in T_n.

For example, in Figure D.4 the three middle strategies share the same tree topology, and the middle one is the canonical one.

In our original problem, left edges are multiplications by ℓ, while right edges are ℓ-isogeny evaluations, and these steps have different costs. Thus it makes sense to assign different weight to left and right edges.

We fix a *measure* on T_n, i.e., a pair (p,q) of positive real numbers, where p represents the weight of a left edge (i.e., a point multiplication) and q represents the weight of a right edge (i.e., an isogeny). For any set of edges S of T_n, we write (S) for the sum of the weights of all edges of S.

If $x \to y$, then all paths going from x to y in T_n have the same measure; we write (xy) for this measure. For all vertices x, y, y', y'' such that $x \to y$, $y \to y''$ and $y \to y''$, we have the inequality $(xy) + (yy') + (yy'') \leq (xy') + (xy'')$.

We find optimal strategies in two steps. The first step (Lemma D.4) shows that, for any measure, only canonical strategies are interesting. Then we determine the optimal full strategy for a given measure (Proposition D.7).

Lemma D.4. *Among all the strategies sharing the same tree topology, the canonical strategy is minimal with respect to any measure.*

Lemma D.4 follows from Lemma D.5.

Lemma D.5. *Let S be a strategy with root r. Let $\widehat{S}$ be the canonical strategy associated to S and $\widehat{r}$ be the root of $\widehat{S}$. Then $r \to \widehat{r}$, and $(r\widehat{r}) + (\widehat{S}) \leq (S)$.*

[3] We do not know much about well-formed full strategies. One remarkable fact is that they are in one-to-one correspondence with certain instances of Gelfand-Tsetlin patterns [OEI12].

Proof. We prove this by induction on S. Let t be the fork of S. If t has no children in S, then it is the unique leaf of S and the lemma is obviously true. Therefore, we may assume that t has two children in S.

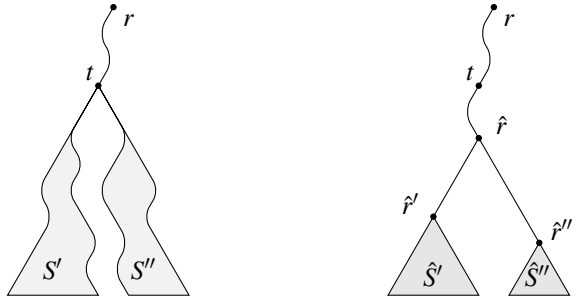


Figure D.5: Proof of Lemma D.5.

Let S', S'' be the right and left sub-strategies of S with root t. We apply the induction hypothesis on S' and S'' and use $\widehat{r} = \widehat{r}' \wedge \widehat{r}''$ to derive the following (in)equalities:

$$\begin{aligned}
(S') + (S'') + (rt) &= (S) && \text{(definition of } (S)\text{)},\\
(t\widehat{r}') + (\widehat{S}') &\leq (S') && \text{(induction hypothesis on } \widehat{S}'\text{)},\\
(t\widehat{r}'') + (\widehat{S}'') &\leq (S'') && \text{(induction hypothesis on } \widehat{S}''\text{)},\\
(t\widehat{r}) + (\widehat{r}\widehat{r}') + (\widehat{r}\widehat{r}'') &\leq (t\widehat{r}') + (t\widehat{r}'') && \text{(definition of } \widehat{r}\text{)},\\
(r\widehat{r}) &= (rt) + (t\widehat{r}) && \text{(path } r \to t \to \widehat{r}\text{)},\\
(\widehat{S}) &= (\widehat{r}\widehat{r}') + (\widehat{r}\widehat{r}'') + \\
&\quad (\widehat{S}') + (\widehat{S}'') && \text{(definition of } \widehat{S}\text{)}.
\end{aligned}$$

By summing all of these and cancelling all extra terms, we obtain the desired result. □

Now that we have ruled out non-canonical strategies, we proceed to determine the best ones. From this point on, we are only interested in full strategies. Thus, we shall call *optimal* a strategy that is minimal, among all full strategies with n leaves, with respect to a measure (p,q).

In practice, we can compute optimal strategies using the fact that, for a fixed measure, optimality is a local property. For a full canonical strategy $S \neq T_1$, having i leaves to the left of its root, we define its *left branch* as $S \cap T_i$. We also define its *right branch* as $S \cap T'$, where T' is $T_{|S|-i}$ translated by i to the right. Because S is full and canonical, both branches are full canonical strategies too.

Lemma D.6. *Let S be an optimal strategy. Let S' and S'' be, respectively, its left and right branch. Then S' and S'' are optimal strategies.*

Proof. By Lemma D.4, we know that S is a canonical strategy. Hence, S' and S'' are well defined. Now, suppose that S' were not optimal. By substituting an optimal strategy for S' inside S, we obtain a strategy with weight lower than (S). The same argument holds for S''. □

Define $C_{p,q}(n)$ to be the cost of the optimal strategies with n leaves. Lemma D.6 says that

$$C_{p,q}(n) = \min_{i=1,\dots,n-1} \big(C_{p,q}(i) + C_{p,q}(n-i) + (n-i)p + iq\big). \tag{D.4}$$

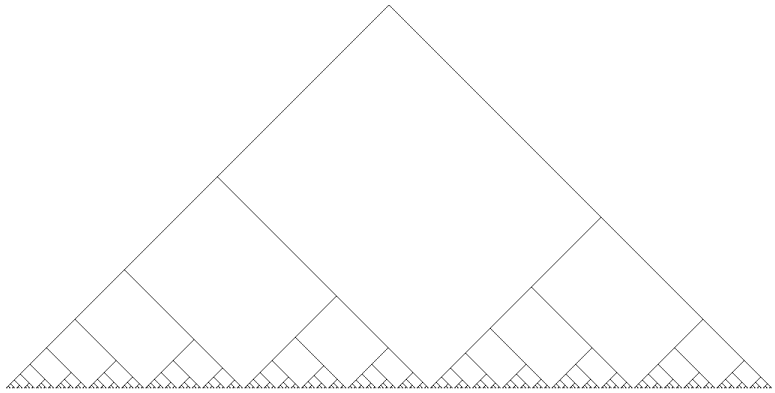


Figure D.6: Optimal strategy for $n = 512, p = 4.6, q = 2.8$.

This equality suggests a dynamic-programming algorithm that, given p and q, computes $C(n)$ in time $O(n^2)$. A straightforward Python implementation computes the optimal strategies for $n = 1024$ in less than one second, indicating that the dynamic-programming approach is very satisfying in practice. Figure D.6 shows an optimal strategy for $n = 512, p = 4.6, q = 2.8$.

However, it is also possible to mathematically characterize the optimal strategies. Since the optimal strategies for (p,q) are symmetrical to those for (q,p), we may assume that $p < q$. For simplicity, we will also assume that p and q are integers: the generalization to the case of rational p and q is straightforward.

Let (u_m) be the sequence defined by $u_{p+1} = \cdots = u_{p+q} = 1$ and $u_m = u_{m-p} + u_{m-q}$, and $f(n)$ be the function defined by $f(1) = 0$, $f(n) = f(u_m) + m(n - u_m)$ for all $n \in [u_m, u_{m+1}]$. We check that f satisfies the recurrence relation

$$f(u_m) = f(u_{m-p}) + f(u_{m-q}) + pu_{m-p} + qu_{m-q}. \tag{D.5}$$

Proposition D.7. *A full strategy S is optimal if, and only if, it is canonical, both its left and right branches S' and S'' are optimal, and*

$$u_{m-q} \leq \left|S'\right| \leq u_{m-q+1}, \quad u_{m-p} \leq \left|S''\right| \leq u_{m-p+1},$$

where m is such that $u_m \leq |S| < u_{m+1}$. In this case, the weight of S is $f(|S|)$.

For example, in the case where $p = 1$ and $q = 2$, the sequence (u_m) is the sequence of Fibonacci numbers, starting at $u_2 = 1, u_3 = 1, u_4 = 2$..., and a strategy S with $u_m \leq |S| < u_{m+1}$ is optimal if, and only if, the left and right branches S' and S'' satisfy $u_{m-2} \leq |S'| \leq u_{m-1}$ and $u_{m-1} \leq |S''| \leq u_m$.

Proof. We prove this by induction on $n = |S|$. Let $n = n' + n''$ with $n', n'' > 0$ and let $S = S_{n',n''}$ be any canonical full strategy with optimal left and right branches S', S'', such that $|S'| = n'$ and $|S''| = n''$. Then any optimal strategy is one of the $S_{n',n''}$. Therefore, we find them by looking at the sign of $\delta_n = (S_{n'+1,n''-1}) - (S_{n',n''})$.

Let m', m'' be such that $u_{m'} \leq n' < u_{m'+1}$ and $u_{m''} \leq n'' < u_{m''+1}$. Using the induction hypothesis on S' and S'', we have $(S_{n',n''}) = f(n') + f(n'') + n'q + n''p$, and therefore

$$\delta_n = \begin{cases} (m'+q) - (m''+p), & \text{if } n'' \geq u_{m''} + 1, \\ (m'+q) - (m''+p) + 1, & \text{if } n'' = u_{m''}. \end{cases} \tag{D.6}$$

We find that $\delta_n \leq 0$ exactly when $n' < u_{m-q+1}$ and $n'' > u_{m-p}$, and $\delta_n \geq 0$ exactly when $n' \geq u_{m-q}$ and $n'' \leq u_{m-p+1}$. The optimality condition follows from this. It remains only to check that $(S) = f(|S|)$: this results from relation (D.5). □

In particular, there are in general several optimal strategies of a given size. Moreover, with z being the (unique) root in $[0,1]$ of the equation $z^p + z^q - 1 = 0$, this gives the asymptotic equivalents

$$\left|S'\right| \sim z^q |S|, \tag{D.7}$$

$$C_{p,q}(n) = (S) \sim -\frac{1}{\log z} n \log n. \tag{D.8}$$

The relative cost of an optimal strategy is therefore asymptotically $-\frac{2\log 2}{\log(z^{p+q})}$ times the cost of the balanced strategy. In the case where $(p,q) = (4.6, 2.8)$, as in Table D.1 with $\ell = 2$, this gives an improvement of 2.1% over the balanced strategy.

Choice of the models

At each phase of the key generation it is important to use models for elliptic curves that offer the fastest formulas for doubling, addition, isogeny computation and evaluation, etc.

To measure efficiency, we count the number of elementary operations in $\mathbb{F}_{p^2}$: we write I, M, S for the costs of one inversion, multiplication and squaring respectively, and we make the assumption $S \leq M \leq I$. We neglect additions, subtractions and comparisons. We refer to the Explicit Formulas Database (EFD) [BL07] for operation counts of elliptic point addition, doubling, etc. in various models and coordinate systems. Contrary to the convention taken in the EFD, we count multiplications by constants (other than small integers) as ordinary multiplications.

Any curve used in our cryptosystem has group structure $(\mathbb{Z}/(p \mp 1)\mathbb{Z})^2$. Hence either the curve or its twist has a point of order 4. Consequently, it is isomorphic to a twisted Edwards curve and to a Montgomery curve [Ber+08].

Twisted Edwards curves [Ber+08] have equation

$$E_{a,d} \ : \ ax^2 + y^2 = 1 + dx^2y^2. \tag{D.9}$$

They have many interesting properties, but what interests us most is their very efficient addition and doubling formulas. Using projective coordinates, one point addition costs $12M + 1S$ and one point doubling costs $4M + 4S$. When one of the points is scaled to have Z-coordinate equal to 1, these costs drop to $11M + 1S$ and $3M + 4S$ respectively.

Montgomery curves [Mon87] have equation

$$M_{B,A} \ : \ By^2 = x^3 + Ax^2 + x. \tag{D.10}$$

They have very efficient arithmetic on their Kummer line, i.e., by representing points by the coordinates $(X : Z)$ where $x = X/Z$. Using this representation, a point can be doubled using $3M + 2S$, or $2M + 2S$ when it is scaled to have Z-coordinate equal to 1.

The Kummer line identifies P with $-P$; thus it is not possible to add two distinct points. However it is still possible to perform what is called *differential addition*, i.e., adding points P and Q for which the difference $P - Q$ is known. One differential addition can be computed using $4M + 2S$, or $3M + 2S$ when the difference $P - Q$ is scaled to have Z-coordinate equal to 1.

By using doublings and differential additions, it is possible to compute any scalar multiple of a point using a Montgomery ladder [Mon87]. Also observe that since P and $-P$ generate the same subgroup, isogenies can be defined and evaluated correctly on the Kummer line. We shall give formulas for this operation later.

It is shown in [Ber+08] that the twisted Edwards curve $E_{a,d}$ is birationally equivalent to the Montgomery curve $M_{A,B}$ where

$$A = \frac{2(a+d)}{a-d}, \qquad B = \frac{4}{a-d}, \tag{D.11}$$

and where the transformation is given by the following maps (in affine coordinates)

$$\begin{aligned} \psi &: E_{a,d} \to M_{A,B}, & \psi^{-1} &: M_{A,B} \to E_{a,d}, \\ (x,y) &\mapsto \left(\frac{1+y}{1-y}, \frac{1+y}{(1-y)x}\right), & (x,y) &\mapsto \left(\frac{x}{y}, \frac{x-1}{x+1}\right). \end{aligned} \tag{D.12}$$

Hence, after the models $E_{a,d}$ and $M_{A,B}$ are computed, points in projective coordinates can be moved from one model to the other at a cost of a few multiplications (and no inversions).

Now we detail the cost of each step of the key generation algorithm.

Computing $\langle [m]P + [n]Q \rangle$

We have already mentioned two algorithms to compute the value of $P + [m^{-1}n]Q$. One solution is to express the points P and Q in projective Edwards coordinates and perform a double and add followed by an addition. If we scale Q to have Z-coordinate equal to 1, computing $[m^{-1}n]Q$ costs $9.5M + 4.5S$ per bit on average.

Because addition by P is performed last, this approach cannot be used with points on the Montgomery curve in Kummer coordinates, but we can use the ladder given in Algorithm 11 instead. We first compute $P - Q$ using point addition in full projective coordinates (either by using the standard chord-and-tangent law, or by using the equivalence with twisted Edwards curves). Then we scale P, Q and $P - Q$ to have Z-coordinate equal to 1. We can now discard the Y coordinate and work on the Kummer line. At each iteration we perform one doubling and two differential additions (with one of the scaled points P, Q and $P - Q$). The total cost of one iteration is thus $9M + 6S$.

In general the cost of one squaring is close to the cost of one multiplication. Thus the ladder algorithm is slightly slower than double-and-add. However the advantages of the ladder are SPA resistance and an implementation simplified by not using Edwards coordinates at all for key generation.

Isogenies of Montgomery curves

The literature on efficient formulas for evaluating small degree isogenies is much less extensive than for point multiplication. We now give explicit formulas for isogenies of Montgomery curves, and optimize the degree 2 and 3 cases. Our goal is to obtain the most efficient formulas for isogeny evaluation, and thus we seek to avoid inversions and square root computations as much as possible.

Let E be the Montgomery curve defined by Eq. (D.10). It has a point of order two $P_2 = (0,0)$, and a point of order four $P_4 = (1, \sqrt{(A+2)/B})$—sometimes defined over a quadratic extension—such that $[2]P_4 = P_2$. Montgomery curves have twists of the

form $\tilde{y} = \sqrt{c}y$; these are isomorphisms when c is a square. The change of coordinates $\tilde{x} = x/B, \tilde{y} = y/B$ brings the curve E to the Weierstrass form

$$\tilde{y}^2 = \tilde{x}^3 + \frac{A}{B}\tilde{x}^2 + \frac{1}{B^2}\tilde{x}, \tag{D.13}$$

and the point P_4 to $P_4' = (1/B, \ldots)$. Conversely, given a Weierstrass curve with equation $\tilde{y}^2 = \tilde{x}^3 + a\tilde{x}^2 + b\tilde{x}$, together with a point $P_4 = (1/\beta, \ldots)$—with its ordinate possibly lying in a quadratic extension—such that $[2]P_4 = (0,0)$, the change of variables $\tilde{x} = x/\beta, \tilde{y} = y/\beta$ brings the curve to the Montgomery form $\beta y^2 = x^3 + a\beta x^2 + x$.

Given a point $P \neq \pm P_4$ of order 4, we will need to compute the isomorphism of Montgomery curves that brings $[2]P$ in $(0,0)$ and P in $(1, \ldots)$. Let X be the abscissa of P and X_0 the abscissa of $[2]P$; by a straightforward calculation, we find that this isomorphism is given by the map

$$\begin{aligned} &\iota : E \to E', \\ &(x,y) \mapsto \left(\frac{x - X_0}{X - X_0}, \frac{y}{X - X_0} \right), \end{aligned} \tag{D.14}$$

and the new curve has equation

$$E' \ : \ \frac{B}{X - X_0} y^2 = x^3 + \frac{3X_0 + A}{X - X_0} x^2 + x.$$

By precomputing $1/(X - X_0)$, and sharing common subexpressions, the above map can be evaluated on a point in full projective coordinates using $3M$ operations, or $2M$ using Kummer coordinates.

Let G be a subgroup of the Montgomery curve E of odd cardinality ℓ and let h be the degree $(\ell - 1)/2$ polynomial vanishing on the abscissas of G. With a twist $y = \tilde{y}/\sqrt{B}$, we can put E in the form

$$E \ : \ \tilde{y}^2 = \tilde{x}^3 + A\tilde{x}^2 + \tilde{x},$$

and this does not change the abscissas of G nor the polynomial h. Now, composing the twist with Vélu's formulas gives an isogeny

$$\begin{aligned} &\phi : E \to E/G, \\ &(x,y) \mapsto \left(\frac{g(x)}{h(x)^2}, y\sqrt{B} \left(\frac{g(x)}{h(x)^2} \right)' \right). \end{aligned}$$

We now need to express E/G in Montgomery form. Because ℓ is odd, the point $(0,0)$ of E is sent to a point of order two in E/G, and the change of variables $\tilde{x} = x - g(0)/h(0)^2$ brings this point to $(0,0)$.

Now, $\phi(P_4)$ is a point of order four lying above $(0,0)$ (possibly in a quadratic extension). Its abscissa is rational and is given by

$$\frac{1}{\beta} = \frac{g(1)}{h(1)^2} - \frac{g(0)}{h(0)^2}, \tag{D.15}$$

so we further apply the change of variables $\tilde{x} = \bar{x}/\beta, \tilde{y} = \bar{x}/\beta$ to obtain a Montgomery curve. Finally, we have to twist back the model in order to obtain a curve isogenous over the base field: the twist $\bar{y} = y\sqrt{B}$ cancels with the first one and leaves us with square-root-free formulas.

Given h as input, the cost of evaluating the whole path is $O(\ell)$ operations in the base field, using the formula in [Bos+08, Proposition 4.1] to evaluate g/h.

Let now P be a 3-torsion point, let $G = \{0, P, -P\}$ and let X be the abscissa of P (and $-P$). If we specialize the formula to the case $\ell = 3$, we have $h = x - X$ and

$$\begin{aligned} &\phi : E \to E/G, \\ &(x,y) \mapsto \left(\frac{x\left(x-\frac{1}{X}\right)^2}{(x-X)^2} X^2,\ y \frac{\left(x-\frac{1}{X}\right)\left(\left(x-\frac{1}{X}\right)(x-X)+2x\left(\frac{1}{X}-X\right)\right)}{(x-X)^3} X^2 \right), \end{aligned} \tag{D.16}$$

and the curve E/G has equation

$$E/G \ :\ BX^2y^2 = x^3 + \left(A + \frac{6}{X} - 6X\right) X^2x^2 + x.$$

By precomputing X and X^2, and sharing common subexpressions, the above isogeny can be evaluated on a point in full projective coordinates using $11M + 2S$ operations, or $4M + 2S$ using Kummer coordinates.

When ℓ is even, things get more complicated. Recall that $P_2 = (0,0)$ and $P_4 = (1, \sqrt{(A+2)/B})$. The isogeny of degree 2 vanishing on P_2 and mapping P_4 to $(0,0)$ is readily seen as being

$$F \ :\ By^2 = x^3 + (A+6)x^2 + 4(2+A)x, \tag{D.17}$$

$$\begin{aligned} &\phi : E \to F, \\ &(x,y) \mapsto \left(\frac{(x-1)^2}{x}, y\left(1 - \frac{1}{x^2}\right) \right). \end{aligned} \tag{D.18}$$

It is not immediately evident how to put F in Montgomery form without computing square roots. Let P_8 be a point satisfying $[2]P_8 = P_4$. Then we have $\phi(P_8) = (2\sqrt{2+A}, \ldots)$, and F can be put in the form

$$\frac{B}{2\sqrt{2+A}} y^2 = x^3 + \frac{A+6}{2\sqrt{2+A}} x^2 + x,$$

with the point $(1, \ldots)$ being the image of P_8. Any other isogeny of degree 2 can be treated by applying Eq. (D.14) to move the generator of the kernel in $(0,0)$.

By precomputing $1/\phi(P_8)$, and sharing common subexpressions, the above isogeny can be evaluated on a point in full projective coordinates using $5M + 3S$ operations, or $2M + S$ using Kummer coordinates. Unfortunately, this formula takes no square roots only if an 8-torsion point above P_2 is known.

Alternatively, ϕ and F being as before, we consider the isogeny $\psi : F \to F/\langle (0,0) \rangle$ given by

$$G \ :\ \frac{B}{2-A} y^2 = x^3 - 2\frac{A+6}{2-A} x^2 + x, \tag{D.19}$$

$$\begin{aligned} &\psi : F \to G, \\ &(x,y) \mapsto \left(\frac{1}{2-A} \frac{(x+4)(x+(A+2))}{x}, \frac{y}{2-A}\left(1 - \frac{4(2+A)}{x^2}\right) \right). \end{aligned} \tag{D.20}$$

Then $\phi_4 = \psi \circ \phi$ is an isogeny of degree four $\phi_4 \colon E \to E/\langle P_4 \rangle$, and the point $\left(1, \sqrt{-4(A+2)/B}\right)$ of G generates the kernel of the dual isogeny $\hat{\phi}_4$. Any other isogeny of degree 4 can be treated by applying Eq. (D.14) to move the kernel point to $(1, \ldots)$.

By precomputing $1/(2-A)$, and sharing common subexpressions, the isogeny ϕ_4 can be evaluated on a point in full projective coordinates using $10M+4S$ operations, or $4M+S$ using Kummer coordinates.

Unfortunately, this formula cannot be applied twice to obtain a degree 16 cyclic isogeny: indeed, a double application yields the multiplication-by-4 isogeny. If a chain of degree 4 isogenies is wanted, as for the algorithm in Subsection D.4, this formula must be combined with the isomorphism in Eq. (D.14).

Isogenies of composite smooth degree are computed by composing small degree isogenies as discussed in Subsection D.4. We focus on the fine tuning of this algorithm when the small isogenies have degrees $2, 3$ and 4.

It is important to remark that, after a generator R of the kernel of the isogeny has been computed, the algorithm does not use its ordinate at all: indeed, all the previous formulas for small degree isogenies only use the abscissas of the kernel points. Hence, we can throw away the ordinate of R altogether and only use scalar multiplication and isogeny evaluation formulas for points in Kummer coordinates.

As usual, some more care must be taken when computing cyclic isogenies of degree 2^e. In principle, one could compose either degree 2 (Eq. D.17) or degree 4 (Eq. D.19) isogenies; however we have already pointed out some caveats:

- Both equations require the kernel point to be moved to some specific coordinates. This can be achieved using the isomorphism in Eq. (D.14).
- After the first change of variables, Eq. (D.17) can be repeatedly chained with itself; however, from a point of order 2^e only $e-2$ degree 2 isogenies can be computed this way, because the formula requires the knowledge of a point of order 8.
- Eq. (D.19) cannot be directly chained with itself to compute a cyclic isogeny of degree 4^e. It must be composed, instead, with Eq. (D.14) at each step.

Having this in mind, two obvious strategies to compute degree 2^e isogenies are:

1. Use degree 2 isogenies as much as possible; use one degree 4 isogenies for the last two steps.
2. Use one degree 2 isogeny if e is odd, then use only degree 4 isogenies composed with isomorphisms.

Table D.2 below suggests that the second strategy yields a very small improvement over the first. The experiments of Section D.7 show that operations not accounted for in this analysis might give a greater advantage to the second strategy. However, for lack of time, we only implemented the first one.

Finally, it should be noted that both approaches, if implemented as described above, leak two bits of security. Indeed, the point $(1,\dots)$ of the image curve is a point of order 4 in the kernel of the dual of the computed isogeny (the secret). It is easy to mask this leakage by taking a random change of coordinates. However, for practical purposes, we found this masking more costly than simply adding two bits of security and letting the information leak, although technically the leaked information alters the complexity assumptions needed for the security proofs. For ease of analysis, we assume in Section D.5 that coordinate masking has been applied.

Table D.1: Comparative costs for multiplication and isogeny evaluation in projective Kummer coordinates, in number of multiplications and squarings, and assuming $S = 0.8M$.

ℓ	2	3	4
Isogeny	$2M+S$	$4M+2S$	$6M+S$
	2.8	5.6	6.8
Multiplication	$3M+2S$	$7M+4S$	$6M+4S$
	4.6	10.2	9.2

Table D.2: Comparative costs of the balanced and the optimal strategies for computing a degree 2^{514} ($\ell = 2,4$) or 3^{323} ($\ell = 3$) isogeny, assuming $S = 0.8M$.

	optimal strategy			balanced strategy		
ℓ	2	3	4	2	3	4
Isogenies	2741	1610	1166	2323	1430	1033
Multiplications	1995	1151	921	2307	1288	1025
Total cost	16852	20756	16402	17117	21146	16454

Table D.1 summarizes the costs of the isogeny evaluation and scalar multiplication formulas in light of these remarks. Because squaring in $\mathbb{F}_{p^2}$ is faster than multiplication, we also report the cost obtained by taking $S = 0.8M$ (a factor that roughly approximates the fact that squaring requires 2 multiplications in $\mathbb{F}_p$ instead of 3).

Table D.2 compares the total cost of isogeny evaluations and scalar multiplications in a balanced and an optimal strategy for $\ell = 2,3,4$, based on the costs given in Table D.1, at the classical 256-bit security level (see Section D.5 for our complexity assumptions). We make the following observations, backed up by the benchmarks in Section D.7:

- There may be a small advantage (less than 2%) in using isogenies of degree 4 instead of 2;
- The gain in using an optimal strategy instead of a balanced one is consistent with the predictions of Subsection D.4;
- The difference between 2^e and 3^e isogenies is more significant (about 20%), suggesting that degree 2^e isogenies may be preferable for constrained devices.

D.5 Complexity assumptions

As before, let p be a prime of the form $\ell_A^{e_A}\ell_B^{e_B} \cdot f \pm 1$, and fix a supersingular curve E_0 over $\mathbb{F}_{p^2}$ together with bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ of $E_0[\ell_A^{e_A}]$ and $E_0[\ell_B^{e_B}]$ respectively. In analogy with the case of isogenies over ordinary elliptic curves, we define the following computational problems, adapted for the supersingular case:

Problem D.8 (Decisional Supersingular Isogeny (DSSI) problem)**.** Let E_A be another supersingular curve defined over $\mathbb{F}_{p^2}$. Decide whether E_A is $\ell_A^{e_A}$-isogenous to E_0.

Problem D.9 (Computational Supersingular Isogeny (CSSI) problem)**.** Let the map $\phi_A : E_0 \to E_A$ be an isogeny whose kernel is $\langle [m_A]P_A + [n_A]Q_A \rangle$, where m_A and n_A are chosen at random from $\mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ and not both divisible by ℓ_A. Given E_A and the values $\phi_A(P_B)$, $\phi_A(Q_B)$, find a generator R_A of $\langle [m_A]P_A + [n_A]Q_A \rangle$.

We remark that given a generator $R_A = [m_A]P_A + [n_A]Q_A$, it is easy to solve for (m_A, n_A), since E_0 has smooth order and thus extended discrete logarithms are easy in E_0 [Tes99].

Problem D.10 (Supersingular Computational Diffie-Hellman (SSCDH) problem)**.** Let $\phi_A : E_0 \to E_A$ be an isogeny whose kernel is equal to $\langle [m_A]P_A + [n_A]Q_A \rangle$, and let $\phi_B : E_0 \to E_B$ be an isogeny whose kernel is $\langle [m_B]P_B + [n_B]Q_B \rangle$, where m_A, n_A (respectively m_B, n_B) are chosen at random from $\mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ (respectively $\mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$) and not both divisible by ℓ_A (respectively ℓ_B). Given the curves E_A, E_B and the points $\phi_A(P_B)$, $\phi_A(Q_B)$, $\phi_B(P_A)$, $\phi_B(Q_A)$, find the j-invariant of $E_0/\langle [m_A]P_A + [n_A]Q_A, [m_B]P_B + [n_B]Q_B \rangle$.

Problem D.11 (Supersingular Decision Diffie-Hellman (SSDDH) problem)**.** Given a tuple sampled with probability $1/2$ from one of the following two distributions:

- $(E_A, \phi_A(P_B), \phi_A(Q_B), E_B, \phi_B(P_A), \phi_B(Q_A), E_{AB})$, wherein the quantities E_A, $\phi_A(P_B)$, $\phi_A(Q_B)$, E_B, $\phi_B(P_A)$, and $\phi_B(Q_A)$ are as in the SSCDH problem and

$$E_{AB} \cong E_0/\langle [m_A]P_A + [n_A]Q_A, [m_B]P_B + [n_B]Q_B \rangle,$$

- $(E_A, \phi_A(P_B), \phi_A(Q_B), E_B, \phi_B(P_A), \phi_B(Q_A), E_C)$, wherein the quantities E_A, $\phi_A(P_B)$, $\phi_A(Q_B)$, E_B, $\phi_B(P_A)$, and $\phi_B(Q_A)$ are as in the SSCDH problem and

$$E_C \cong E_0/\langle [m'_A]P_A + [n'_A]Q_A, [m'_B]P_B + [n'_B]Q_B \rangle,$$

  where m'_A, n'_A (respectively m'_B, n'_B) are chosen at random from $\mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ (respectively $\mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$) and not both divisible by ℓ_A (respectively ℓ_B),

determine from which distribution the tuple is sampled.

The ordinary case analogue of the following problem is trivially solvable in polynomial time. Its supposed difficulty in the supersingular case is at the heart of the security of our identification scheme.

Problem D.12 (Decisional Supersingular Product (DSSP) problem)**.** Given a degree $\ell_A^{e_A}$ isogeny $\phi : E_0 \to E_3$ and a tuple sampled with probability $1/2$ from one of the following two distributions:

- (E_1, E_2, ϕ'), where the product $E_1 \times E_2$ is chosen at random among those $\ell_B^{e_B}$-isogenous to $E_0 \times E_3$, and where $\phi' : E_1 \to E_2$ is an isogeny of degree $\ell_A^{e_A}$, and

- (E_1, E_2, ϕ'), where E_1 is chosen at random among the curves having the same cardinality as E_0, and $\phi' : E_1 \to E_2$ is a random isogeny of degree $\ell_A^{e_A}$,

determine from which distribution the tuple is sampled.

We conjecture that these problems are computationally infeasible, in the sense that for any polynomial-time solver algorithm, the advantage of the algorithm is a negligible function of the security parameter $\log p$. The resulting security assumptions are referred to as the DSSI assumption, CSSI assumption, etc.

Hardness of the underlying assumptions

Given a CSSI (respectively, SSCDH) solver, it is trivial to solve SSCDH (respectively, SSDDH). It is also trivial to solve SSDDH given a DSSI solver. There are no known reductions in the other direction, and given that the corresponding question of equivalence for discrete logarithms and Diffie-Hellman has not yet been completely resolved in all cases, it is reasonable to assume that the question of equivalence of CSSI, SSCDH, and SSDDH is at least hard to resolve. For the purposes of this discussion, we will presume that DSSI and CSSI are equivalent to SSDDH. Concerning DSSP, there is an evident reduction to DSSI. However, it seems reasonable to assume that DSSP is easier than the latter.

In the context of cryptography, the problem of computing an isogeny between isogenous supersingular curves was first considered by Galbraith [Gal99] in 1999. The first published cryptographic primitive based on supersingular isogeny graphs is the hash function proposal of Charles et al. [CGL09], which remains unbroken to date (the cryptanalysis of [PLQ08] applies only to the LPS graph-based hash function from [CGL09], and not to the supersingular isogeny graph-based hash functions). The fastest known algorithm for finding isogenies between supersingular curves in general takes $O(\sqrt{p}\log^2 p)$ time [CGL09, §5.3.1]; however our problem is less general because the degree of the isogeny is known in advance and is smooth. In addition, the distribution of isogenous curves obtained from taking kernels of the form $\langle [m_A]P_A + [n_A]Q_A\rangle$ is not quite uniform: a simple calculation against Proposition D.1 indicates that a sequence of e_A isogenies of degree ℓ_A falls short of the length needed to ensure uniform mixing, regardless of the value of p. Since we are the first to propose using isogenies of this type, there is no existing literature addressing the security of the isogenies of the special form that we propose.

There are easy exponential attacks against DSSI and CSSI that improve upon exhaustive search. To find an isogeny of degree $\ell_A^{e_A}$ between E and E_A, an attacker builds two trees of all curves isogenous to E (respectively, E_A) via isogenies of degree $\ell_A^{e_A/2}$. Once the trees are built, the attacker tries to find a curve lying in both trees. Since the degree of the isogeny ϕ_A is $\sim\sqrt{p}$ (much shorter than the size of the isogeny graph), it is unlikely that there will be more than one isogeny path—and thus more than one match—from E to E_A. Given two functions $f : A \to C$ and $g : B \to C$ with domain of equal size, finding a pair (a,b) such that $f(a) = g(b)$ is known as the *claw problem* in complexity theory. The claw problem can obviously be solved in $O(|A| + |B|)$ time and $O(|A|)$ space on a classical computer by building a hash table holding $f(a)$ for any $a \in A$ and looking for hits for $g(b)$ where $b \in B$. This gives a $O(\ell_A^{e_A/2}) = O(\sqrt[4]{p})$ classical attack against our cryptosystems. With a quantum computer, one can do better using the algorithm in [Tan09], which has complexity $O(\sqrt[3]{|A||B|})$, thus giving an $O(\ell_A^{e_A/3}) = O(\sqrt[6]{p})$ quantum attack against our cryptosystems. These complexities are optimal for a black-box claw attack [Zha05].

We consider the question of whether the auxiliary data points $\phi_A(P_B)$ and $\phi_A(Q_B)$ might assist an adversary in determining ϕ_A. Since (P_B, Q_B) forms a basis for $E_0[\ell_B^{e_B}]$, the values $\phi_A(P_B)$ and $\phi_A(Q_B)$ allow the adversary to compute ϕ_A on all of $E_0[\ell_B^{e_B}]$. This is because any element of $E_0[\ell_B^{e_B}]$ is a (known) linear combination of P_B and Q_B (known since extended discrete logarithms are easy [Tes99]). However, there does not appear to be any way to use this capability to determine ϕ_A. Even on a quantum computer, where finding abelian hidden subgroups is easy, there is no hidden subgroup to find, since ϕ_A has degree $\ell_A^{e_A}$, and thus does not annihilate any point in $E_0[\ell_B^{e_B}]$ other than

the identity. Of course, if one could evaluate ϕ_A on arbitrary points of $E_0[\ell_A^{e_A}]$, then a quantum computer could easily break the scheme, and indeed in this case the scheme is also easily broken classically by using a few calls to the oracle to compute a generator of the kernel of the dual isogeny $\hat{\phi}_A$. However, it does not seem possible to translate the values of ϕ_A on $E_0[\ell_B^{e_B}]$ into values on $E_0[\ell_A^{e_A}]$.

For both ordinary and supersingular curves, there is a natural bijection between isogenies (up to isomorphism) and (left) ideals in the endomorphism ring. In the ordinary case the endomorphism ring is commutative, and ideal classes form a finite abelian group. This property has been used by Childs et al. [CJS14] to solve the ordinary analogue of CSSI in quantum subexponential time. It is natural to ask whether their algorithm can be adapted to the supersingular setting. Here the endomorphism ring is a maximal order in a noncommutative quaternion algebra, and the left ideal classes do not form a group at all (though they do form a groupoid). Since the algorithm of Childs et al. depends crucially on the properties of abelian groups, we believe that no reasonable variant of this strategy would apply.

The same correspondence between isogenies and ideals can be applied to DSSP. Indeed, deciding DSSP amounts to deciding whether the ideals S, S' associated to ϕ, ϕ' are conjugated, i.e., whether there exists a left ideal $R \in \mathrm{End}(E_0)$ such that $S = RS'R^{-1}$. Although it can be hoped that deciding conjugacy of ideal classes in the quaternion algebra $\mathbb{Q}_{p,\infty}$ is feasible, we are still faced with the problem that the best known algorithms to compute the endomorphism rings of supersingular curves are exponential in $\log p$ [Koh96; Cer04; Bel08]. Hence, we deem DSSP secure given the current knowledge.

The fact that it is possible to obtain a zero-knowledge identification scheme from CSSI comes as no surprise, since it is well known that a zero-knowledge protocol can be obtained from any problem in NP [GMW91]. Nevertheless, the generic construction is not very efficient, and many efforts have been made to obtain efficient *ad-hoc* schemes from NP-complete problems [Sha89; Ste94b; Ste94a; Poi95]. While the security of most of these schemes is based on two solid assumptions, namely that P $\neq$ NP and that *secure commitment schemes* exist, our identification scheme stands on a much weaker ground: the CSSI and DSSP problems. As performances go, it is reasonable to assume that our scheme will be some orders of magnitude slower than the best zero-knowledge protocols. We can thus conclude that our scheme is of a purely theoretical and pedagogical interest. Yet it is remarkable that an efficient identification scheme based on graphs of supersingular isogenies simply exists, while the analogous construction for ordinary curves is trivially broken and no other identification scheme is currently known to work in that case [Sto10].

D.6 Security proofs

In this section we state formal security reductions relating the security of our protocols to the hardness of the appropriate underlying isogeny computation problem. The security proofs are routine, but tedious, and contain little original contribution on our part. For this reason, we only prove a representative selection of our theorems.

The statements of the theorems are as follows:

Theorem D.13. *If the SSDDH assumption holds, then the key-agreement protocol of Section D.3 is session-key secure in the authenticated-links adversarial model of Canetti and Krawczyk [CK01].*

Theorem D.14. *If the SSDDH assumption holds, and the hash function family $\mathscr{H}$ is entropy-smoothing, then the public-key cryptosystem of Section D.3 is IND-CPA.*

Theorem D.15. *Under the CSSI and DSSP assumptions, the identification scheme of Section D.3 is zero-knowledge.*

Proof of Theorem D.13

The proofs of Theorems D.13 and D.14 are easily adapted from the corresponding proofs given by Stolbunov [Sto09]. As an illustration of the proof techniques, we provide a proof of Theorem D.13.

We recall the definition of session-key security in the authenticated-links adversarial model of Canetti and Krawczyk [CK01]. We consider a finite set of *parties* $P_1, P_2, \ldots, P_n$ modeled by probabilistic Turing machines. The adversary $\mathscr{I}$, also modeled by a probabilistic Turing machine, controls all communication, with the exception that the adversary cannot inject or modify messages (except for messages from corrupted parties or sessions), and any message may be delivered at most once. Parties give outgoing messages to the adversary, who has control over their delivery via the Send query. Parties are activated by Send queries, so the adversary has control over the creation of protocol sessions, which take place within each party. Two sessions s and s' are *matching* if the outgoing messages of one are the incoming messages of the other, and vice versa.

We allow the adversary black-box access to the queries SessionStateReveal, SessionKeyReveal, and Corrupt. The $\mathsf{SessionStateReveal}(\mathfrak{s})$ query allows the adversary to obtain the contents of the session state, including any secret information. The query is noted and $\mathfrak{s}$ produces no further output. The $\mathsf{SessionKeyReveal}(\mathfrak{s})$ query enables the adversary to obtain the session key for the specified session $\mathfrak{s}$, so long as $\mathfrak{s}$ holds a session key. The $\mathsf{Corrupt}(\mathsf{P_i})$ query allows the adversary to take over the party P_i, i.e., the adversary has access to all information in P_i's memory, including long-lived keys and any session-specific information still stored. A corrupted party produces no further output. We say a session $\mathfrak{s}$ with owner P_i is *locally exposed* if the adversary has issued $\mathsf{SessionKeyReveal}(\mathfrak{s})$, $\mathsf{SessionStateReveal}(\mathfrak{s})$, or $\mathsf{Corrupt}(\mathsf{P_i})$ before $\mathfrak{s}$ is expired. We say $\mathfrak{s}$ is *exposed* if $\mathfrak{s}$ or its matching session have been locally exposed, and otherwise we say $\mathfrak{s}$ is *fresh*.

We allow the adversary $\mathscr{I}$ a single $\mathsf{Test}(\mathfrak{s})$ query, which can be issued at any stage to a completed, fresh, unexpired session $\mathfrak{s}$. A bit b is then picked at random. If $b = 0$, the test oracle reveals the session key, and if $b = 1$, it generates a random value in the key space. $\mathscr{I}$ can then continue to issue queries as desired, with the exception that it cannot expose the test session. At any point, the adversary can try to guess b. Let $\mathrm{GoodGuess}^{\mathscr{I}}(k)$ be the event that $\mathscr{I}$ correctly guesses b, and define

$$\mathrm{Advantage}^{\mathscr{I}}(k) = \max\left\{0, \left|\Pr[\mathrm{GoodGuess}^{\mathscr{I}}(k)] - \frac{1}{2}\right|\right\},$$

where k is a security parameter.

The definition of security is as follows:

Definition D.16. A key exchange protocol Π in security parameter k is said to be *session-key secure* in the authenticated-links adversarial model of Canetti and Krawczyk if for any polynomial-time adversary $\mathscr{I}$,

Algorithm 12 SSDDH distinguisher

Input: $E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A), \phi_B(Q_A), E$

1: $r \xleftarrow{R} \{1, \ldots, k\}$, where k is an upper bound on the number of sessions activated by $\mathscr{I}$ in any interaction.
2: Invoke $\mathscr{I}$ and simulate the protocol to $\mathscr{I}$, except for the r-th activated protocol session.
3: For the r-th session, let Alice send A, i, E_A, $\phi_A(P_B)$, $\phi_A(Q_B)$ to Bob, and let Bob send B, i, E_B, $\phi_B(P_A)$, $\phi_B(Q_A)$ to Alice, where i is the session identifier.
4: **if** the r-th session is chosen by $\mathscr{I}$ as the test session **then**
5: Provide $\mathscr{I}$ as the answer to the test query.
6: $d \leftarrow \mathscr{I}$'s output.
7: **else**
8: $d \xleftarrow{R} \{0,1\}$.
9: **end if**

Output: d

1. If two uncorrupted parties have completed matching sessions, these sessions produce the same key as output;
2. $\text{Advantage}^{\mathscr{I}}(k)$ is negligible.

Proof of Theorem D.13. We adapt the proof given by Canetti and Krawczyk [CK01, §5.1] for two-party Diffie-Hellman over $\mathbb{Z}_q^*$. A similar strategy was used by Stolbunov [Sto09] in the case of ordinary elliptic curves.

It has been shown in Section D.3 that two uncorrupted parties in matching sessions output the same session key, and thus the first part of Definition D.16 is satisfied. To show that the second part of the definition is satisfied, assume that there is a polynomial-time adversary $\mathscr{I}$ with a non-negligible advantage ε. We claim that Algorithm 12 forms a polynomial-time distinguisher for SSDDH having non-negligible advantage.

To prove the claim, we must show that Algorithm 12 has non-negligible advantage (it is clear that it runs in polynomial time). We consider separately the cases where the r-th session is (respectively, is not) chosen by $\mathscr{I}$ as the test session. If the r-th session is not the test session, then Algorithm 12 outputs a random bit, and thus its advantage in solving the SSDDH is 0. If the r-th session is the test session, then $\mathscr{I}$ will succeed with advantage ε, since the simulated protocol provided to $\mathscr{I}$ is indistinguishable from the real protocol. Since the latter case occurs with probability $1/k$, the overall advantage of the SSDDH distinguisher is ε/k, which is non-negligible. □

Proof of Theorem D.15

Using classical techniques from [GMW91; FFS88], the theorem is proved in three steps, known by the names of *completeness*, *soundness* and *zero-knowledge*.

Proof of Theorem D.15 (sketch). Completeness is obvious: using the algorithms of Section D.4, Peggy can always compute the diagram (D.3) in polynomial time and make Vic accept.

To prove soundness, we let Charles be any polynomially bounded adversary capable of convincing Vic with a non-negligible probability. We use Charles as a black-box for which we can control the random coin tosses. By restarting it a polynomial number of

times with the same random input, and by asking each time a different set of questions, we learn with overwhelming probability a diagram

$$\begin{array}{ccc} E & & E/\langle S\rangle \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle\psi'} \\ E/\langle R\rangle & \xrightarrow{\phi'} & E/\langle S,R\rangle \end{array} \tag{D.21}$$

It is then straightforward to compute the secret. Let R be a generator of the kernel of ϕ'. Using the theorem of the dual isogeny it is easy to compute $\hat{\psi}$, then $\langle\hat{\psi}(R)\rangle$ is the kernel of an isogeny $E \to E/\langle S\rangle$ of degree $\ell_A^{e_A}$. This contradicts the CSSI assumption.

To prove zero-knowledge, we use a *cheating verifier* (CV) as a black-box to construct a *simulator* (S). At each iteration, S makes a (uniformly) random guess at what the next question by V will be.

If S guesses $b = 0$, it chooses a random primitive $\ell_B^{e_B}$-torsion point $R \in E$ and computes $\phi(R)$ (recall that the action of ϕ on $E[\ell_B^{e_B}]$ is part of the public data). Then it constructs the isogenies $\psi : E \to E/\langle R\rangle$ and $\psi' : E/\langle S\rangle \to E/\langle S,R\rangle$

$$\begin{array}{ccc} E & \overset{\phi}{\dashrightarrow} & E/\langle S\rangle \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle\psi'} \\ E/\langle R\rangle & & E/\langle S,R\rangle \end{array} \tag{D.22}$$

and sends $E_1 = E/\langle R\rangle$ and $E_2 = E/\langle S,R\rangle$ to CV.

If S guesses $b = 1$, it chooses a random supersingular curve E' having the same cardinality as E, and a random primitive $\ell_A^{e_A}$-torsion point $R \in E'$. Then, it constructs the isogeny $\phi' : E' \to E'/\langle R\rangle$

$$\begin{array}{ccc} E & \overset{\phi}{\dashrightarrow} & E/\langle S\rangle \\ \\ E' & \xrightarrow{\phi'} & E'/\langle R\rangle \end{array} \tag{D.23}$$

and sends $E_1 = E'$ and $E_2 = E'/\langle R\rangle$ to CV.

If CV does not ask the expected question, S simply discards the attempt and restarts. If CV asks the expected question, S writes (E_1, E_2, b, R) on its output. S stops whenever CV rejects, or after m successful interactions with CV.

To prove zero-knowledge, we must show that S runs in polynomial time and that its output is polynomially indistinguishable from the transcript of a conversation between CV and Peggy.

To show that S runs in polynomial time, it is enough to show that, at any iteration, for any guess b made by S, the probability that CV asks question $1-b$ is exponentially close to $1/2$. Suppose this were not the case, then CV can be used as an oracle for DSSP.

To prove indistinguishability, using the *hybrid* technique of [GMW91, Claim 4.2], it is enough to prove that no polynomial-time distinguisher exists for a single round of the identification scheme. It is obvious that no such distinguisher can exist for questions of type $b = 0$, because the outputs of S and Peggy are identical in this case. Suppose,

Table D.3: Benchmarks for various key sizes. Alice uses $\ell = 2$, Bob uses $\ell = 3$.

	tuned 2/1			balanced	
	512 bits	768 bits	1024 bits	768 bits	1024 bits
Alice round 1	28.1 ms	65.7 ms	122 ms	66.8 ms	123 ms
Alice round 2	23.3 ms	54.3 ms	101 ms	55.5 ms	102 ms
Bob round 1	28.0 ms	65.6 ms	125 ms	67.1 ms	128 ms
Bob round 2	22.7 ms	53.7 ms	102 ms	55.1 ms	105 ms

now, that there exists a distinguisher D which, on input $\phi' : E_1 \to E_2$, can tell with non-negligible probability whether it comes from S or from a conversation between CV and Peggy, then D can be used as an oracle for DSSP. □

D.7 Implementation results and example

To evaluate the performance of our schemes, we implemented the key exchange protocol in the computer algebra system Sage [Sage] using a mixed C/Cython/Sage architecture. This allows us to access the large palette of number theoretic algorithms distributed with Sage, while still permitting very efficient code in C/Cython for the critical parts such as the algorithms of Section D.4. The source code can be downloaded from De Feo's web page.

Arithmetic in $\mathbb{F}_{p^2}$ is written in C. We use the library GMP for arithmetic modulo p. The field $\mathbb{F}_{p^2}$ is implemented as $\mathbb{F}_{p^2}[X]/(X^2+1)$ (this requires $p = 3 \bmod 4$); using this representation, one multiplication in $\mathbb{F}_{p^2}$ requires three multiplications in $\mathbb{F}_p$, one squaring requires two multiplications in $\mathbb{F}_p$, and one inversion requires one inversion, two squarings, and two multiplications in $\mathbb{F}_p$. Our experiments show that, for the sizes we are interested in, $I = 10M$ and $S = 0.8M$.

The computation of the optimal strategies as described in Section D.4 is done in pure Python, using a dynamic programming algorithm.

We implemented the key exchange algorithm in C for $\ell = 2, 3$, and in Cython for any ℓ. Our experiments show that, in the C implementation, the ratio between scalar multiplications and isogeny evaluations is about 2, which is consistent with the predictions made in Table D.1. We used this ratio to tune the optimal strategies for $\ell = 2, 3$: starting from 768 bits, a gain over the balanced strategy, comprised between 1% and 3%, starts getting noticeable. The performances of 3-isogenies are comparable to those of 2-isogenies, thus contradicting the prediction made by Table D.2; this is explained by the operations not accounted for in Table D.2, such as the computation of the isogenies and of the isogenous curves (3-isogenies gain a factor of about $\log_3 2$ on these operations). This suggests that well optimized 4-isogeny formulas may eventually outperform 2-isogenies.

Finally, the parameter generation is implemented in plain Sage. Because Sage is a collection of many open source mathematical systems, various of its subsystems are involved in this last part. Of these, Pari [PARI] plays an important role because it is used to compute Hilbert class polynomials and to factor polynomials over finite fields.

All tests ran on a 2.4 GHz Opteron running in 64-bit mode. The results are summarized in Table D.3. At the quantum 128-bit security level (768-bit p), our numbers improve upon Stolbunov's reported performance figures [Sto10, Table 1] by over three orders of magnitude (.066 seconds vs. 229 seconds). This is the highest security level appearing in [Sto10, Table 1], so comparisons at higher levels are difficult.

Nevertheless, it seems safe to assume that the improvement is even greater at the 256-bit security level. Our results demonstrate that the proposed scheme is practical.

Example

As a convenience, we provide an example calculation of a key-exchange transaction. Let $\ell_A = 2$, $\ell_B = 3$, $e_A = 63$, $e_B = 41$, and $f = 11$. We use the starting curve $E_0 : y^2 = x^3 + x$. For the torsion bases, we use

$$P_A = (237409306833625077410793642140789388589 7i + 252464670185239634930842532821820356969 3,$$
$$194486926041457420622915324351010478172 5i + 130909941321176707805523276846048341720 1)$$
$$P_B = (155671603365753087672852505928443176120 6i + 174740732959516524133513164729866065215,$$
$$345695620285202883552941999547591538848 3i + 197591287424745857265472071715575500556 6)$$

and $Q_A = \psi(P_A), Q_B = \psi(P_B)$, where $i = \sqrt{-1}$ in $\mathbb{F}_{p^2}$ and $\psi(x,y) = (-x, iy)$ is a distortion map [Jou02]. The secret values are

$$m_A = 2575042839726612324,\ n_A = 8801426132580632841,$$
$$m_B = 4558164392438856871,\ n_B = 20473135767366569910$$

The isogeny $\phi_A : E_0 \to E_A$ is specified by its kernel, and thus the curve E_A is only well defined up to isomorphism; its exact value may vary depending on the implementation. In our case, the curve is $E_A : y^2 = x^3 + ax + b$ where

$$a = 428128245356224894562061821180718114127i + 214770800990771179013498662460467452576 9$$
$$b = 323035926720219746030433183517042405309 3i + 157726433648237019704536235910489488486 2$$

and the values of $\phi_A(P_B)$ and $\phi_A(Q_B)$ are

$$\phi_A(P_B) = (121624303795507829290097485944106602697 6i + 166629113680473868483263718767433090557 2,$$
$$313292160945399836185337294189350010792 3i + 282316493857354948561980003461685523 66)$$
$$\phi_A(Q_B) = (203972869442093051915573296501829191066 0i + 242209261432298811249293161552815572738 8,$$
$$168811581269435514554988923851045703427 2i + 137918598460824063891294889034973846753 6)$$

Similarly, in our implementation $E_B : y^2 = x^3 + ax + b$ is the curve with

$$a = 257472239809402296857831386188460894312 2i + 464507557149559062184174132571647427722$$
$$b = 286347890751308879214499831122977288619 7i + 176707803671410940579677706508986838675 3$$

and the values of $\phi_B(P_A)$ and $\phi_B(Q_A)$ are

$$\phi_B(P_A) = (251908600334797321477049915416254009818 1i + 145970297400960919872398112545754844087 2,$$
$$207205706793329259932692876625515508138 0i + 891622100638258849401618552145232311395)$$
$$\phi_B(Q_A) = (537939945228033932439214329827985436666 i + 369874160978813685588489568343190504844,$$
$$285386807397180839864966365216121532375 0i + 186973048005362414137237328279585869113 9)$$

The common j-invariant of $E_{AB} \cong E_{BA}$, computed by both Alice and Bob, is equal to

$$j(E_{AB}) = 143714549436265511916848280870211141374 4i + 833498096778386452951722285310592056351.$$

D.8 Conclusion

We propose a new family of conjecturally quantum-resistant public-key cryptographic protocols using isogenies between supersingular elliptic curves of smooth order. In order to compensate for the noncommutative endomorphism rings that arise in this setting, we introduce the idea of providing the images of torsion bases as part of the protocol. Against the fastest known attacks, the resulting key exchange scheme improves upon all previous isogeny-based schemes by orders of magnitude in performance at conventional security levels, making it the first practical isogeny-based public-key cryptosystem. Unlike prior such schemes, our proposal admits no known subexponential-time attacks even in the quantum setting.

D.9 References for "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies"

[Ant+06] Adrian Antipa, Daniel Brown, Robert Gallant, Rob Lambert, René Struik, and Scott Vanstone. "Accelerated Verification of ECDSA Signatures". In: *Selected Areas in Cryptography 2005*. Vol. 3897. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2006. Chap. 21, pp. 307–318. ISBN: 978-3-540-33108-7. DOI: 10.1007/11693383_21.

[Bel08] Juliana V. Belding. "Number Theoretic Algorithms for Elliptic Curves". PhD thesis. University of Maryland, 2008.

[Ber+08] Daniel Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. "Twisted Edwards Curves". In: *Progress in Cryptology — AFRICACRYPT 2008*. 2008, pp. 389–405. DOI: 10.1007/978-3-540-68164-9_26.

[BL07] Daniel J. Bernstein and Tanja Lange. *Explicit-Formulas Database*. 2007. URL: http://www.hyperelliptic.org/EFD/index.html.

[Bos+08] Alin Bostan, François Morain, Bruno Salvy, and Éric Schost. "Fast algorithms for computing isogenies between elliptic curves". In: *Mathematics of Computation* 77.263 (Sept. 2008), pp. 1755–1778. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-08-02066-8.

[Brö09] Reinier Bröker. "Constructing supersingular elliptic curves". In: *Journal of Combinatorics and Number Theory* 1.3 (2009), pp. 269–273. ISSN: 1942-5600.

[Cer04] Juan M. Cerviño. *On the Correspondence between Supersingular Elliptic Curves and maximal quaternionic Orders*. Apr. 2004. URL: http://arxiv.org/abs/math/0404538.

[CGL09] Denis X. Charles, Eyal Z. Goren, and Kristin E. Lauter. "Cryptographic Hash Functions from Expander Graphs". In: *Journal of Cryptology* 22.1 (Jan. 2009), pp. 93–113. ISSN: 0933-2790. DOI: 10.1007/s00145-007-9002-x.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. "Constructing elliptic curve isogenies in quantum subexponential time". In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29.

[CK01] Ran Canetti and Hugo Krawczyk. "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels". In: *EUROCRYPT*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 453–474. ISBN: 3-540-42070-3.

[Cou06] Jean-Marc Couveignes. *Hard Homogeneous Spaces*. Cryptology ePrint Archive, Report 2006/291. 2006. URL: https://eprint.iacr.org/2006/291.

[DSV03] Giuliana Davidoff, Peter Sarnak, and Alain Valette. *Elementary number theory, group theory, and Ramanujan graphs*. Vol. 55. London Mathematical Society Student Texts. Cambridge: Cambridge University Press, 2003. DOI: 10.1017/CBO9780511615825.

[ElG85] Taher ElGamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. In: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. “Zero-knowledge proofs of identity”. In: *Journal of Cryptology* 1.2 (June 1988), pp. 77–94. ISSN: 0933-2790. DOI: 10.1007/BF02351717.

[Gal99] Steven D. Galbraith. “Constructing Isogenies between Elliptic Curves Over Finite Fields”. In: *LMS Journal of Computation and Mathematics* 2 (1999), pp. 118–138. DOI: 10.1112/S1461157000000097.

[GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. “Extending the GHS Weil descent attack”. In: *Advances in cryptology–EUROCRYPT 2002 (Amsterdam)*. Vol. 2332. Lecture Notes in Computer Science. Berlin: Springer, 2002, pp. 29–44. ISBN: 3-540-43553-0.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. “Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems”. In: *Journal of the Association for Computing Machinery* 38.3 (July 1991), pp. 690–728. ISSN: 0004-5411. DOI: 10.1145/116825.116852.

[GS13] Steven D. Galbraith and Anton Stolbunov. “Improved algorithm for the isogeny problem for ordinary elliptic curves”. In: *Applicable Algebra in Engineering, Communication and Computing* 24.2 (June 2013), pp. 107–131. ISSN: 1432-0622. DOI: 10.1007/s00200-013-0185-0.

[JD11] David Jao and Luca De Feo. “Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies”. In: *Post-Quantum Cryptography*. Ed. by Bo-Yin Yang. Vol. 7071. Lecture Notes in Computer Science. Taipei, Taiwan: Springer Berlin / Heidelberg, 2011. Chap. 2, pp. 19–34. ISBN: 978-3-642-25404-8. DOI: 10.1007/978-3-642-25405-5_2.

[JMV09] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. “Expander graphs based on GRH with an application to elliptic curve cryptography”. In: *Journal of Number Theory* 129.6 (June 2009), pp. 1491–1504. ISSN: 0022314X. DOI: 10.1016/j.jnt.2008.11.006.

[Jou02] Antoine Joux. *The Weil and Tate pairings as building blocks for public key cryptosystems*. Berlin, 2002. DOI: 10.1007/3-540-45455-1_3.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential Power Analysis”. In: *Advances in Cryptology — CRYPTO’ 99*. Vol. 1666. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, Dec. 1999. Chap. 25, pp. 388–397. ISBN: 978-3-540-66347-8. DOI: 10.1007/3-540-48405-1_25.

[Koh96] David R. Kohel. “Endomorphism rings of elliptic curves over finite fields”. PhD thesis. University of California at Berkeley, 1996.

[LO77] Jeffrey C. Lagarias and Andrew M. Odlyzko. “Effective versions of the Chebotarev density theorem”. In: *Algebraic number fields: L-functions and Galois properties*. London: Academic Press, 1977, pp. 409–464.

[LPS88] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. “Ramanujan graphs”. In: *Combinatorica* 8.3 (1988). DOI: 10.1007/BF02126799.

[Lub94] Alexander Lubotzky. *Discrete groups, expanding graphs and invariant measures*. Vol. 125. Progress in Mathematics. Basel: Birkhäuser Verlag, 1994. ISBN: 978-3-0346-0332-4. DOI: 10.1007/978-3-0346-0332-4.

[Mes86] Jean-François Mestre. "La méthode des graphes. Exemples et applications". In: *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata, 1986)*. Nagoya: Nagoya University, 1986. URL: http://boxen.math.washington.edu/msri06/refs/mestre-method-of-graphs/mestre-fr.pdf.

[Mon87] Peter L. Montgomery. "Speeding the Pollard and Elliptic Curve Methods of Factorization". In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. ISSN: 00255718. DOI: 10.2307/2007888.

[OEI12] OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences*. http://oeis.org/A130715. 2012.

[PARI] *PARI/GP, version 2.8.0*. The PARI Group. Bordeaux, 2016. URL: https://pari.math.u-bordeaux.fr/.

[Piz90] Arnold K. Pizer. "Ramanujan graphs and Hecke operators". In: *Bulletin of the American Mathematical Society (N.S.)* 23.1 (1990). DOI: 10.1090/S0273-0979-1990-15918-X.

[Piz98] Arnold K. Pizer. "Ramanujan graphs". In: *Computational perspectives on number theory (Chicago, IL, 1995)*. Vol. 7. AMS/IP Stud. Adv. Math. Providence, RI: Amer. Math. Soc., 1998.

[PLQ08] Christophe Petit, Kristin Lauter, and Jean-Jacques Quisquater. "Full Cryptanalysis of LPS and Morgenstern Hash Functions". In: *Proceedings of the 6th international conference on Security and Cryptography for Networks*. SCN '08. Berlin, Heidelberg: Springer-Verlag, 2008. DOI: 10.1007/978-3-540-85855-3_18.

[Poi95] David Pointcheval. "A New Identification Scheme Based on the Perceptrons Problem". In: *Advances in Cryptology — EUROCRYPT '95*. Vol. 921. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1995. Chap. 26, pp. 319–328. ISBN: 978-3-540-59409-3. DOI: 10.1007/3-540-49264-X_26. URL: http://dx.doi.org/10.1007/3-540-49264-X%5C_26.

[RS06] Alexander Rostovtsev and Anton Stolbunov. *Public-key cryptosystem based on isogenies*. Cryptology ePrint Archive, Report 2006/145. Apr. 2006. URL: http://eprint.iacr.org/2006/145/.

[Sage] *SageMath, the Sage Mathematics Software System (Version 8.0)*. The Sage Developers. 2018. URL: https://www.sagemath.org.

[Sar90] Peter Sarnak. *Some applications of modular forms*. Vol. 99. Cambridge Tracts in Mathematics. Cambridge: Cambridge University Press, 1990.

[Sha89] Adi Shamir. "An efficient identification scheme based on permuted kernels (extended abstract)". In: *Proceedings on Advances in cryptology*. CRYPTO '89. New York, NY, USA: Springer-Verlag New York, Inc., 1989, pp. 606–609. ISBN: 0-387-97317-6.

[Sil92] Joseph H. Silverman. *The arithmetic of elliptic curves*. Vol. 106. Graduate Texts in Mathematics. Corrected reprint of the 1986 original. New York: Springer-Verlag, 1992, pp. xii+400. ISBN: 0-387-96203-4.

[Sol01] Jerome A. Solinas. *Low-Weight Binary Representations for Pairs of Integers*. Tech. rep. National Security Agency, USA, 2001.

[Ste94a] Jacques Stern. "A new identification scheme based on syndrome decoding". In: *Advances in Cryptology — CRYPTO' 93*. Vol. 773. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1994. Chap. 2, pp. 13–21. ISBN: 978-3-540-57766-9. DOI: 10.1007/3-540-48329-2_2. URL: http://dx.doi.org/10.1007/3-540-48329-2%5C_2.

[Ste94b] Jacques Stern. "Designing Identification Schemes with Keys of Short Size". In: *Advances in Cryptology — CRYPTO '94*. Vol. 839. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1994. Chap. 18, pp. 164–173. ISBN: 978-3-540-58333-2. DOI: 10.1007/3-540-48658-5_18. URL: http://dx.doi.org/10.1007/3-540-48658-5%5C_18.

[Sto09] Anton Stolbunov. "Reductionist Security Arguments for Public-Key Cryptographic Schemes Based on Group Action". In: *Norsk informasjonssikkerhetskonferanse (NISK)*. Ed. by Stig F. Mjølsnes. 2009.

[Sto10] Anton Stolbunov. "Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves". In: *Advances in Mathematics of Communications* 4.2 (2010).

[Tan09] Seiichiro Tani. "Claw finding algorithms using quantum walk". In: *Theoretical Computer Science* 410.50 (2009), pp. 5285–5297.

[Tat66] John Tate. "Endomorphisms of abelian varieties over finite fields". In: *Inventiones mathematicae* 2.2 (Apr. 1966), pp. 134–144. ISSN: 1432-1297. DOI: 10.1007/BF01404549.

[Tes99] Edlyn Teske. "The Pohlig-Hellman Method Generalized for Group Structure Computation". In: *Journal of Symbolic Computation* 27.6 (1999), pp. 521–534. ISSN: 0747-7171. DOI: 10.1006/jsco.1999.0279.

[Vél71] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus de l'Académie des Sciences de Paris* 273 (1971), pp. 238–241.

[Zha05] Shengyu Zhang. "Promised and Distributed Quantum Search Computing and Combinatorics". In: *Proceedings of the Eleventh Annual International Conference on Computing and Combinatorics*. Vol. 3595. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2005. Chap. 44, pp. 430–439. ISBN: 978-3-540-28061-3. DOI: 10.1007/11533719_44.

E Towards practical key exchange from ordinary isogeny graphs

Abstract

We revisit the ordinary isogeny-graph based cryptosystems of Couveignes and Rostovtsev–Stolbunov, long dismissed as impractical. We give algorithmic improvements that accelerate key exchange in this framework, and explore the problem of generating suitable system parameters for contemporary pre- and post-quantum security that take advantage of these new algorithms. We also prove the session-key security of this key exchange in the Canetti–Krawczyk model, and the IND-CPA security of the related public-key encryption scheme, under reasonable assumptions on the hardness of computing isogeny walks. Our systems admit efficient key-validation techniques that yield CCA-secure encryption, thus providing an important step towards efficient post-quantum non-interactive key exchange (NIKE).

E.1 Introduction

Isogeny-based protocols form one of the youngest and least-explored families of post-quantum candidate cryptosystems. The best-known isogeny-based protocol is Jao and De Feo's SIDH key exchange [JD11], from which the NIST candidate key-encapsulation mechanism SIKE was derived [SIKE; Nat16]. SIDH was itself inspired by earlier key-exchange constructions by Couveignes [Cou06] and Rostovtsev and Stolbunov [RS06; Sto09; Sto10], which were widely considered unwieldy and impractical.

Indeed, the origins of isogeny-based cryptography can be traced back to Couveignes' "Hard Homogeneous Spaces" manuscript, that went unpublished for ten years before appearing in [Cou06]. A *principal homogeneous space* (PHS) for a group G is a set X with an action of G on X such that for any $x, x' \in X$, there is a unique $g \in G$ such that $g \cdot x = x'$. Equivalently, the map $\varphi_x : g \mapsto g \cdot x$ is a bijection between G and X for any $x \in X$. Couveignes defines a *hard homogeneous space* (HHS) to be a PHS where the action of G on X is efficiently computable, but inverting the isomorphism φ_x is computationally hard for any x.

Algorithm 1: Key generation for cryptosystems in an HHS X for a group G, with a fixed "base point" x_0 in X.

Input: ()
Output: A private-public keypair $(g, x) \in G \times X$ s.t. $x = g \cdot x_0$

```
function KeyGen()
    g ← Random(G)      // g is sampled uniformly at random from G
    x ← g · x_0
    return (g, x)
```

Any HHS X for an *abelian* group G can be used to construct a key exchange based on the hardness of inverting φ_x, as shown in Algorithms 1 and 2. If Alice and Bob have keypairs (g_A, x_A) and (g_B, x_B), respectively, then the commutativity of G lets them derive a shared secret

$$\mathtt{DH}(g_A, x_B) = g_A \cdot (g_B \cdot x_0) = g_B \cdot (g_A \cdot x_0) = \mathtt{DH}(g_B, x_A)\,.$$

The analogy with classic group-based Diffie–Hellman is evident.

Algorithm 2: Diffie–Hellman in an HHS X for a group G

Input: A private key $g_A \in G$ and a public key $x_B \in X$, each generated by calls to `KeyGen`

Output: A shared secret value $k \in X$

function `DH`(g_A, x_B)
  $k \leftarrow g_A \cdot x_B$
  return k

For example, if $X = \langle x \rangle$ is cyclic of order p and $G = (\mathbb{Z}/p\mathbb{Z})^*$ acts on $X \setminus \{1\}$ by $g \cdot x = x^g$, then inverting φ_x is the discrete logarithm problem (DLP) in X. But inverting φ_x for other homogeneous spaces may not be related to any DLP, and might resist attacks based on Shor's quantum algorithm. Similar ideas have occasionally appeared in the literature in different forms [Ko+00; MMR07].

Couveignes viewed HHS chiefly as a general framework encompassing various Diffie–Hellman-like systems. Nevertheless, he suggested using a specific HHS based on the theory of complex multiplication of elliptic curves, in a sense generalizing Buchmann and Williams' class-group-based Diffie–Hellman key exchange [BW88]. Independently, Rostovtsev and Stolbunov proposed in [RS06] a public key encryption scheme based on the same HHS. Later, Stolbunov [Sto10] derived more protocols from this, including an interactive key exchange scheme similar to Algorithm 2. Rostovtsev and Stolbunov's proposal deviates from the HHS paradigm in the way random elements of G are sampled, as we will explain in §E.3. This makes the primitive less flexible, but also more practical.

Rostovtsev and Stolbunov advertised their cryptosystems as potential post-quantum candidates, leading Childs, Jao and Soukharev to introduce the first subexponential quantum algorithm capable of breaking them [CJS14]. Hence, being already slow enough to be impractical in a classical security setting, their primitive appeared even more impractical in a quantum security setting.

But the Couveignes–Rostovtsev–Stolbunov primitive (CRS) has some important advantages over SIDH which make it worth pursuing. Unlike SIDH, CRS offers efficient and safe public key validation, making it suitable for non-interactive key exchange (NIKE). Further, CRS does not suffer from some of the potential cryptographic weaknesses that SIDH has, such as short paths and the publication of image points.

This paper aims to improve and modernize the CRS construction, borrowing techniques from SIDH and point-counting algorithms, to the point of making it usable in a post-quantum setting. Our main contributions are in §§E.3–E.4, where we present a new, more efficient way of computing the CRS group action, and in §E.5, where we give precise classic and quantum security estimates, formalize hardness assumptions, and sketch security proofs in stronger models than those previously considered. In §E.6 we present a proof-of-concept implementation and measure its performance. While the final result is far from competitive, we believe it constitutes progress towards a valid isogeny-based alternative to SIDH.

CSIDH. While preparing this paper we were informed of recent work by Castryck, Lange, Martindale, Panny, and Renes, introducing CSIDH, an efficient post-quantum primitive based on CRS [Cas+18]. Their work builds upon the ideas presented in §§E.3–E.4, using them in a different homogeneous space where they apply effortlessly.

Their breakthrough confirms that, if anything, our techniques were a fundamental step towards the first practical post-quantum non-interactive key exchange protocol.

Side channel awareness. The algorithms we present here are not intended to provide any protection against basic side-channel attacks. Uniform and constant-time algorithms for arbitrary-degree isogeny computations are an interesting open problem, but they are beyond the scope of this work.

E.2 Isogenies and complex multiplication

We begin by recalling some basic facts on isogenies of elliptic curves over finite fields. For an in-depth introduction to these concepts, we refer the reader to [Sil92]. For a general overview of isogenies and their use in cryptography, we suggest [De 17].

Isogenies between elliptic curves

In what follows $\mathbb{F}_q$ is a finite field of characteristic p with q elements, and $\overline{\mathbb{F}}_q$ is its algebraic closure. Let E and E' be elliptic curves defined over $\mathbb{F}_q$. A homomorphism $\phi : E \to E'$ is an algebraic map sending 0_E to $0_{E'}$; it induces a group homomomorphism from $E(\overline{\mathbb{F}}_q)$ to $E'(\overline{\mathbb{F}}_q)$ [Sil92, p. III.4]. An *endomorphism* is a homomorphism from a curve to itself. The endomorphisms of E form a ring $\mathrm{End}(E)$, with the group law on E for addition and composition for multiplication. The simplest examples of endomorphisms are the scalar multiplications $[m]$ (mapping P to the sum of m copies of P) and the *Frobenius* endomorphism

$$\begin{aligned} \pi : E &\longrightarrow E\,, \\ (x,y) &\longmapsto (x^q, y^q)\,. \end{aligned}$$

As an element of $\mathrm{End}(E)$, Frobenius satisfies a quadratic equation $\pi^2 + q = t\pi$. The integer t (the *trace*) fully determines the order of E as $\#E(\mathbb{F}_q) = q + 1 - t$. A curve is called *supersingular* if p divides t, *ordinary* otherwise.

An *isogeny* is a non-zero homomorphism of elliptic curves. The degree of an isogeny is its degree as an algebraic map, so for example the Frobenius endomorphism π has degree q, and the scalar multiplication $[m]$ has degree m^2. Isogenies of degree ℓ are called ℓ-isogenies. The kernel $\ker\phi$ of ϕ is the subgroup of $E(\overline{\mathbb{F}}_q)$ that is mapped to $0_{E'}$. An isogeny ϕ is *cyclic* if $\ker\phi$ is a cyclic group.

An *isomorphism* is an isogeny of degree 1. An *isomorphism class* of elliptic curves is fully determined by their common *j-invariant* in $\overline{\mathbb{F}}_q$. If any curve in the isomorphism class is defined over $\mathbb{F}_q$, then its j-invariant is in $\mathbb{F}_q$.

Any isogeny can be factored as a composition of a *separable* and a *purely inseparable* isogeny. *Purely inseparable* isogenies have trivial kernel, and degree a power of p. *Separable* isogenies include all isogenies of degree coprime to p. Up to isomorphism, separable isogenies are in one-to-one correspondence with their kernels: for any finite subgroup $G \subset E$ of order ℓ there is an elliptic curve E/G and an ℓ-isogeny $\phi : E \to E/G$ such that $\ker\phi = G$, and the curve and isogeny are unique up to isomorphism. In particular, if ϕ is separable then $\deg\phi = \#\ker\phi$. It is convenient to encode $\ker\phi$ as the polynomial whose roots are the x-coordinates of the points in $\ker\phi$, called the *kernel polynomial* of ϕ.

For any ℓ-isogeny $\phi : E \to E'$, there is a unique ℓ-isogeny $\hat{\phi} : E' \to E$ such that $\phi \circ \hat{\phi} = [\ell]$ on E' and $\hat{\phi} \circ \phi = [\ell]$ on E. We call $\hat{\phi}$ the *dual* of ϕ. This shows that being

ℓ-*isogenous* is a symmetric relation, and that being isogenous is an equivalence relation. Further, a theorem of Tate states that two curves are isogenous over $\mathbb{F}_q$ if and only if they have the same number of points over $\mathbb{F}_q$.

Isogeny graphs

Isogeny-based cryptosystems are based on *isogeny graphs*. These are (multi)-graphs whose vertices are elliptic curves up to isomorphism, and whose edges are isogenies between them (again up to isomorphism). The use of isogeny graphs for algorithmic applications goes back to Mestre and Oesterlé [Mes86], followed notably by Kohel [Koh96], and has been continued by many authors [Gal99; FM02; GHS02; Mir+06; JMV09].

We write $E[\ell]$ for the subgroup of ℓ-torsion points of $E(\overline{\mathbb{F}}_q)$. If ℓ is coprime to p, then $E[\ell]$ is isomorphic to $(\mathbb{Z}/\ell\mathbb{Z})^2$. Furthermore, if ℓ is prime then $E[\ell]$ contains exactly $\ell+1$ cyclic subgroups of order ℓ; it follows that, over $\overline{\mathbb{F}}_q$, there are exactly $\ell+1$ distinct (non-isomorphic) separable ℓ-isogenies from E to other curves. Generically, a connected component of the ℓ-isogeny graph over $\overline{\mathbb{F}}_q$ will be an infinite $(\ell+1)$-regular graph (a notable exception is the finite connected component of *supersingular* curves, used in SIDH and related protocols).

We now restrict to isogenies defined over $\mathbb{F}_q$. If E and E' are elliptic curves over $\mathbb{F}_q$, then an isogeny $\phi : E \to E'$ is defined over $\mathbb{F}_q$ (up to a twist of E') if and only if the Frobenius endomorphism π on E stabilizes $\ker\phi$. We emphasize that the points in $\ker\phi$ need not be defined over $\mathbb{F}_q$ themselves.

For the vertices of the $\overline{\mathbb{F}}_q$-isogeny graph we use j-invariants, which classify elliptic curves up to $\overline{\mathbb{F}}_q$-isomorphism; but in the sequel we want to work up to $\mathbb{F}_q$-isomorphism, a stronger equivalence. If E and $\tilde{E}$ are not $\mathbb{F}_q$-isomorphic but $j(E) = j(\tilde{E})$, then $\tilde{E}$ is the *quadratic twist* of E (which is defined and unique up to $\mathbb{F}_q$-isomorphism).[1] When E is ordinary, its quadratic twist has a different cardinality (if $\#E(\mathbb{F}_q) = q+1-t$, then $\#\tilde{E}(\mathbb{F}_q) = q+1+t$), so E and $\tilde{E}$ are in different components of the isogeny graph. But every $\mathbb{F}_q$-isogeny $\phi : E \to E'$ corresponds to an $\mathbb{F}_q$-isogeny $\tilde{\phi} : \tilde{E} \to \tilde{E}'$ of the same degree between the quadratic twists. The component of the $\mathbb{F}_q$-isogeny graph containing an ordinary curve and the component containing its twist are thus isomorphic; we are therefore justified in identifying them, using j-invariants in $\mathbb{F}_q$ for vertices in the $\mathbb{F}_q$-graph.[2] This is not just a mathematical convenience: we will see in §E.3 below that switching between a curve and its twist often allows a useful optimization in isogeny computations.

If an isogeny ϕ is defined over $\mathbb{F}_q$ *and cyclic*, then π acts like a scalar on the points of $\ker\phi$. Thus, for any prime $\ell \neq p$, the number of outgoing ℓ-isogenies from E defined over $\mathbb{F}_q$ can be completely understood by looking at how π acts on $E[\ell]$. Since $E[\ell]$ is a $\mathbb{Z}/\ell\mathbb{Z}$-module of rank 2, the action of π is represented by a 2×2 matrix with entries in $\mathbb{Z}/\ell\mathbb{Z}$ and characteristic polynomial $X^2 - tX + q \mod \ell$. We then have four possibilities:

(0) π has no eigenvalues in $\mathbb{Z}/\ell\mathbb{Z}$, i.e. $X^2 - tX + q$ is irreducible modulo ℓ; then E has no ℓ-isogenies.

(1.1) π has one eigenvalue of (geometric) multiplicity one, i.e. it is conjugate to a non-diagonal matrix $\left(\begin{smallmatrix}\lambda & * \\ 0 & \lambda\end{smallmatrix}\right)$; then there is one ℓ-isogeny from E.

[1] There is a slight technicality here for j-invariants 0 and 1728, where non-quadratic twists may exist. We ignore these special cases because these curves never appear in our cryptosystem: the class groups of their endomorphism rings are trivial, and keyspaces of size 1 are of limited utility in cryptography.

[2] The situation is much more complicated for supersingular graphs, because the curve and its twist are in the same component of the graph; see [DG16, §2] for details.

(1.2) π has one eigenvalue of multiplicity two, i.e. it acts like a scalar matrix $\left(\begin{smallmatrix}\lambda & 0\\ 0 & \lambda\end{smallmatrix}\right)$; then there are $\ell+1$ isogenies of degree ℓ from E.

(2) π has two distinct eigenvalues, i.e. it is conjugate to a diagonal matrix $\left(\begin{smallmatrix}\lambda & 0\\ 0 & \mu\end{smallmatrix}\right)$ with $\lambda \neq \mu$; then there are two ℓ-isogenies from E.

The primes ℓ in Case (2) are called *Elkies primes* for E; these are the primes of most interest to us. Cases (1.x) are only possible if ℓ divides $\Delta_\pi = t^2 - 4q$, the discriminant of the characteristic equation of π; for ordinary curves $\Delta_\pi \neq 0$, so only a finite number of ℓ will fall in these cases, and they will be mostly irrelevant to our cryptosystem. We do not use any ℓ in Case (0).

Since all curves in the same isogeny class over $\mathbb{F}_q$ have the same number of points, they also have the same trace t and discriminant Δ_π. It follows that if ℓ is Elkies for some E in $\mathrm{Ell}_q(\mathcal{O})$, then it is Elkies for every curve in $\mathrm{Ell}_q(\mathcal{O})$.

Hence, if ℓ is an Elkies prime for a curve E, then the connected component of E in the ℓ-isogeny graph is a finite 2-regular graph—that is, a cycle. In the next subsection we describe a group action on this cycle, and determine its size.

Complex multiplication

In this subsection we focus exclusively on ordinary elliptic curves. If E is an ordinary curve with Frobenius π, then $\mathrm{End}(E)$ is isomorphic to an *order*[3] in the quadratic imaginary field $\mathbb{Q}(\sqrt{\Delta_\pi})$ (see [Sil92, p. III.9]). A curve whose endomorphism ring is isomorphic to an order $\mathcal{O}$ is said to have *complex multiplication* by $\mathcal{O}$. For a detailed treatment of the theory of complex multiplication, see [Lan87; Sil94].

The ring of integers $\mathcal{O}_K$ of $K = \mathbb{Q}(\sqrt{\Delta_\pi})$ is its *maximal order*: it contains any other order of K. Hence $\mathbb{Z}[\pi] \subset \mathrm{End}(E) \subset \mathcal{O}_K$, and there is only a finite number of possible choices for $\mathrm{End}(E)$. If we write $\Delta_\pi = d^2\Delta_K$, where Δ_K is the discriminant[4] of $\mathcal{O}_K$, then the index $[\mathcal{O}_K : \mathrm{End}(E)]$ must divide $d = [\mathcal{O}_K : \mathbb{Z}[\pi]]$.

It turns out that isogenies allow us to navigate the various orders. If $\phi : E \to E'$ is an ℓ-isogeny, then one of the following holds [Koh96, Prop. 21]:

- $\mathrm{End}(E) = \mathrm{End}(E')$, and then ϕ is said to be *horizontal*;
- $[\mathrm{End}(E) : \mathrm{End}(E')] = \ell$, and then ϕ is said to be *descending*;
- $[\mathrm{End}(E') : \mathrm{End}(E)] = \ell$, and then ϕ is said to be *ascending*.

Notice that the last two cases can only happen if ℓ divides $d^2 = \Delta_\pi/\Delta_K$, and thus correspond to Cases (1.x) in the previous subsection. If ℓ does not divide Δ_π, then ϕ is necessarily horizontal.

We now present a group action on the set of all curves up to isomorphism having complex multiplication by a fixed order $\mathcal{O}$; the key exchange protocol of §E.3 will be built on this action. Let $\mathfrak{a}$ be an invertible ideal in $\mathrm{End}(E) \simeq \mathcal{O}$ of norm prime to p, and define the $\mathfrak{a}$-*torsion* subgroup of E as

$$E[\mathfrak{a}] = \left\{ P \in E(\overline{\mathbb{F}}_q) \,\middle|\, \sigma(P) = 0 \text{ for all } \sigma \in \mathfrak{a} \right\}.$$

[3] An *order* is a subring which is a $\mathbb{Z}$-module of rank 2.

[4] Δ_K is a *fundamental discriminant*: $\Delta_K \equiv 0, 1 \pmod 4$, and Δ_K or $\frac{\Delta_K}{4}$ is squarefree.

This subgroup is the kernel of a separable isogeny $\phi_{\mathfrak{a}}$. [5] The codomain $E/E[\mathfrak{a}]$ of $\phi_{\mathfrak{a}}$ is well-defined up to isomorphism and will be denoted $\mathfrak{a}\cdot E$. The isogeny $\phi_{\mathfrak{a}}$ is always horizontal—that is, $\mathrm{End}(\mathfrak{a}\cdot E)=\mathrm{End}(E)$—and its degree is the *norm* of $\mathfrak{a}$ as an ideal of $\mathrm{End}(E)$.

Let $\mathrm{Ell}_q(\mathscr{O})$ be the set of isomorphism classes over $\overline{\mathbb{F}}_q$ of curves with complex multiplication by $\mathscr{O}$, and assume it is non-empty. The construction above gives rise to an action of the group of fractional ideals of $\mathscr{O}$ on $\mathrm{Ell}_q(\mathscr{O})$. Furthermore, the principal ideals act trivially (the corresponding isogenies are endomorphisms), so this action induces an action of the *ideal class group* $\mathrm{Cl}(\mathscr{O})$ on $\mathrm{Ell}_q(\mathscr{O})$.

The main theorem of complex multiplication states that this action is *simply transitive*. In other terms, $\mathrm{Ell}_q(\mathscr{O})$ is a PHS under the group $\mathrm{Cl}(\mathscr{O})$: if we fix a curve E as base point, then we have a bijection

$$\begin{aligned}\mathrm{Cl}(\mathscr{O}) &\longrightarrow \mathrm{Ell}_q(\mathscr{O})\\ \text{Ideal class of } \mathfrak{a} &\longmapsto \text{Isomorphism class of } \mathfrak{a}\cdot E.\end{aligned}$$

The order of $\mathrm{Cl}(\mathscr{O})$ is called the *class number* of $\mathscr{O}$, and denoted by $h(\mathscr{O})$. An immediate consequence of the theorem is that $\#\mathrm{Ell}_q(\mathscr{O})=h(\mathscr{O})$.

As before, we work with $\mathbb{F}_q$-isomorphism classes. Then $\mathrm{Ell}_q(\mathscr{O})$ decomposes into two isomorphic PHSes under $\mathrm{Cl}(\mathscr{O})$, each containing the quadratic twists of the curves in the other. We choose one of these two components, that we will also denote $\mathrm{Ell}_q(\mathscr{O})$ in the sequel. (The choice is equivalent to a choice of isomorphism $\mathrm{End}(E)\cong\mathscr{O}$, and thus to a choice of sign on the image of π in $\mathscr{O}$.)

Now let ℓ be an Elkies prime for $E\in\mathrm{Ell}_q(\mathscr{O})$. So far, we have seen that the connected component of E in the ℓ-isogeny graph is a cycle of horizontal isogenies. Complex multiplication tells us more. The restriction of the Frobenius to $E[\ell]$ has two eigenvalues $\lambda\neq\mu$, to which we associate the prime ideals $\mathfrak{a}=(\pi-\lambda,\ell)$ and $\hat{\mathfrak{a}}=(\pi-\mu,\ell)$, both of norm ℓ. We see then that $E[\mathfrak{a}]$ is the eigenspace of λ, defining an isogeny $\phi_{\mathfrak{a}}$ of degree ℓ. Furthermore $\mathfrak{a}\hat{\mathfrak{a}}=\hat{\mathfrak{a}}\mathfrak{a}=(\ell)$, implying that $\mathfrak{a}$ and $\hat{\mathfrak{a}}$ are the inverse of one another in $\mathrm{Cl}(\mathscr{O})$, thus the isogeny $\phi_{\hat{\mathfrak{a}}}:\mathfrak{a}\cdot E\to E$ of kernel $(\mathfrak{a}\cdot E)[\hat{\mathfrak{a}}]$ is the dual of $\phi_{\mathfrak{a}}$ (up to isomorphism).

The eigenvalues λ and μ define opposite directions on the ℓ-isogeny cycle, independent of the starting curve, as shown in Figure E.1. The size of the cycle is the order of $(\pi-\lambda,\ell)$ in $\mathrm{Cl}(\mathscr{O})$, thus partitioning $\mathrm{Ell}_q(\mathscr{O})$ into cycles of equal size.

E.3 Key exchange from isogeny graphs

We would like to instantiate the key exchange protocol of Algorithm 2 with the PHS $X=\mathrm{Ell}_q(\mathscr{O})$ for the group $G=\mathrm{Cl}(\mathscr{O})$, for some well chosen order $\mathscr{O}$ in a quadratic imaginary field. However, given a generic element of $\mathrm{Cl}(\mathscr{O})$, the best algorithm [JS10] to evaluate its action on $\mathrm{Ell}_q(\mathscr{O})$ has subexponential complexity in q, making the protocol infeasible. The solution, following Couveignes [Cou06], is to fix a set S of small prime ideals in $\mathscr{O}$, whose action on X can be computed efficiently, and such that compositions of elements of S cover the whole of G. The action of an arbitrary element of G is then the composition of a series of actions by small elements in S. As Rostovtsev and Stolbunov [RS06] observed, it is useful to visualise this decomposed action as a walk in an isogeny graph.

[5] In fact, one can define $\phi_{\mathfrak{a}}$ for any invertible ideal $\mathfrak{a}$, but it is not always separable.

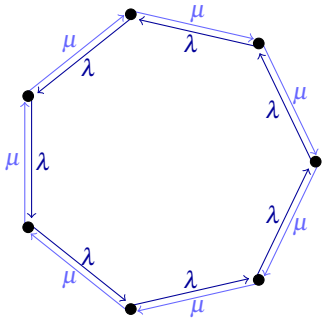


Figure E.1: An isogeny cycle for an Elkies prime ℓ, with edge directions associated with the Frobenius eigenvalues λ and μ.

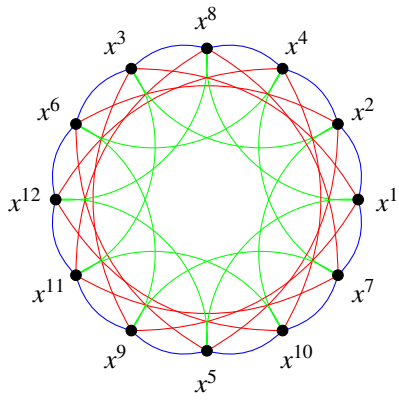


Figure E.2: Undirected Schreier graph on $\langle x\rangle \setminus \{1\}$ where $x^{13} = 1$, acted upon by $(\mathbb{Z}/13\mathbb{Z})^*$, generated by $S = \{2,3,5\}$ (resp. blue, red and green edges).

Walks in isogeny graphs

Let G be a group, X a PHS for G, and S a subset of G. The Schreier graph $\mathscr{G}(G,S,X)$ is the labelled directed graph whose vertex set is X, and where an edge labelled by $s \in S$ links x_1 to x_2 if and only if $s \cdot x_1 = x_2$. It is isomorphic to a Cayley graph for G. If S is symmetric (that is, $S^{-1} = S$), then we associate the same label to s and s^{-1}, making the graph undirected.

A *walk* in $\mathscr{G}(G,S,X)$ is a finite sequence $(s_1,\ldots,s_n)$ of *steps* in S. We define the action of this walk on X as

$$(s_1,\ldots,s_n)\cdot x = \Big(\prod_{i=1}^{n} s_i\Big)\cdot x.$$

In our application G is abelian, so the order of the steps s_i does not matter. We can use this action directly in the key exchange protocol of Algorithm 2, by simply taking private keys to be walks instead of elements in G.

Example E.1. Figure E.2 shows $\mathscr{G}(G,S,X)$ where $G = (\mathbb{Z}/13\mathbb{Z})^*$, $S = \{2,3,5\} \cup \{2^{-1},3^{-1},5^{-1}\}$, and $X = \langle x\rangle \setminus \{1\}$ is a cyclic group of order 13, minus its identity element. The action of G on X is exponentiation: $g \cdot x = x^g$. The action of 11, which takes x^k to x^{11k}, can be expressed using the walks $(2,5,5)$, or $(2^{-1},3^{-1})$, or $(3,5)$, for example. Note that 5 has order 4 modulo 13, thus partitioning $\langle x\rangle \setminus \{1\}$ into 3 cycles of length 4.

Returning to the world of isogenies, we now take

- $X = \mathrm{Ell}_q(\mathscr{O})$ as the vertex set, for some well-chosen q and $\mathscr{O}$; in particular we require $\mathscr{O}$ to be the maximal order (see §E.5).
- $G = \mathrm{Cl}(\mathscr{O})$ as the group acting on X;
- S a set of ideals, whose norms are small Elkies primes in $\mathscr{O}$.

The graph $\mathcal{G}(G,S,X)$ is thus an isogeny graph, composed of many isogeny cycles (one for the norm of each prime in S) superimposed on the vertex set $\mathrm{Ell}_q(\mathcal{O})$. It is connected if S generates $\mathrm{Cl}(\mathcal{O})$. Walks in $\mathcal{G}(G,S,X)$ are called *isogeny walks*.

We compute the action of an ideal $\mathfrak{s}$ (a single *isogeny step*) on an $x \in \mathrm{Ell}_q(\mathcal{O})$ by choosing a representative curve E with $x = j(E)$, and computing an isogeny $\phi_\mathfrak{s} : E \to E'$ from E corresponding to $\mathfrak{s}$; the resulting vertex is $\mathfrak{s} \cdot x = j(E')$. The action of an isogeny walk $(\mathfrak{s}_i)_i$ is then evaluated as the sequence of isogeny steps $\phi_{\mathfrak{s}_i}$. Algorithms for these operations are given in the next subsection.

Using this "smooth" representation of elements in $\mathrm{Cl}(\mathcal{O})$ as isogeny walks lets us avoid computing $\mathrm{Cl}(\mathcal{O})$ and $\mathrm{Ell}_q(\mathcal{O})$, and avoid explicit ideal class arithmetic; only isogenies between elliptic curves are computed. In practice, we re-use the elliptic curve E' from one step as the E in the next; but we emphasize that when isogeny walks are used for Diffie–Hellman, the resulting public keys and shared secrets are not the final elliptic curves, but their j-invariants.

Computing isogeny walks

Since $\mathrm{Cl}(\mathcal{O})$ is commutative, we can break isogeny walks down into a succession of walks corresponding to powers of single primes $\mathfrak{s} = (\ell, \pi - \lambda)$; that is, repeated applications of the isogenies $\phi_\mathfrak{s}$. Depending on $\mathfrak{s}$, we will compute each sequence of $\phi_\mathfrak{s}$ using one of two different methods:

- Algorithm 5 (ElkiesWalk) uses Algorithm 3 (ElkiesFirstStep) followed by a series of calls to Algorithm 4 (ElkiesNextStep), both which use the *modular polynomial* $\Phi_\ell(X,Y)$. This approach works for any $\mathfrak{s}$.

- Algorithm 7 (VéluWalk) uses a series of calls to Algorithm 6 (VéluStep). This approach, which uses torsion points on E, can only be applied when λ satisfies certain properties.

Rostovtsev and Stolbunov only used analogues of Algorithms 3 and 4. The introduction of VéluStep, inspired by SIDH and related protocols (and now a key ingredient in the CSIDH protocol [Cas+18]), speeds up our protocol by a considerable factor; this is the main practical contribution of our work.

Algorithm 3: ElkiesFirstStep

Input: $E \in \mathrm{Ell}_q(\mathcal{O})$; (ℓ, λ) encoding $\mathfrak{s} = (\pi - \lambda, \ell)$
Output: $j(\mathfrak{s} \cdot E)$

$P \leftarrow \Phi_\ell(X, j(E))$
$\{j_1, j_2\} \leftarrow \mathrm{Roots}(P, \mathbb{F}_q)$
$K \leftarrow \mathrm{KernelPolynomial}(\mathrm{Isogeny}(E, j_1, \ell))$ `// e.g. BMSS algorithm [Bos+08]`
$Q \leftarrow$ a nonzero point in K `// e.g.` $(x,y) \in E(\mathbb{F}_q[x,y]/(y^2 - f_E(x), K(x)))$
if $\pi(Q) = [\lambda]Q$ **then**
  return j_1
else
  return j_2

Algorithm 4: ElkiesNextStep

Input: (ℓ,λ) encoding $\mathfrak{s}=(\pi-\lambda,\ell)$; $(j_0,j_1)=(j(E),j(\mathfrak{s}\cdot E))$ for E in $\mathrm{Ell}_q(\mathscr{O})$
Output: $j(\mathfrak{s}\cdot\mathfrak{s}\cdot E)$

$P\leftarrow\Phi_\ell(X,j_1)/(X-j_0)$
$j_2\leftarrow\textsc{Root}(P,\mathbb{F}_q)$ `// It is unique`
return j_2

Algorithm 5: ElkiesWalk

Input: $E\in\mathrm{Ell}_q(\mathscr{O})$; (ℓ,λ) encoding $\mathfrak{s}=(\pi-\lambda,\ell)$; $k\geq 1$
Output: $\mathfrak{s}^k\cdot E$

$j_0\leftarrow j(E)$
$j_1\leftarrow\textsc{ElkiesFirstStep}(E,(\ell,\lambda))$
for $2\leq i\leq k$ **do**
  $(j_0,j_1)\leftarrow(j_1,\textsc{ElkiesNextStep}((\ell,\lambda),(j_0,j_1)))$
$E_R\leftarrow\textsc{EllipticCurveFromJInvariant}(j_1)$
if *not* $\textsc{CheckTrace}(E_R,t)$ **then**
  $E_R\leftarrow\textsc{QuadraticTwist}(E_R)$
return E_R

Algorithm 6: VéluStep

Input: $E\in\mathrm{Ell}_q(\mathscr{O})$; (ℓ,λ) encoding $\mathfrak{s}=(\pi-\lambda,\ell)$; $r>0$; $C_r=\#E(\mathbb{F}_{q^r})$
Output: $\mathfrak{s}\cdot E$

repeat
  $P\leftarrow\textsc{Random}(E(\mathbb{F}_{q^r}))$
  $Q\leftarrow[C_r/\ell]P$
until $Q\neq 0_E$
$K\leftarrow\prod_{i=0}^{(\ell-1)/2}(X-x([i]Q))$ `// Kernel polynomial of isogeny`
$E_R\leftarrow\textsc{IsogenyFromKernel}(E,K)$ `// Apply Vélu's formulæ`
return E_R

Algorithm 7: VéluWalk

Input: $E\in\mathrm{Ell}_q(\mathscr{O})$; (ℓ,λ) encoding $\mathfrak{s}=(\ell,\pi-\lambda)$; $k\geq 1$
Output: $\mathfrak{s}^k\cdot E$

$r\leftarrow\textsc{Order}(\lambda,\ell)$ `// Precompute and store for each` (ℓ,λ)
$C_r\leftarrow\#E(\mathbb{F}_{q^r})$ `// Precompute and store for each` r
for $1\leq i\leq k$ **do**
  $E\leftarrow\textsc{VéluStep}(E,(\ell,\lambda),r,C_r)$
return E

Elkies steps. Algorithms 3 and 4 compute single steps in the ℓ-isogeny graph. Their correctness follows from the definition of the modular polynomial Φ_ℓ: a cyclic ℓ-isogeny exists between two elliptic curves E and E' if and only if $\Phi_\ell(j(E), j(E')) = 0$ (see [Sch95, §6] and [Elk98, §3] for the relevant theory). One may use the classical modular polynomials here, or alternative, lower-degree modular polynomials (Atkin polynomials, for example) with minimal adaptation to the algorithms. In practice, Φ_ℓ is precomputed and stored: several publicly available databases exist (see [Koh18] and [Sut18; BLS12; BOS16], for example).

Given a j-invariant $j(E)$, we can compute its two neighbours in the ℓ-isogeny graph by evaluating $P(X) = \Phi_\ell(j(E), X)$ (a polynomial of degree $\ell + 1$), and then computing its two roots in $\mathbb{F}_q$. Using a Cantor–Zassenhaus-type algorithm, this costs $\tilde{O}(\ell \log q)$ $\mathbb{F}_q$-operations.

We need to make sure we step towards the neighbour in the correct direction. If we have already made one such step, then this is easy: it suffices to avoid backtracking. Algorithm 4 (ElkiesNextStep) does this by removing the factor corresponding to the previous j-invariant in Line 4; this algorithm can be used for all but the first of the steps corresponding to $\mathfrak{s}$.

It remains to choose the right direction in the first step for $\mathfrak{s} = (\ell, \pi - \lambda)$. In Algorithm 3 we choose one of the two candidates for $\phi_\mathfrak{s}$ arbitrarily, and compute its kernel polynomial. This costs $\tilde{O}(\ell)$ $\mathbb{F}_q$-operations using the Bostan–Morain–Salvy–Schost algorithm [Bos+08] with asymptotically fast polynomial arithmetic. We then compute an element Q of $\ker \phi_\mathfrak{s}$ over an extension of $\mathbb{F}_q$ of degree at most $\frac{\ell-1}{2}$, then evaluate $\pi(Q)$ and $[\lambda]Q$. If they match, then we have chosen the right direction; otherwise we take the other root of $P(X)$.

Algorithm 5 (ElkiesWalk) combines these algorithms to compute the iterated action of $\mathfrak{s}$. Line 5 ensures that the curve returned is the the correct component of the ℓ-isogeny graph. Both ElkiesFirstStep and ElkiesNextStep cost $\tilde{O}(\ell \log q)$ $\mathbb{F}_q$-operations, dominated by the calculation of the roots of $P(X)$.

Vélu steps. For some ideals $\mathfrak{s} = (\ell, \pi - \lambda)$, we can completely avoid modular polynomials, and the costly computation of their roots, by constructing $\ker \phi_\mathfrak{s}$ directly from ℓ-torsion points. Let r be the order of λ modulo ℓ; then $\ker \phi_\mathfrak{s} \subseteq E(\mathbb{F}_{q^r})$. If r is not a multiple of the order of the other eigenvalue μ of π on $E[\ell]$, then $E[\ell](\mathbb{F}_{q^r}) = \ker \phi_\mathfrak{s}$. Algorithm 6 (VéluStep) exploits this fact to construct a generator Q of $\ker \phi_\mathfrak{s}$ by computing a point of order ℓ in $E(\mathbb{F}_{q^r})$. The roots of the kernel polynomial of $\phi_\mathfrak{s}$ $x(Q), \ldots, x([(\ell-1)/2]Q)$.[6]

Constructing a point Q of order ℓ in $E(\mathbb{F}_{q^r})$ is straightforward: we take random points and multiply by the cofactor C_r/ℓ, where $C_r := \#E(\mathbb{F}_{q^r})$. Each trial succeeds with probability $1 - 1/\ell$. Note that C_r can be easily (pre)computed from the Frobenius trace t: if we write $C_r = q - t_r + 1$ for $r > 0$ (so $t_1 = t$) and $t_0 = 2$, then the t_r satisfy the recurrence $t_r = t \cdot t_{r-1} - q \cdot t_{r-2}$.

We compute the quotient curve in Line 6 with Vélu's formulæ [Vél71] in $O(\ell)$ $\mathbb{F}_q$-operations. Since $\log C_r \simeq r \log q$, provided $\ell = O(\log q)$, the costly step in Algorithm 6 is the scalar multiplication at Line 3, which costs $\tilde{O}(r^2 \log q)$ $\mathbb{F}_q$-operations.

Comparing the costs. To summarize:

[6] If the order of μ divides r, Algorithm 6 can be extended as follows: take $P \in E[\ell]$, and compute $\pi(P) - [\mu]P$; the result is either zero, or an eigenvector for λ. This is not necessary for any of the primes in our proposed parameters.

- Elkies steps cost $\tilde{O}(\ell \log q)$ $\mathbb{F}_q$-operations;
- Vélu steps cost $\tilde{O}(r^2 \log q)$ $\mathbb{F}_q$-operations, where r is the order of λ in $\mathbb{Z}/\ell\mathbb{Z}$.

In general $r = O(\ell)$, so Elkies steps should be preferred. However, when r is particularly small (and not a multiple of the order of the other eigenvalue), a factor of ℓ can be saved using Vélu steps. The value of r directly depends on λ, which is in turn determined by $\#E(\mathbb{F}_p) \bmod \ell$. Thus, we see that better Step performances depend on the ability to find elliptic curves whose order satisfies congruence conditions modulo small primes. Unfortunately, we can only achieve this partially (see §E.4), so the most efficient solution is to use Vélu steps when we can, and Elkies steps for some other primes.

In practice, Algorithm 6 can be improved by using elliptic curve models with more efficient arithmetic. In our implementation (see §E.6), we used x-only arithmetic on Montgomery models [Mon87; CS17], which also have convenient Vélu formulæ [CH17; Ren18]. Note that we can also avoid computing y-coordinates in Algorithm 3 at Line 5 if $\lambda \neq \pm\mu$: this is the typical case for Elkies steps, and we used this optimization for all Elkies primes in our implementation.

Remark E.2. Note that, in principle, Algorithm 6, can only be used to walk in one direction $\mathfrak{s}_\lambda = (\ell, \pi - \lambda)$, and not in the opposite one $\mathfrak{s}_\mu = (\ell, \pi - \mu)$. Indeed we have assumed that $E[\mathfrak{s}_\lambda]$ is in $E(\mathbb{F}_{q^r})$, while $E[\mathfrak{s}_\mu]$ is not. However, switching to a quadratic twist $\tilde{E}$ of E over $\mathbb{F}_{q^r}$ changes the sign of the Frobenius eigenvalues, thus it may happen that $\tilde{E}[\mathfrak{s}_{-\mu}]$ is in $\tilde{E}(\mathbb{F}_{q^r})$, while $\tilde{E}[\mathfrak{s}_{-\lambda}]$ is not. It is easy to force this behavior by asking that $p \equiv -1 \pmod{\ell}$, indeed then $\lambda = -1/\mu$.

For these eigenvalue pairs we can thus walk in both directions using Vélu steps at no additional cost, following either the direction λ on E, or the direction $-\mu$ on a twist. In Algorithm 6, only the curve order and the random point sampling need to be modified when using quadratic twists.

Sampling isogeny walks for key exchange

We now describe how keys are generated and exchanged in our protocol. Since the cost of the various isogeny walks depends on the ideals chosen, we will use adapted, or *skewed*, smooth representations when sampling elements in $\mathrm{Cl}(\mathcal{O})$ in order to minimize the total computational cost of a key exchange.

We take a (conjectural) generating set for $\mathrm{Cl}(\mathcal{O})$ consisting of ideals over a set S of small Elkies primes, which we partition into three sets according to the step algorithms to be used. We maintain three lists of tuples encoding these primes:

S_{VV} is a list of tuples (ℓ, λ, μ) such that the ideal $(\ell, \pi - \lambda)$ and its inverse $(\ell, \pi - \mu)$ are *both* amenable to VéluStep.

S_{VE} is a list of tuples (ℓ, λ) such that $(\ell, \pi - \lambda)$ is amenable to VéluStep but its inverse $(\ell, \pi - \mu)$ is *not*.

S_{EE} is a list of tuples (ℓ, λ, μ) such that *neither* $(\ell, \pi - \lambda)$ nor $(\ell, \pi - \mu)$ are amenable to VéluStep.

In S_{VV} and S_{EE}, the labelling of eigenvalues as λ and μ is fixed once and for all (that is, the tuples (ℓ, λ, μ) and (ℓ, μ, λ) do not both appear). This fixes directions in each of the ℓ-isogeny cycles. Looking back at Figure E.1, for ℓ associated with S_{EE} and

S_{VV}, both directions in the ℓ-isogeny graph will be available for use in walks; for S_{VE}, only the Vélu direction will be used.

Each secret key in the cryptosystem is a walk in the isogeny graph. Since the class group $\mathrm{Cl}(\mathcal{O})$ is commutative, such a walk is determined by the multiplicities of the primes $\mathfrak{s}$ that appear in it. Algorithm 8 (KeyGen) therefore encodes private-key walks as *exponent vectors*, with one integer exponent for each tuple in S_{VV}, S_{VE}, and S_{EE}. For a tuple (ℓ, λ, μ),

- a positive exponent k_ℓ indicates a walk of k_ℓ ℓ-isogeny steps in direction λ;
- a negative exponent $-k_\ell$ indicates k_ℓ ℓ-isogeny steps in direction μ.

For the tuples (ℓ, λ) in S_{VE}, where we do not use the slower μ-direction, we only allow non-negative exponents. We choose bounds M_ℓ on the absolute value of the exponents k_ℓ so as to minimize the total cost of computing isogeny walks, while maintaining a large keyspace. As a rule, the bounds will be much bigger for the primes in S_{VV} and S_{VE}, where Vélu steps can be applied.

The public keys are j-invariants in $\mathbb{F}_q$, so they can be stored in $\log_2 q$ bits; the private keys are also quite compact, but their precise size depends on the number of primes ℓ and the choice of exponent bounds M_ℓ, which is a problem we will return to in §E.6.

Algorithm 8: KeyGen for cryptosystems in the isogeny graph on $\mathrm{Ell}_q(\mathcal{O})$ with walks based on S, and initial curve E_0. The ideal lists S_{EE}, S_{VV}, and S_{VE}, and the walk bounds M_ℓ, are system parameters.

Input: ()
Output: A secret key $(k_\ell)_{\ell \in S}$ and the corresponding public key $j(E)$

$E \leftarrow E_0$
for $(\ell, \lambda, \mu) \in S_{EE}$ **do**
  $k_\ell \leftarrow \textsc{Random}([-M_\ell, M_\ell])$
  if $k_\ell \geq 0$ **then** $\nu \leftarrow \lambda$
  else $\nu \leftarrow \mu$
  $E \leftarrow \textsc{ElkiesWalk}(E, (\ell, \nu), |k_\ell|)$
for $(\ell, \lambda, \nu) \in S_{VV}$ **do**
  $k_\ell \leftarrow \textsc{Random}([-M_\ell, M_\ell])$
  if $k_\ell \geq 0$ **then** $\nu \leftarrow \lambda$
  else $\nu \leftarrow \mu$
  $E \leftarrow \textsc{VéluWalk}(E, (\ell, \nu), |k_\ell|)$
for $(\ell, \lambda) \in S_{VE}$ **do**
  $k_\ell \leftarrow \textsc{Random}([0, M_\ell])$
  $E \leftarrow \textsc{VéluWalk}(E, (\ell, \lambda), k_\ell)$
return $((k_\ell)_{\ell \in S}, j(E))$

Algorithm 9 completes a Diffie–Hellman key exchange by applying a combination of Elkies and Vélu walks (Algorithms 5 and 7, respectively).

Algorithm 9: DH for the isogeny graph on $\mathrm{Ell}_q(\mathcal{O})$ with primes in S. The ideal lists S_{EE}, S_{VV}, and S_{VE}, and the walk bounds M_ℓ, are system parameters. Public key validation is not included here, but (if desired) should be carried out as detailed in §E.5.

Input: A private key $k_A = (k_{A,\ell})_{\ell \in S}$ corresponding to a walk $(\mathfrak{s}_1, \ldots, \mathfrak{s}_n)$, and a public key $j_B = j(E_B)$ for $E_B \in \mathrm{Ell}_q(O)$
Output: A shared secret $j(\prod_{i=1}^{n} \mathfrak{s}_i \cdot E_B)$

$E \leftarrow \textsc{EllipticCurveFromJInvariant}(j_B)$
if ***not*** $\textsc{CheckTrace}(E,t)$ **then**
  $E \leftarrow \textsc{QuadraticTwist}(E)$
for $(\ell,\lambda,\mu) \in S_{EE}$ **do**
  if $k_{A,\ell} \geq 0$ **then** $\nu \leftarrow \lambda$
  else $\nu \leftarrow \mu$
  $E \leftarrow \textsc{ElkiesWalk}(E,(\ell,\nu),|k_{A,\ell}|)$
for $(\ell,\lambda,\mu) \in S_{VV}$ **do**
  if $k_{A,\ell} \geq 0$ **then** $\nu \leftarrow \lambda$
  else $\nu \leftarrow \mu$
  $E \leftarrow \textsc{VéluWalk}(E,(\ell,\nu),|k_{A,\ell}|)$
for $(\ell,\lambda) \in S_{VE}$ **do**
  $E \leftarrow \textsc{VéluWalk}(E,(\ell,\lambda),k_{A,\ell})$
return $j(E)$

E.4 Public parameter selection

It is evident that the choice of public parameters has a heavy impact on the execution time: smaller Elkies primes, and smaller multiplicative orders of the Frobenius eigenvalues, will lead to better performance. Since all of this information is contained in the value of $\#E(\mathbb{F}_q)$, we now face the problem of constructing ordinary elliptic curves of prescribed order modulo small primes. Unfortunately, and in contrast with the supersingular case, no polynomial-time method to achieve this is known in general: the CM method [AM93; Sut12a], which solves this problem when the corresponding class groups are small, is useless in our setting (see §E.5).

In this section we describe how to use the Schoof–Elkies–Atkin (SEA) point counting algorithm with early abort, combined with the use of certain modular curves, to construct curves whose order satisfies some constraints modulo small primes. This is faster than choosing curves at random and computing their orders completely until a convenient one is found, but it still does not allow us to use the full power of Algorithm VéluStep.

Early-abort SEA. The SEA algorithm [Sch95; Mor95] is the state-of-the-art point-counting algorithm for elliptic curves over large-characteristic finite fields. In order to compute $N = \#E(\mathbb{F}_p)$, it computes N modulo a series of small Elkies primes ℓ, before combining the results via the CRT to get the true value of N.

Cryptographers are usually interested in generating elliptic curves of prime or nearly prime order, and thus without small prime factors. While running SEA on random candidate curves, one immediately detects if $N \equiv 0 \pmod{\ell}$ for the small

primes ℓ; if this happens then the SEA execution is aborted, and restarted with a new curve.

Here, the situation is the opposite: we *want* elliptic curves whose cardinality has many small prime divisors. To fix ideas, we choose the 512-bit prime

$$p := 7 \left(\prod_{2 \leq \ell \leq 380,\ \ell \text{ prime}} \ell \right) - 1 .$$

Then, according to Remark E.2, Algorithm VÉLUSTEP can be used for ℓ-isogenies in both directions for any prime $\ell \leq 380$, as soon as the order of its eigenvalues is small enough. We now proceed as follows:

- Choose a smoothness bound B (we used $B = 13$).
- Pick elliptic curves E at random in $\mathbb{F}_p$, and use the SEA algorithm, aborting when any $\ell \leq B$ with $\#E(\mathbb{F}_p) \not\equiv 0 \pmod{\ell}$ is found.
- For each E which passed the tests above, complete the SEA algorithm to compute $\#E(\mathbb{F}_p)$, and estimate the key exchange running time using this curve as a public parameter (see §E.6).
- The "fastest" curves now give promising candidates for $\#E(\mathbb{F}_p)$.

In considering the efficiency of this procedure, it is important to remark that very few curves will pass the early-abort tests. The bound B should be chosen to balance the overall cost of the first few tests with that of the complete SEA algorithm for the curves which pass them. Therefore, its value is somewhat implementation-dependent.

Finding the maximal order. Once a "good" curve E has been computed, we want to find a curve E_0 having the same number of points, but whose endomorphism ring is maximal, and to ensure that its discriminant is a large integer. Therefore, we attempt to factor the discriminant Δ_π of $\mathbb{Z}[\pi]$: if it is squarefree, then E already has maximal endomorphism ring, and in general the square factors of Δ_π indicate which ascending isogenies have to be computed in order to find E_0.

Remark E.3. Factoring random 512-bit integers is not hard in general, and discriminants of quadratic fields even tend to be slightly smoother than random integers. If a discriminant fails to be completely factored, a conservative strategy would be to discard it, but ultimately undetected large prime-square factors do not present a security issue because computing the possible corresponding large-degree isogenies is intractable (see §E.5).

Using the modular curve $X_1(N)$. Since we are looking for curves with smooth cardinalities, another improvement to this procedure is available: instead of choosing elliptic curves uniformly at random, we pick random candidates using an equation for the modular curve $X_1(N)$ [Sut12b], which guarantees the existence of a rational N-torsion point on the sampled elliptic curve. This idea is used in the procedure of selecting elliptic curves in the Elliptic Curve Method for factoring [ZD06; Zim+18]. In our implementation we used $N = 17$, and also incorporated the existence test in [OKS00] for Montgomery models for the resulting elliptic curves.

Results. We implemented this search using the Sage computer algebra system. Our experiments were conducted on several machines running Intel Xeon E5520 processors at 2.27GHz. After 17,000 hours of CPU time, we found the Montgomery elliptic curve $E : y^2 = x^3 + Ax^2 + x$ over $\mathbb{F}_p$ with p as above, and

$$A = \begin{array}{l} 10861338504649280383859950140772947007703646408372831934324660566888732797778 9 \\ 32142488253565145603672591944602210571423767689240032829444439469242521864171 . \end{array}$$

The trace of Frobenius t of E is

$$-147189550172528104900422131912266898599387555512924231762107728432541952979290 .$$

There is a rational ℓ-torsion point on E, or its quadratic twist, for each ℓ in

$$\{3, 5, 7, 11, 13, 17, 103, 523, 821, 947, 1723\} ;$$

each of these primes is Elkies. Furthermore, $\mathrm{End}(E)$ is the maximal order, and its discriminant is a 511-bit integer that has the following prime factorization:

$$\begin{aligned} &-2^3 \cdot 20507 \cdot 67429 \cdot 11718238170290677 \cdot 12248034502305872059 \\ &\cdot 60884358188204745129468762751254728712569 \\ &\cdot 68495197685926430905162211241300486171895491480444062860794276603493 . \end{aligned}$$

In §E.6, we discuss the practical performance of our key-exchange protocol using these system parameters. Other proposals for parameters are given in [Kie17].

E.5 Security

We now address the security of the CRS primitive, and derived protocols. Intuitively, these systems rely on two assumptions:

1. given two curves E and E' in $\mathrm{Ell}_q(\mathcal{O})$, it is hard to find a (smooth degree) isogeny $\phi : E \to E'$; and
2. the distribution on $\mathrm{Ell}_q(\mathcal{O})$ induced by the random walks sampled in Algorithm 8 is computationally undistinguishable from the uniform distribution.

We start by reviewing the known attacks for the first problem, both in the classical and the quantum setting. Then, we formalize security assumptions and give security proofs against passive adversaries. Finally, we discuss key validation and protection against active adversaries.

Classical attacks

We start by addressing the following, more general, problem:

Problem E.4. Given two ordinary elliptic curves E, E' defined over a finite field $\mathbb{F}_q$, such that $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$, find an isogeny walk $(\phi_i)_{1 \le i \le n}$ such that $\phi_n \circ \cdots \circ \phi_1(E) = E'$.

The curves in Problem E.4 are supposed to be sampled uniformly, though this is never exactly the case in practice. This problem was studied before the emergence of isogeny-based cryptography [Gal99; GHS02; GS13], because of its applications to conventional elliptic-curve cryptography [GHS02; Tes06; JMV09]. The algorithm with the best asymptotic complexity is due to Galbraith, Hess and Smart [GHS02]. It consists of three stages:

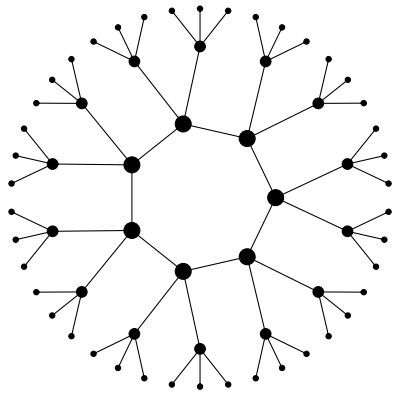


Figure E.3: 3-isogeny graph (*volcano*) containing the curve with $j(E) = 607$ over $\mathbb{F}_{6007}$. A larger vertex denotes a larger endomorphism ring.

Stage 0. Use walks of *ascending* isogenies to reduce to the case where $\mathrm{End}(E) \cong \mathrm{End}(E')$ is the maximal order.

Stage 1. Start two random walks of horizontal isogenies from E and E'; detect the moment when they collide using a Pollard-rho type of algorithm.

Stage 2. Reduce the size of the obtained walk using index-calculus techniques.

To understand Stage 0, recall that all isogenous elliptic curves have the same order, and thus the same trace t of the Frobenius endomorphism π. We know that $\mathrm{End}(E)$ is contained in the ring of integers $\mathcal{O}_K$ of $K = \mathbb{Q}(\sqrt{\Delta_\pi})$, where $\Delta_\pi = t^2 - 4q$ is the Frobenius discriminant. As before we write $\Delta_\pi = d^2\Delta_K$, where Δ_K is the discriminant of $\mathcal{O}_K$; then for any $\ell \mid d$, the ℓ-isogeny graph of E contains *ascending* and *descending* ℓ-isogenies; these graphs are referred to as *volcanoes* [FM02] (see Figure E.3). Ascending isogenies go from curves with smaller endomorphism rings to curves with larger ones, and take us to a curve with $\mathrm{End}(E) \simeq \mathcal{O}_K$ in $O(\log d)$ steps; they can be computed efficiently using the algorithms of [Koh96; FM02; IJ13; De +16]. Assuming[7] all prime factors of d are in $O(\log q)$, we can therefore compute Stage 0 in time polynomial in $\log q$.

The set $\mathrm{Ell}_q(\mathcal{O}_K)$ has the smallest size among all sets $\mathrm{Ell}_q(\mathcal{O})$ for $\mathcal{O} \subset \mathcal{O}_K$, so it is always interesting to reduce to it. This justifies using curves with maximal endomorphism ring in the definition of the protocol in §E.3. When Δ_π is square-free, $\mathbb{Z}[\pi]$ is the maximal order, and the condition is automatically true.

The collision search in Stage 1 relies on the birthday paradox, and has a complexity of $O(\sqrt{h(\mathcal{O}_K)})$. It is known that, on average, $h(\mathcal{O}_K) \approx 0.461\cdots\sqrt{|\Delta_K|}$ (see [Coh93, p. 5.10]), and, assuming the extended Riemann hypothesis, we even have a lower bound (see [Lit28])

$$h(\mathcal{O}_K) \geq 0.147\cdots\frac{(1+o(1))\sqrt{|\Delta_K|}}{\log\log|\Delta_K|}.$$

Since $\Delta_K \sim q$, we expect Stage 1 to take time $O(q^{1/4})$, which justifies a choice of q four times as large as the security parameter. Unfortunately, class numbers are notoriously difficult to compute, the current record being for a discriminant of 300 bits [BJS10]. Computing class numbers for ~ 500-bit discriminants seems to be expensive, albeit

[7] This is typical for isogeny-based protocols. No counter-example has ever been constructed.

feasible; thus, we can only rely on these heuristic arguments to justify the security of our proposed parameters.

The horizontal isogeny produced by Stage 1 is represented by an ideal constructed as a product of exponentially many small ideals. Stage 2 converts this into a sequence of small ideals of length polynomial in $\log q$. Its complexity is bounded by that of Stage 1, so it has no impact on our security estimates.

Remark E.5. The Cohen–Lenstra heuristic [CL84] predicts that the odd part of $\mathrm{Cl}(\mathscr{O}_K)$ is cyclic with overwhelming probability, and other heuristics [HM00] indicate that $h(\mathscr{O}_K)$ is likely to have a large prime factor. However, since there is no known way in which the group structure of $\mathrm{Cl}(\mathscr{O}_K)$ can affect the security of our protocol, we can disregard this matter. No link between the group structure of $E(\mathbb{F}_q)$ itself and the security is known, either.

Quantum attacks

On a quantum computer, an attack with better asymptotic complexity is given by Childs, Jao and Soukharev in [CJS14]. It consists of two algorithms:

1. A (classical) algorithm that takes as input an elliptic curve $E \in \mathrm{Ell}_q(\mathscr{O})$ and an ideal $\mathfrak{a} \in \mathrm{Cl}(\mathscr{O})$, and outputs the curve $\mathfrak{a} \cdot E$;

2. A generic quantum algorithm for the dihedral hidden subgroup problem (dHSP), based upon previous work of Kuperberg [Kup05; Kup13] and Regev [Reg04].

The ideal evaluation algorithm has sub-exponential complexity $L_q(\frac{1}{2}, \frac{\sqrt{3}}{2})$. However, after a subexponential-time classical precomputation, any adversary can know the exact class group structure; in that case, this ideal evaluation step could possibly be performed in polynomial time (and non-negligible success probability) using LLL-based methods, as discussed in [Sto12] and [Cou06, §5].

The dHSP algorithm uses the ideal evaluation algorithm as a (quantum) black box, the number of queries depending on the variant. Childs–Jao–Soukharev gave two versions of this algorithm, Kuperberg's [Kup05] and Regev's[Reg04]. However, both are superseded by Kuperberg's recent work [Kup13]: his new algorithm solves the dHSP in any abelian group of order N using $2^{O(\sqrt{\log N})}$ quantum queries and classical space, but only $O(\log N)$ quantum space. Given this estimate, we expect the bit size of q to grow at worst like the square of the security parameter.

Unfortunately, the analysis of Kuperberg's new algorithm is only asymptotic, and limited to N of a special form; it cannot be directly used to draw conclusions on concrete cryptographic parameters at this stage, especially since the value of the constant hidden by the $O()$ in the exponent is unclear. Thus, it is hard to estimate the impact of this attack at concrete security levels such as those required by NIST [Nat16].

Nevertheless, we remark that the first version of Kuperberg's algorithm, as described in [Reg04, Algorithm 5.1 and Remark 5.2] requires $O(2^{3\sqrt{\log N}} \log N)$ black-box queries and $\sim 2^{3\sqrt{\log N}}$ qubits of memory. Although the quantum memory requirements of this algorithm are rather high, we will take its query complexity as a crude lower bound for the complexity of Kuperberg's newer algorithm in the general case. Of course, this assumption is only heuristic, and should be validated by further study of quantum dHSP solvers; at present time, unfortunately, no precise statement can be made.

Table E.1 thus proposes various parameter sizes, with associated numbers of quantum queries based on the observations above; we also indicate the estimated time

$\log \Delta_K$	$\log h(\mathcal{O}_K)$	classical security	$L_{\lvert\Delta_K\rvert}(1/2,1)$	quantum queries	NIST category
512	256	2^{128}	$2^{56.6}$	$> 2^{56}$	
688	344	2^{172}	$2^{67.0}$	$> 2^{64}$	1
768	384	2^{192}	$2^{71.4}$	$> 2^{67}$	1
1024	512	2^{256}	$2^{84.2}$	$> 2^{76}$	1
1656	828	2^{414}	$2^{110.8}$	$> 2^{96}$	3
3068	1534	2^{767}	$2^{156.9}$	$> 2^{128}$	5

Table E.1: Suggested parameter sizes and associated classical security, class group computation time, and query complexity, using the heuristic estimations of §E.5.

to (classically) precompute the class group structure according to [BJS10].[8] Whenever the quantum query complexity alone is enough to put a parameter in one of NIST's security categories [Nat16], we indicate it in the table. We believe that using query complexity alone is a very conservative choice, and should give more than enough confidence in the post-quantum security properties of our scheme.

The system parameters we proposed in §E.4 correspond to the first line of Table E.1, thus offering at least 56-bit quantum and 128-bit classical security.

Security proofs

We now formalize the assumptions needed to prove the security of the key exchange protocol, and other derived protocols such as PKEs and KEMs, in various models. Given the similarity with the classical Diffie–Hellman protocol on a cyclic group, our assumptions are mostly modeled on those used in that context. Here we are essentially following the lead of Couveignes [Cou06] and Stolbunov [Sto10; Sto12]. However, we take their analyses a step further by explicitly modeling the hardness of distinguishing random walks on Cayley graphs from the uniform distribution: this yields stronger proofs and a better separation of security concerns.

For the rest of this section q is a prime power, $\mathcal{O}$ is an order in a quadratic imaginary field with discriminant $\Delta \sim q$, $\mathrm{Cl}(\mathcal{O})$ is the class group of $\mathcal{O}$, $\mathrm{Ell}_q(\mathcal{O})$ is the (non-empty) set of elliptic curves with complex multiplication by $\mathcal{O}$, and E_0 is a fixed curve in $\mathrm{Ell}_q(\mathcal{O})$. Finally, S is a set of ideals of $\mathcal{O}$ with norm polynomial in $\log q$, and σ is a probability distribution on the set S^* of isogeny walks (i.e. finite sequences of elements in S) used to sample secrets in the key exchange protocol. We write $x \xleftarrow{\sigma} X$ for an element taken from a set X according to σ, and $x \xleftarrow{R} X$ for an element taken according to the uniform distribution.

[8] Computing the class group structure is an instance of the hidden subgroup problem, and thus can be solved in quantum polynomial time by Shor's algorithm.

Our security proofs use four distributions on $\mathrm{Ell}_q(\mathcal{O})^3$:

$$\mathscr{G}_{q,\Delta} := \left\{ (\mathfrak{a}\cdot E_0, \mathfrak{b}\cdot E_0, \mathfrak{a}\mathfrak{b}\cdot E_0) \,\middle|\, \mathfrak{a},\mathfrak{b} \xleftarrow{R} \mathrm{Cl}(\mathcal{O}) \right\},$$

$$\mathscr{W}_{q,\Delta,\sigma} := \left\{ ((\mathfrak{a}_i)_i \cdot E_0, (\mathfrak{b}_j)_j \cdot E_0, (\mathfrak{a}_i)_i \cdot (\mathfrak{b}_j)_j \cdot E_0) \,\middle|\, (\mathfrak{a}_i)_i, (\mathfrak{b}_j)_j \xleftarrow{\sigma} S^* \right\},$$

$$\mathscr{R}_{q,\Delta,\sigma} := \left\{ ((\mathfrak{a}_i)_i \cdot E_0, (\mathfrak{b}_i)_i \cdot E_0, E') \,\middle|\, (\mathfrak{a}_i)_i, (\mathfrak{b}_i)_i \xleftarrow{\sigma} S^*,\ E' \xleftarrow{R} \mathrm{Ell}_q(\mathcal{O}) \right\},$$

$$\mathscr{U}_{q,\Delta} := \left\{ (E_a, E_b, E_{ab}) \,\middle|\, E_a, E_b, E_{ab} \xleftarrow{R} \mathrm{Ell}_q(\mathcal{O}) \right\}.$$

The assumption needed to prove security of the protocols is the hardness of a problem analogous to the classic Decisional Diffie–Hellman (DDH) problem.

Definition E.6 (Isogeny Walk DDH (IW-DDH))**.** Given a triplet of curves (E_a, E_b, E_{ab}) sampled with probability $\frac{1}{2}$ from $\mathscr{R}_{q,\Delta,\sigma}$ and $\frac{1}{2}$ from $\mathscr{W}_{q,\Delta,\sigma}$, decide from which it was sampled.

We split this problem into two finer-grained problems. The first is that of distinguishing between commutative squares sampled uniformly at random and commutative squares sampled from the distribution σ.

Definition E.7 (Isogeny Walk Distinguishing (IWD))**.** Given a triplet of curves (E_a, E_b, E_{ab}) sampled with probability $\frac{1}{2}$ from $\mathscr{W}_{q,\Delta,\sigma}$ and $\frac{1}{2}$ from $\mathscr{G}_{q,\Delta}$, decide from which it was sampled.

The second problem is a group-action analogue of DDH. It also appears in [Cou06] under the name *vectorization*, and in [Sto10; Sto12] under the name DDHAP.

Definition E.8 (Class Group Action DDH (CGA-DDH))**.** Given a triplet of curves (E_a, E_b, E_{ab}) sampled with probability $\frac{1}{2}$ from $\mathscr{G}_{q,\Delta}$ and $\frac{1}{2}$ from $\mathscr{U}_{q,\Delta}$, decide from which it was sampled.

We want to prove the security of protocols based on the primitive of §E.3 under the CGA-DDH and IWD assumptions combined. To do this we give a lemma showing that CGA-DDH and IWD together imply IW-DDH. The technique is straightforward: we use an IW-DDH oracle to solve both the CGA-DDH and IWD problems, showing that at least one of the two must be solvable with non-negligible advantage. The only technical difficulty is that we need an efficient way to simulate the uniform distribution on $\mathrm{Ell}_q(\mathcal{O})$; for this, we use another Cayley graph on $\mathrm{Ell}_q(\mathcal{O})$, with a potentially larger edge set, that is proven in [JMV09] to be an expander under the generalized Riemann hypothesis (GRH).

We let $\mathrm{Adv}^A_{\text{IW-DDH}}$ be the *advantage* of an adversary A against IW-DDH, defined as the probability that A answers correctly, minus $1/2$:

$$2\mathrm{Adv}^A_{\text{IW-DDH}} = \Pr\left[A(\mathscr{R}_{q,\Delta,\sigma}) = 1\right] - \Pr\left[A(\mathscr{W}_{q,\Delta,\sigma}) = 1\right].$$

We define $\mathrm{Adv}^A_{\text{CGA-DDH}}$ and $\mathrm{Adv}^A_{\text{IWD}}$ similarly. Switching answers if needed, we can assume all advantages are positive. We let $\mathrm{Adv}_{\mathrm{X}}(t)$ denote the maximum of $\mathrm{Adv}^A_{\mathrm{X}}$ over all adversaries using at most t resources (running time, queries, etc.).

Lemma E.9. *Assuming GRH, for q large enough and for any bound t on running time, and for any $\varepsilon > 0$,*

$$\mathsf{Adv}_{\text{IW-DDH}}(t) \leq 2\mathsf{Adv}_{\text{IWD}}(t + \text{poly}(\log q, \log \varepsilon)) + \mathsf{Adv}_{\text{CGA-DDH}}(t) + \varepsilon\,.$$

Sketch. We start with an adversary A for IW-DDH, and we construct two simulators S and T for CGA-DDH and IWD respectively.

- The simulator S simply passes its inputs to A, and returns A's response.
- The simulator T receives a triplet (E_a, E_b, E_{ab}) taken from $\mathscr{G}_{q,\Delta}$ or $\mathscr{W}_{q,\Delta,\sigma}$, and flips a coin to decide which of the two following actions it will do:
  - forward (E_a, E_b, E_{ab}) to A, and return the bit given by A; or
  - generate a random curve $E_c \in \text{Ell}_q(\mathscr{O})$, forward (E_a, E_b, E_c) to A, and return the opposite bit to the one given by A.

The curve E_c must be sampled from a distribution close to uniform for the simulator T to work. The only way at our disposal to sample E_c uniformly would be to sample a uniform $\mathfrak{c} \in \text{Cl}(\mathscr{O})$ and take $E_c = \mathfrak{c} \cdot E_0$, but this would be too costly. Instead we use [JMV09, Theorem 1.5], combined with standard results about random walks in expander graphs (for instance, an easy adaptation of the proof of [JMV09, Lemma 2.1]), to sample E_c so that any curve in $\text{Ell}_q(\mathscr{O})$ is taken with probability between $(1-\varepsilon)/h(\mathscr{O})$ and $(1+\varepsilon)/h(\mathscr{O})$, using only $\text{poly}(\log q, \log \varepsilon)$ operations. We can consider this sampling as follows: with probability $1-\varepsilon$, sample E_c uniformly, and with probability ε sample it from an unknown distribution.

Now, if T forwarded (E_a, E_b, E_{ab}) untouched, then we immediately get

$$2\mathsf{Adv}^T_{\text{IWD}} = \mathsf{Adv}^A_{\text{IW-DDH}} - \mathsf{Adv}^S_{\text{CGA-DDH}}\,;$$

if T forwarded (E_a, E_b, E_c), then we get

$$2\mathsf{Adv}^T_{\text{IWD}} \geq \mathsf{Adv}^A_{\text{IW-DDH}} - (1-\varepsilon)\mathsf{Adv}^S_{\text{CGA-DDH}} - \varepsilon\,.$$

Averaging over the two outcomes concludes the proof. □ □

Finally, we define an isogeny-walk analogue of the classic Computational Diffie–Hellman (CDH) problem for groups. Using the same techniques as above, we can prove the security of the relevant protocols based only on CGA-CDH and IWD, without the generalized Riemann hypothesis.

Definition E.10 (Class Group Action CDH (CGA-CDH))**.** Given $E_a = \mathfrak{a} \cdot E_0$ and $E_b = \mathfrak{b} \cdot E_0$ with $\mathfrak{a}, \mathfrak{b} \xleftarrow{R} \text{Cl}(\mathscr{O})$, compute the curve $E_{ab} = \mathfrak{a}\mathfrak{b} \cdot E_0$.

Stolbunov proved the security of HHS Diffie–Hellman under the equivalent of CGA-DDH [Sto10]. Repeating the same steps, we can prove the following theorem.

Theorem E.11. *If the CGA-DDH and IWD assumptions hold, assuming GRH, the key-agreement protocol defined by Algorithms 8 and 9 is session-key secure in the authenticated-links adversarial model of Canetti and Krawczyk [CK01].*

Similarly, we can prove the IND-CPA security of the hashed ElGamal protocol derived from Algorithm 8 by replicating the techniques of e.g. [Gal12, §20.4.11].

Theorem E.12. *Assuming CGA-CDH and IWD, the hashed ElGamal protocol derived from Algorithms 8 and 9 is IND-CPA secure in the random oracle model.*

A heuristic discussion of the IWD assumption. From its very definition, the IWD problem depends on the probability distribution σ we use to sample random walks in the isogeny graph. In this paragraph, we provide heuristic arguments suggesting that the IWD instances generated by Algorithm 9 are hard, provided

1. the keyspace size is at least $\sqrt{|\Delta_K|}$, and
2. S is *not too small*, i.e. the number of isogeny degrees used is in $\Omega(\log q)$.

Proving rapid mixing of isogeny walks with such parameters seems out of reach at present, even under number-theoretic hypotheses such as GRH. The best results available, like [JMV09, Theorem 1.5] (used in the proof of Lemma E.9), typically require isogeny degrees in $\Omega((\log q)^B)$ for some $B > 2$, and fully random walks that are not, for example, skewed towards smaller-degree isogenies.

However, numerical evidence suggests that these theoretical results are too weak. In [JMV09, p. 7.2], it is asked whether an analogue of the previous theorem would be true with the sole constraint $B > 1$. In [GHS02, Section 3], it is mentioned that many fewer split primes are needed to walk in the isogeny graph than theoretically expected. Practical evidence also suggests that the rapid mixing properties are not lost with skewed random walks: such walks are used in [GS13] to accelerate an algorithm solving Problem E.4. We believe that these experiments can bring some evidence in favor of relying on the IWD assumptions with more aggressive parameters than those provided by GRH, although further investigation is required.

Key validation and active security

Modern practice in cryptography mandates the use of stronger security notions than IND-CPA. From the DLP assumption, it is easy to construct protocols with strong security against active adversaries. For example, it is well-known that the hashed ElGamal KEM achieves IND-CCA security in the random oracle model under various assumptions [ABR01; ABR99; CS03].

All of these constructions crucially rely on *key validation*: that is, Alice must verify that the public data sent by Bob defines valid protocol data (e.g., valid elements of a cyclic group), or abort if this is not the case. Failure to perform key validation may result in catastrophic attacks, such as small subgroup [LL97], invalid point [BMM00], and invalid curve attacks [CJ05].

In our context, key validation amounts to verifying that the curve sent by Bob really is an element of $\mathrm{Ell}_q(\mathcal{O}_K)$. Failure to do so exposes Alice to an *invalid graph attack*, where Bob forces Alice onto an isogeny class with much smaller discriminant, or different Elkies primes, and learns something on Alice's secret.

Fortunately, key validation is relatively easy for protocols based on the CRS primitive. All we need to check is that the received j-invariant corresponds to a curve with the right order, and with maximal endomorphism ring.

Verifying the curve order. Since we already know the trace t of the Frobenius endomorphism of all curves in $\mathrm{Ell}_q(\mathcal{O})$, we only need to check that the given E has order $q+1-t$. Assuming that E is cyclic, or contains a cyclic group of order larger than $4\sqrt{q}$, a very efficient randomized algorithm consists in taking a random point P and verifying that it has the expected order. This task is easy if the factorization of $q+1-t$ is known.

Concretely, the curve given in §E.4 has order

$$N = 2^2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13^2 \cdot 17 \cdot 103 \cdot 523 \cdot 821 \cdot 1174286389 \cdot (\text{432-bit prime}),$$

and its group structure is $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/\frac{N}{2}\mathbb{Z}$. To check that a curve is in the same isogeny class, we repeatedly take random points until we find one of order $N/2$.

Verifying the endomorphism ring level. The curve order verification proves that $\mathrm{End}(E)$ is contained between $\mathbb{Z}[\pi]$ and $\mathcal{O}_K$. We have already seen that there is only a finite number of possible rings: their indices in $\mathcal{O}_K$ must divide d where $d^2 = \Delta_\pi/\Delta_K$. Ascending and descending isogenies connect curves with different endomorphism rings, thus we are left with the problem of verifying that E is on the crater of any ℓ-volcano for $\ell \mid d$. Assuming no large prime divides d, this check can be accomplished efficiently by performing random walks in the volcanoes, as described in [Koh96, §4.2] or [FM02]. Note that if we choose Δ_π square-free, then the only possible endomorphism ring is $\mathcal{O}_K$, and there is nothing to be done.

Concretely, for the curve of §E.4 we have $\Delta_\pi/\Delta_K = 2^2$, so there are exactly two possible endomorphism rings. Looking at the action of the Frobenius endomorphism, we see that $\mathrm{End}(E) = \mathcal{O}_K$ if and only if $E[2] \simeq (\mathbb{Z}/2\mathbb{Z})^2$.

Example E.13. Let p and $\mathcal{O}$ be as in §E.4. Suppose we are given the value

$$\alpha = \frac{6774653762400376370473362072511594555277819004969905295950079381173567249377 5}{1873774891388281639871569508662389079106938177131139788464911133375566528902 5}$$

in $\mathbb{F}_p$. It is claimed that α is in $\mathrm{Ell}_p(\mathcal{O})$; that is, it is a valid public key for the system with parameters defined in §E.4. Following the discussion above, to validate α as a public key, it suffices to exhibit a curve with j-invariant α, full rational 2-torsion, and a point of order $N/2$. Using standard formulæ, we find that the two $\mathbb{F}_p$-isomorphism classes of elliptic curves with j-invariant α are represented by the Montgomery curve $E_\alpha/\mathbb{F}_p : y^2 = x(x^2 + Ax + 1)$ with

$$A = \frac{4193809979435365668528368375333535083388979993941154941880421834369488741588 4}{6612599927969489869548583644605423817546131207840311667164101730172820139490 7}$$

and its quadratic twist E'_α. Checking the 2-torsion first, we have $E_\alpha[2](\mathbb{F}_p) \cong E'_\alpha[2](\mathbb{F}_p) \cong (\mathbb{Z}/2\mathbb{Z})^2$, because $A^2 - 4$ is a square in $\mathbb{F}_p$. Trying points on E_α, we find that $(23, \sqrt{23(23^2 + 23A + 1)})$ in $E_\alpha(\mathbb{F}_p)$ has exact order $N/2$. We conclude that $\mathrm{End}(E_\alpha) = \mathcal{O}$, so α is a valid public key. (In fact, E_α is connected to the initial curve by a single 3-isogeny step.)

Consequences for cryptographic constructions. Since both of the checks above can be done much more efficiently than evaluating a single isogeny walk, we conclude that key validation is not only possible, but highly efficient for protocols based on the CRS construction. This stands in stark contrast to the case of SIDH, where key validation is known to be problematic [Gal+16], and even conjectured to be as hard as breaking the system [UJ18].

Thanks to this efficient key validation, we can obtain CCA-secure encryption from the CRS action without resorting to generic transforms such as Fujisaki–Okamoto [FO99], unlike the case of SIKE [SIKE; HHK17]. This in turn enables applications such as non-interactive key exchange, for which no practical post-quantum scheme was known prior to [Cas+18].

r	1	3	4	5	7	8	9
time (s)	0.02	0.10	0.15	0.24	0.8	1.15	1.3

Table E.2: Timings for computing scalar multiplications in $E(\mathbb{F}_{p^r})$, the dominant operation in VÉLUSTEP (Algorithm 6), as a function of the extension degree r.

r	M_ℓ	ℓ	r	M_ℓ	ℓ	r	M_ℓ	ℓ
1*	409	3, 5, 7, 11, 13, 17, 103	4	54	1013, 1181	8	7	881
1	409	523, 821, 947, 1723	5	34	31*, 61*, 1321	9	6	37*, 1693
3	81	19*, 661	7	10	29*, 71*, 547			

Table E.3: Primes ℓ amenable to Algorithm 6 (VÉLUSTEP) for our candidate isogeny graph, with corresponding extension degrees r and proposed walk length bounds M_ℓ.

E.6 Experimental results

In order to demonstrate that our protocol is usable at standard security levels, we implemented it in the Julia programming language. This proof of concept also allowed us to estimate isogeny step costs, which we needed to generate the initial curve in §E.4. We developed several Julia packages[9], built upon the computer algebra package Nemo [Fie+17]. Experiments were conducted using Julia 0.6 and Nemo 0.7.3 on Linux, with an Intel Core i7-5600U cpu at 2.60GHz.

Consider the time to compute one step for an ideal $\mathfrak{s} = (\ell, \pi - \lambda)$. Using Elkies steps, this is approximately the cost of finding the roots of the modular polynomial: roughly $0.017 \cdot \ell$ seconds in our implementation. Using Vélu steps, the cost is approximately that of one scalar multiplication in $E(\mathbb{F}_{q^r})$; timings for the extension degrees r relevant to our parameters appear in Table E.2.

Using this data, finding efficient walk length bounds M_ℓ offering a sufficient keyspace size is easily seen to be an integer optimization problem. We used the following heuristic procedure to find a satisfactory solution. Given a time bound T, let KEYSPACESIZE(T) be the keyspace size obtained when each M_ℓ is the greatest such that the *total time spent on ℓ-isogenies* is less than T. Then, if n is the (classical) security parameter, we look for the *least* T such that KEYSPACESIZE$(T) \geq 2^{2n}$ (according to §E.5), using binary search. While the M_ℓ we obtain are most likely not the best possible, intuitively the outcome is not too far from optimal.

In this way, we obtain a proposal for the walk length bounds M_ℓ to be used in Algorithm 8 along with the curve found in §E.4, to achieve 128-bit classical security. Table E.3 lists the isogeny degrees amenable to Algorithm 6, each with the corresponding extension degree r (a star denotes that the twisted curve allows us to use both directions in the isogeny graph, as in Remark E.2). Table E.4 lists other primes for which we apply Algorithm 5.

Using these parameters, we perform one isogeny walk in approximately 520 seconds. These timings are *worst-case*: the number of isogeny steps is taken to be exactly M_ℓ for each ℓ. This is about as fast as Stolbunov's largest parameter [Sto10], which is for a prime of 428 bits and a keyspace of only 216 bits.

[9] The main code is available at https://github.com/defeo/hhs-keyex/, and the additional dependencies at https://github.com/defeo/EllipticCurves.jl/ and https://github.com/defeo/ClassPolynomials.jl/.

M_ℓ	ℓ	M_ℓ	ℓ	M_ℓ	ℓ
20	23	6	73	2	157, 163, 167, 191, 193, 197, 223, 229
11	41	5	89	1	241, 251, 257, 277, 283, 293, 307
10	43	4	107, 109, 113	1	317, 349, 359
9	47	3	131, 151		

Table E.4: Primes ℓ amenable to Algorithm 5 (ElkiesWalk) for our candidate isogeny graph, with proposed walk length bounds M_ℓ.

We stress that our implementation is *not* optimised. General gains in field arithmetic aside, optimised code could easily beat our proof-of-concept implementation at critical points of our algorithms, such as the root finding steps in Algorithms 3 and 4.

For comparison, without Algorithm 6 the total isogeny walk time would exceed 2000 seconds. Our ideas thus yield an improvement by a factor of over 4 over the original protocol. A longer search for efficient public parameters would bring further improvement.

E.7 Conclusion

We have shown that the Couveignes–Rostovtsev–Stolbunov framework can be improved to become practical at standard pre- and post-quantum security levels; even more so if an optimized C implementation is made. The main obstacle to better performance is the difficulty of generating optimal system parameters: even with a lot of computational power, we cannot expect to produce ordinary curve parameters that allow us to use *only* Vélu steps. In this regard, the CSIDH protocol [Cas+18], which overcomes this problem using supersingular curves instead of ordinary ones, is promising.

One particularly nice feature of our protocol is its highly efficient key validation, which opens a lot of cryptographic doors. However, side-channel-resistant implementations remain an interesting problem for future work.

E.8 References for "Towards practical key exchange from ordinary isogeny graphs"

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. "The Oracle Diffie–Hellman Assumptions and an Analysis of DHIES". In: *Topics in Cryptology — CT-RSA 2001*. Ed. by David Naccache. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 143–158. ISBN: 978-3-540-45353-6.

[ABR99] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. *DHAES: An Encryption Scheme Based on the Diffie–Hellman Problem*. Cryptology ePrint Archive, Report 1999/007. 1999. URL: https://eprint.iacr.org/1999/007.

[AM93] Arthur O. L. Atkin and François Morain. "Elliptic curves and primality proving". In: *Mathematics of Computation* 61.203 (1993), pp. 29–68. ISSN: 0025-5718. DOI: 10.2307/2152935.

[BJS10] Jean-François Biasse, Michael J. Jacobson, and Alan K. Silvester. "Security Estimates for Quadratic Field Based Cryptosystems". In: *Information Security and Privacy*. Ed. by Ron Steinfeld and Philip Hawkes. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 233–247. ISBN: 978-3-642-14081-5.

[BLS12] Reinier Bröker, Kristin Lauter, and Andrew Sutherland. "Modular polynomials via isogeny volcanoes". In: *Mathematics of Computation* 81.278 (2012), pp. 1201–1231. DOI: 10.1090/S0025-5718-2011-02508-1.

[BMM00] Ingrid Biehl, Bernd Meyer, and Volker Müller. "Differential Fault Attacks on Elliptic Curve Cryptosystems". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 131–146. ISBN: 978-3-540-44598-2.

[Bos+08] Alin Bostan, François Morain, Bruno Salvy, and Éric Schost. "Fast algorithms for computing isogenies between elliptic curves". In: *Mathematics of Computation* 77.263 (Sept. 2008), pp. 1755–1778. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-08-02066-8.

[BOS16] Jan Hendrik Bruinier, Ken Ono, and Andrew V. Sutherland. "Class polynomials for nonholomorphic modular functions". In: *Journal of Number Theory* 161 (2016), pp. 204–229. ISSN: 0022-314X. DOI: 10.1016/j.jnt.2015.07.002.

[BW88] Johannes Buchmann and Hugh C. Williams. "A key-exchange system based on imaginary quadratic fields". In: *Journal of Cryptology* 1.2 (June 1988), pp. 107–118. ISSN: 1432-1378. DOI: 10.1007/BF02351719.

[Cas+18] 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*. Ed. by Thomas Peyrin and Steven D. Galbraith. Springer International Publishing, 2018, pp. 395–427. ISBN: 978-3-030-03332-3.

[CH17] Craig Costello and Huseyin Hisil. "A Simple and Compact Algorithm for SIDH with Arbitrary Degree Isogenies". In: *Advances in Cryptology – ASIACRYPT 2017*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Springer International Publishing, 2017, pp. 303–329. ISBN: 978-3-319-70697-9.

[CJ05] Mathieu Ciet and Marc Joye. "Elliptic Curve Cryptosystems in the Presence of Permanent and Transient Faults". In: *Designs, Codes and Cryptography* 36.1 (July 2005), pp. 33–43. ISSN: 1573-7586. DOI: 10.1007/s10623-003-1160-8.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. "Constructing elliptic curve isogenies in quantum subexponential time". In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29.

[CK01] Ran Canetti and Hugo Krawczyk. "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels". In: *EUROCRYPT*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 453–474. ISBN: 3-540-42070-3.

[CL84] Henri Cohen and Hendrik W. Lenstra. "Heuristics on class groups of number fields". In: *Number Theory Noordwijkerhout 1983*. Ed. by Hendrik Jager. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 33–62. ISBN: 978-3-540-38906-4.

[Coh93] Henri Cohen. *A Course in Computational Algebraic Number Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1993. ISBN: 0-387-55640-0.

[Cou06] Jean-Marc Couveignes. *Hard Homogeneous Spaces*. Cryptology ePrint Archive, Report 2006/291. 2006. URL: https://eprint.iacr.org/2006/291.

[CS03] Ronald Cramer and Victor Shoup. "Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack". In: *SIAM Journal on Computing* 33.1 (2003), pp. 167–226. DOI: 10.1137/S0097539702403773.

[CS17] Craig Costello and Benjamin Smith. "Montgomery curves and their arithmetic". In: *Journal of Cryptographic Engineering*. Special issue on Montgomery arithmetic (2017). DOI: 10.1007/s13389-017-0157-6. URL: https://hal.inria.fr/hal-01483768.

[De +16] Luca De Feo, Cyril Hugounenq, Jérôme Plût, and Éric Schost. "Explicit isogenies in quadratic time in any characteristic". In: *LMS Journal of Computation and Mathematics* 19.A (2016), pp. 267–282.

[De 17] Luca De Feo. *Mathematics of Isogeny Based Cryptography*. 2017. arXiv: 1711.04062. URL: http://arxiv.org/abs/1711.04062.

[DG16] Christina Delfs and Steven D. Galbraith. "Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$". In: *Designs, Codes and Cryptography* 78.2 (Feb. 2016), pp. 425–440. ISSN: 1573-7586. DOI: 10.1007/s10623-014-0010-1.

[Elk98] Noam D. Elkies. "Elliptic and modular curves over finite fields and related computational issues". In: *Computational perspectives on number theory (Chicago, IL, 1995)*. Vol. 7. Studies in Advanced Mathematics. Providence, RI: AMS International Press, 1998, pp. 21–76. URL: http://www.ams.org/mathscinet-getitem?mr=1486831.

[Fie+17] Claus Fieker, William Hart, Tommy Hofmann, and Fredrik Johansson. "Nemo/Hecke: Computer Algebra and Number Theory Packages for the Julia Programming Language". In: *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*. ISSAC '17. New York, NY, USA: ACM, 2017, pp. 157–164. ISBN: 978-1-4503-5064-8. DOI: 10.1145/3087604.3087611. URL: http://nemocas.org/.

[FM02] Mireille Fouquet and François Morain. "Isogeny Volcanoes and the SEA Algorithm". In: *Algorithmic Number Theory Symposium*. Ed. by Claus Fieker and David R. Kohel. Vol. 2369. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2002. Chap. 23, pp. 47–62. ISBN: 978-3-540-43863-2. DOI: 10.1007/3-540-45455-1_23.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Advances in Cryptology — CRYPTO' 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 537–554. ISBN: 978-3-540-48405-9.

[Gal+16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. "On the security of supersingular isogeny cryptosystems". In: *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2016, pp. 63–91.

[Gal12] Steven D. Galbraith. *Mathematics of public key cryptography*. https://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html. Cambridge University Press, 2012.

[Gal99] Steven D. Galbraith. "Constructing Isogenies between Elliptic Curves Over Finite Fields". In: *LMS Journal of Computation and Mathematics* 2 (1999), pp. 118–138. DOI: 10.1112/S1461157000000097.

[GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. "Extending the GHS Weil descent attack". In: *Advances in cryptology–EUROCRYPT 2002 (Amsterdam)*. Vol. 2332. Lecture Notes in Computer Science. Berlin: Springer, 2002, pp. 29–44. ISBN: 3-540-43553-0.

[GS13] Steven D. Galbraith and Anton Stolbunov. "Improved algorithm for the isogeny problem for ordinary elliptic curves". In: *Applicable Algebra in Engineering, Communication and Computing* 24.2 (June 2013), pp. 107–131. ISSN: 1432-0622. DOI: 10.1007/s00200-013-0185-0.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation". In: *Theory of Cryptography*. Ed. by Yael Kalai and Leonid Reyzin. Springer International Publishing, 2017, pp. 341–371. ISBN: 978-3-319-70500-2.

[HM00] Safuat Hamdy and Bodo Möller. "Security of Cryptosystems Based on Class Groups of Imaginary Quadratic Orders". In: *Advances in Cryptology — ASIACRYPT 2000*. Ed. by Tatsuaki Okamoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 234–247. ISBN: 978-3-540-44448-0.

[IJ13] Sorina Ionica and Antoine Joux. "Pairing the volcano". In: *Mathematics of Computation* 82.281 (2013), pp. 581–603.

[JD11] David Jao and Luca De Feo. "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies". In: *Post-Quantum Cryptography*. Ed. by Bo-Yin Yang. Vol. 7071. Lecture Notes in Computer Science. Taipei, Taiwan: Springer Berlin / Heidelberg, 2011. Chap. 2, pp. 19–34. ISBN: 978-3-642-25404-8. DOI: 10.1007/978-3-642-25405-5_2.

[JMV09] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. "Expander graphs based on GRH with an application to elliptic curve cryptography". In: *Journal of Number Theory* 129.6 (June 2009), pp. 1491–1504. ISSN: 0022314X. DOI: 10.1016/j.jnt.2008.11.006.

[JS10] David Jao and Vladimir Soukharev. "A Subexponential Algorithm for Evaluating Large Degree Isogenies". In: *ANTS IX: Proceedings of the Algorithmic Number Theory 9th International Symposium*. Vol. 6197. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010. Chap. 19, pp. 219–233. ISBN: 978-3-642-14517-9. DOI: 10.1007/978-3-642-14518-6_19.

[Kie17] Jean Kieffer. "Étude et accélération du protocole d'échange de clés de Couveignes–Rostovtsev–Stolbunov". MA thesis. Inria Saclay & Université Paris VI, 2017.

[Ko+00] Ki Hyoung Ko, Sang Jin Lee, Jung Hee Cheon, Jae Woo Han, Ju-sung Kang, and Choonsik Park. "New Public-Key Cryptosystem Using Braid Groups". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 166–183. ISBN: 978-3-540-44598-2.

[Koh18] David R. Kohel. *Echidna databases*. 2018. URL: http://iml.univ-mrs.fr/~kohel/dbs/.

[Koh96] David R. Kohel. "Endomorphism rings of elliptic curves over finite fields". PhD thesis. University of California at Berkeley, 1996.

[Kup05] Greg Kuperberg. "A subexponential-time quantum algorithm for the dihedral hidden subgroup problem". In: *SIAM Journal of Computing* 35.1 (2005), pp. 170–188. eprint: quant-ph/0302112.

[Kup13] Greg Kuperberg. "Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem". In: *8th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2013)*. Ed. by Simone Severini and Fernando Brandao. Vol. 22. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013, pp. 20–34. ISBN: 978-3-939897-55-2. DOI: 10.4230/LIPIcs.TQC.2013.20. URL: http://drops.dagstuhl.de/opus/volltexte/2013/4321.

[Lan87] Serge Lang. *Elliptic Functions*. Vol. 112. Graduate texts in mathematics. Springer, 1987. ISBN: 9780387965086.

[Lit28] John E Littlewood. "On the Class-Number of the Corpus $P(\sqrt{k})$". In: *Proceedings of the London Mathematical Society* 2.1 (1928), pp. 358–372.

[LL97] Chae Hoon Lim and Pil Joong Lee. "A key recovery attack on discrete log-based schemes using a prime order subgroup". In: *Advances in Cryptology — CRYPTO '97*. Ed. by Burton S. Kaliski. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 249–263. ISBN: 978-3-540-69528-8.

[Mes86] Jean-François Mestre. "La méthode des graphes. Exemples et applications". In: *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata, 1986)*. Nagoya: Nagoya University, 1986. URL: http://boxen.math.washington.edu/msri06/refs/mestre-method-of-graphs/mestre-fr.pdf.

[Mir+06] Josep M. Miret, Ramiro Moreno, Daniel Sadornil, Juan Tena, and Magda Valls. "An algorithm to compute volcanoes of 2-isogenies of elliptic curves over finite fields". In: *Applied Mathematics and Computation* 176.2 (2006), pp. 739–750. DOI: 10.1016/j.amc.2005.10.020.

[MMR07] Gérard Maze, Chris Monico, and Joachim Rosenthal. "Public key cryptography based on semigroup actions". In: *Advances in Mathematics of Communications* 1.4 (2007), pp. 489–507. ISSN: 1930-5346. DOI: 10.3934/amc.2007.1.489.

[Mon87] Peter L. Montgomery. "Speeding the Pollard and Elliptic Curve Methods of Factorization". In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. ISSN: 00255718. DOI: 10.2307/2007888.

[Mor95] François Morain. "Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques". In: *Journal de Théorie des Nombres Bordeaux* 7.1 (1995). Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993), pp. 255–282. ISSN: 1246-7405. URL: http://jtnb.cedram.org/item?id=JTNB_1995__7_1_255_0.

[Nat16] National Institute of Standards and Technology. *Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms*. Ed. by The Federal Register. National Institute of Standards and Technology. 2016. URL: https://www.federalregister.gov/documents/2016/12/20/2016-30615/announcing-request-for-nominations-for-public-key-post-quantum-cryptographic-algorithms.

[OKS00] Katsuyuki Okeya, Hiroyuki Kurumatani, and Kouichi Sakurai. "Elliptic Curves with the Montgomery-Form and Their Cryptographic Applications". In: *Public Key Cryptography — PKC 2000*. Ed. by Hideki Imai and Yuliang Zheng. Vol. 1751. Lecture Notes in Computer Science. Springer, 2000, pp. 238–257. ISBN: 3-540-66967-1. DOI: 10.1007/978-3-540-46588-1_17.

[Reg04] Oded Regev. *A Subexponential Time Algorithm for the Dihedral Hidden Subgroup Problem with Polynomial Space*. arXiv:quant-ph/0406151. June 2004. URL: http://arxiv.org/abs/quant-ph/0406151.

[Ren18] Joost Renes. "Computing Isogenies Between Montgomery Curves Using the Action of (0, 0)". In: *Post-Quantum Cryptography*. Ed. by Tanja Lange and Rainer Steinwandt. Springer International Publishing, 2018, pp. 229–247. ISBN: 978-3-319-79063-3.

[RS06] Alexander Rostovtsev and Anton Stolbunov. *Public-key cryptosystem based on isogenies*. Cryptology ePrint Archive, Report 2006/145. Apr. 2006. URL: http://eprint.iacr.org/2006/145/.

[Sch95] René Schoof. "Counting points on elliptic curves over finite fields". In: *Journal de Théorie des Nombres de Bordeaux* 7.1 (1995), pp. 219–254. URL: http://www.ams.org/mathscinet-getitem?mr=1413578.

[SIKE] Reza Azarderakhsh, Brian Koziel, Matt Campagna, Brian LaMacchia, Craig Costello, Patrick Longa, Luca De Feo, Michael Naehrig, Basil Hess, Joost Renes, Amir Jalali, Vladimir Soukharev, David Jao, and David Urbanik. *Supersingular Isogeny Key Encapsulation*. Nov. 30, 2017. URL: http://sike.org.

[Sil92] Joseph H. Silverman. *The arithmetic of elliptic curves*. Vol. 106. Graduate Texts in Mathematics. Corrected reprint of the 1986 original. New York: Springer-Verlag, 1992, pp. xii+400. ISBN: 0-387-96203-4.

[Sil94] Joseph H. Silverman. *Advanced Topics in the Arithmetic of Elliptic Curves*. Vol. 151. Graduate Texts in Mathematics. Springer, Jan. 1994. ISBN: 0387943285.

[Sto09] Anton Stolbunov. "Reductionist Security Arguments for Public-Key Cryptographic Schemes Based on Group Action". In: *Norsk informasjonssikkerhetskonferanse (NISK)*. Ed. by Stig F. Mjølsnes. 2009.

[Sto10] Anton Stolbunov. "Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves". In: *Advances in Mathematics of Communications* 4.2 (2010).

[Sto12] Anton Stolbunov. "Cryptographic schemes based on isogenies". PhD thesis. Norges teknisk-naturvitenskapelige universitet, Fakultet for informasjonsteknologi, matematikk og elektroteknikk, Institutt for telematikk, Jan. 2012.

[Sut12a] Andrew V. Sutherland. “Accelerating the CM method”. In: *LMS Journal of Computational Mathematics* 15 (2012), pp. 172–204. ISSN: 1461-1570. DOI: 10.1112/S1461157012001015.

[Sut12b] Andrew V. Sutherland. “Constructing elliptic curves over finite fields with prescribed torsion”. In: *Mathematics of Computation* 81 (2012), pp. 1131–1147.

[Sut18] Andrew V. Sutherland. *Modular polynomials*. 2018. URL: https://math.mit.edu/~drew/ClassicalModPolys.html.

[Tes06] Edlyn Teske. “An Elliptic Curve Trapdoor System”. In: *Journal of Cryptology* 19.1 (Jan. 2006), pp. 115–133. ISSN: 0933-2790. DOI: 10.1007/s00145-004-0328-3.

[UJ18] David Urbanik and David Jao. “SoK: The Problem Landscape of SIDH”. In: *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop*. APKC ’18. Incheon, Republic of Korea: ACM, 2018, pp. 53–60. ISBN: 978-1-4503-5756-2. DOI: 10.1145/3197507.3197516.

[Vél71] Jacques Vélu. “Isogénies entre courbes elliptiques”. In: *Comptes Rendus de l’Académie des Sciences de Paris* 273 (1971), pp. 238–241.

[ZD06] Paul Zimmermann and Bruce Dodson. “20 Years of ECM”. In: *Algorithmic Number Theory, 7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings*. Ed. by Florian Hess, Sebastian Pauli, and Michael E. Pohst. Vol. 4076. Lecture Notes in Computer Science. Springer, 2006, pp. 525–542. ISBN: 3-540-36075-1. DOI: 10.1007/11792086_37.

[Zim+18] Paul Zimmermann et al. *GMP-ECM software*. 2018. URL: http://ecm.gforge.inria.fr/.

Bibliography

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. "The Oracle Diffie–Hellman Assumptions and an Analysis of DHIES". In: *Topics in Cryptology — CT-RSA 2001*. Ed. by David Naccache. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 143–158. ISBN: 978-3-540-45353-6.

[ABR99] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. *DHAES: An Encryption Scheme Based on the Diffie–Hellman Problem*. Cryptology ePrint Archive, Report 1999/007. 1999. URL: https://eprint.iacr.org/1999/007.

[Adj+18] Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. *On the cost of computing isogenies between supersingular elliptic curves*. Cryptology ePrint Archive, Report 2018/313. 2018. URL: https://eprint.iacr.org/2018/313.

[AL86] Leonard M. Adleman and Hendrik W. Lenstra. "Finding Irreducible Polynomials over Finite Fields". In: *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*. STOC '86. New York, NY, USA: ACM, 1986, pp. 350–355. ISBN: 0-89791-193-8. DOI: 10.1145/12130.12166.

[All02a] Bill Allombert. "Explicit Computation of Isomorphisms between Finite Fields". In: *Finite Fields and Their Applications* 8.3 (2002), pp. 332–342. DOI: 10.1006/ffta.2001.0344.

[All02b] Bill Allombert. *Explicit Computation of Isomorphisms between Finite Fields*. Revised version. 2002. URL: https://www.math.u-bordeaux.fr/~ballombe/fpisom.ps.

[ALM99] Philippe Aubry, Daniel Lazard, and Marc Moreno Maza. "On the Theories of Triangular Sets". In: *Journal of Symbolic Computation* 28.1 (July 1999), pp. 105–124. ISSN: 0747-7171. DOI: 10.1006/jsco.1999.0269.

[AM93] Arthur O. L. Atkin and François Morain. "Elliptic curves and primality proving". In: *Mathematics of Computation* 61.203 (1993), pp. 29–68. ISSN: 0025-5718. DOI: 10.2307/2152935.

[Ant+06] Adrian Antipa, Daniel Brown, Robert Gallant, Rob Lambert, René Struik, and Scott Vanstone. "Accelerated Verification of ECDSA Signatures". In: *Selected Areas in Cryptography 2005*. Vol. 3897. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2006. Chap. 21, pp. 307–318. ISBN: 978-3-540-33108-7. DOI: 10.1007/11693383_21.

[Atk91] Arthur O. L. Atkin. *The number of points on an elliptic curve modulo a prime*. 1991. URL: http://www.lix.polytechnique.fr/Labo/Francois.Morain/AtkinEmails/19910614.txt.

[Atk92] Arthur O. L. Atkin. *The number of points on an elliptic curve modulo a prime (II)*. 1992. URL: http://www.lix.polytechnique.fr/Labo/Francois.Morain/AtkinEmails/19920319.txt.

[Aza+16] Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. "Key compression for isogeny-based cryptosystems". In: *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*. ACM. 2016, pp. 1–10.

[BC87] Joel V. Brawley and Leonard Carlitz. “Irreducibles and the composed product for polynomials over a finite field”. In: *Discrete Mathematics* 65.2 (1987), pp. 115–139. ISSN: 0012-365X. DOI: 10.1016/0012-365X(87)90135-X.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. “The MAGMA algebra system I: the user language”. In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 235–265. ISSN: 0747-7171. DOI: 10.1006/jsco.1996.0125.

[BCS97a] Wieb Bosma, John Cannon, and Allan Steel. “Lattices of compatibly embedded finite fields”. In: *Journal of Symbolic Computation* 24.3-4 (1997), pp. 351–369. ISSN: 0747-7171. DOI: 10.1006/jsco.1997.0138.

[BCS97b] Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic Complexity Theory*. Springer, Feb. 1997. ISBN: 3540605827.

[Bel08] Juliana V. Belding. “Number Theoretic Algorithms for Elliptic Curves”. PhD thesis. University of Maryland, 2008.

[Ben81] Michael Ben-Or. “Probabilistic algorithms in finite fields”. In: *22nd Annual Symposium on Foundations of Computer Science (SFCS 1981)*. Oct. 1981, pp. 394–398. DOI: 10.1109/SFCS.1981.37.

[Ber+08] Daniel Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. “Twisted Edwards Curves”. In: *Progress in Cryptology — AFRICACRYPT 2008*. 2008, pp. 389–405. DOI: 10.1007/978-3-540-68164-9_26.

[Ber+18] Daniel J. Bernstein, Tanja Lange, Chloe Martindale, and Lorenz Panny. *Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies*. Cryptology ePrint Archive, Report 2018/1059. 2018. URL: https://eprint.iacr.org/2018/1059.

[Ber70] Elwyn R. Berlekamp. “Factoring polynomials over large finite fields”. In: *Mathematics of computation* 24.111 (1970), pp. 713–735. DOI: 10.1090/S0025-5718-1970-0276200-X.

[Ber82] Elwyn R. Berlekamp. “Bit-serial Reed–Solomon encoders”. In: *IEEE Transactions on Information Theory* 28.6 (1982), pp. 869–874.

[BF17] Joppe W. Bos and Simon Friedberger. “Fast Arithmetic Modulo $2^x p^y \pm 1$”. In: *2017 IEEE 24th Symposium on Computer Arithmetic (ARITH)*. July 2017, pp. 148–155. DOI: 10.1109/ARITH.2017.15.

[BIJ18] Jean-François Biasse, Annamaria Iezzi, and Michael J. Jr. Jacobson. “A note on the security of CSIDH”. In: *Kangacrypt 2018*. 2018. URL: https://arxiv.org/abs/1806.03656.

[BJS10] Jean-François Biasse, Michael J. Jacobson, and Alan K. Silvester. “Security Estimates for Quadratic Field Based Cryptosystems”. In: *Information Security and Privacy*. Ed. by Ron Steinfeld and Philip Hawkes. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 233–247. ISBN: 978-3-642-14081-5.

[BJS14] Jean-François Biasse, David Jao, and Anirudh Sankar. “A quantum algorithm for computing isogenies between supersingular elliptic curves”. In: *International Conference in Cryptology in India*. Springer, 2014, pp. 428–442.

[BJW17] Ernest Hunter Brooks, Dimitar Jetchev, and Benjamin Wesolowski. "Isogeny graphs of ordinary abelian varieties". In: *Research in Number Theory* 3.1 (Nov. 2017), p. 28. ISSN: 2363-9555. DOI: 10.1007/s40993-017-0087-5.

[BK78] Richard P. Brent and Hsiang Te Kung. "Fast Algorithms for Manipulating Formal Power Series". In: *Journal of the ACM* 25.4 (1978), pp. 581–595. ISSN: 0004-5411. DOI: 10.1145/322092.322099.

[BL07] Daniel J. Bernstein and Tanja Lange. *Explicit-Formulas Database*. 2007. URL: http://www.hyperelliptic.org/EFD/index.html.

[BL09] Reinier Bröker and Kristin Lauter. "Modular Polynomials for Genus 2". In: *LMS Journal of Computation and Mathematics* 12 (2009), pp. 326–339. DOI: 10.1112/S1461157000001546.

[BLS03] Alin Bostan, Grégoire Lecerf, and Éric Schost. "Tellegen's principle into practice". In: *ISSAC'03*. ACM, 2003, pp. 37–44. ISBN: 1-58113-641-2. DOI: 10.1145/860854.860870.

[BLS12] Reinier Bröker, Kristin Lauter, and Andrew Sutherland. "Modular polynomials via isogeny volcanoes". In: *Mathematics of Computation* 81.278 (2012), pp. 1201–1231. DOI: 10.1090/S0025-5718-2011-02508-1.

[BMM00] Ingrid Biehl, Bernd Meyer, and Volker Müller. "Differential Fault Attacks on Elliptic Curve Cryptosystems". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 131–146. ISBN: 978-3-540-44598-2.

[Bon+18] Dan Boneh, Darren Glass, Daniel Krashen, Kristin Lauter, Shahed Sharif, Alice Silverberg, Mehdi Tibouchi, and Mark Zhandry. "Multiparty Non-Interactive Key Exchange and More from Isogenies on Elliptic Curves". In: *MathCrypt 2018*. 2018. URL: https://eprint.iacr.org/2018/665.

[Bos+05] Alin Bostan, Laureano González-Vega, Hervé Perdry, and Éric Schost. "From Newton sums to coefficients: complexity issues in characteristic p". In: *MEGA'05*. 2005.

[Bos+06] Alin Bostan, Philippe Flajolet, Bruno Salvy, and Éric Schost. "Fast computation of special resultants". In: *Journal of Symbolic Computation* 41.1 (2006), pp. 1–29. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2005.07.001.

[Bos+08] Alin Bostan, François Morain, Bruno Salvy, and Éric Schost. "Fast algorithms for computing isogenies between elliptic curves". In: *Mathematics of Computation* 77.263 (Sept. 2008), pp. 1755–1778. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-08-02066-8.

[Bos10] Alin Bostan. *Algorithmes rapides pour les polynômes, séries formelles et matrices*. Vol. 1. Les cours du CIRM 2. 2010. URL: https://hal.inria.fr/hal-00780433/.

[BOS16] Jan Hendrik Bruinier, Ken Ono, and Andrew V. Sutherland. "Class polynomials for nonholomorphic modular functions". In: *Journal of Number Theory* 161 (2016), pp. 204–229. ISSN: 0022-314X. DOI: 10.1016/j.jnt.2015.07.002.

[Bou07] Nicolas Bourbaki. *Éléments de mathématique*. Algèbre. Chapitre 9. Springer, 2007.

[Bri+17] Ludovic Brieulle, Luca De Feo, Javad Doliskani, Jean-Pierre Flori, and Éric Schost. “Computing isomorphisms and embeddings of finite fields (extended version)”. In: *arXiv preprint arXiv:1705.01221* (2017). URL: https://arxiv.org/abs/1705.01221.

[Bri+18] Ludovic Brieulle, Luca De Feo, Javad Doliskani, Jean-Pierre Flori, and Éric Schost. “Computing isomorphisms and embeddings of finite fields”. In: *Mathematics of Computation* (2018). DOI: 10.1090/mcom/3363.

[Brö09] Reinier Bröker. “Constructing supersingular elliptic curves”. In: *Journal of Combinatorics and Number Theory* 1.3 (2009), pp. 269–273. ISSN: 1942-5600.

[BS11] Gaetan Bisson and Andrew V. Sutherland. “Computing the endomorphism ring of an ordinary elliptic curve over a finite field”. In: *Journal of Number Theory* 131.5 (May 2011), pp. 815–831. ISSN: 0022314X. DOI: 10.1016/j.jnt.2009.11.003.

[BS18] Xavier Bonnetain and André Schrottenloher. *Quantum Security Analysis of CSIDH and Ordinary Isogeny-based Schemes*. Cryptology ePrint Archive, Report 2018/537. 2018. URL: https://eprint.iacr.org/2018/537.

[BSS03] Alin Bostan, Bruno Salvy, and Éric Schost. “Fast Algorithms for Zero-Dimensional Polynomial Systems using Duality”. In: *Applicable Algebra in Engineering, Communication and Computing* 14.4 (Nov. 2003), pp. 239–272. ISSN: 0938-1279. DOI: 10.1007/s00200-003-0133-5.

[BSS99] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart. *Elliptic curves in cryptography*. New York, NY, USA: Cambridge University Press, 1999. ISBN: 0-521-65374-6.

[BW88] Johannes Buchmann and Hugh C. Williams. “A key-exchange system based on imaginary quadratic fields”. In: *Journal of Cryptology* 1.2 (June 1988), pp. 107–118. ISSN: 1432-1378. DOI: 10.1007/BF02351719.

[Can89] David G. Cantor. “On arithmetical algorithms over finite fields”. In: *Journal of Combinatiorial Theory*. Series A 50.2 (1989), pp. 285–300. ISSN: 0097-3165. DOI: 10.1016/0097-3165(89)90020-4.

[Cas+18] 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*. Ed. by Thomas Peyrin and Steven D. Galbraith. Springer International Publishing, 2018, pp. 395–427. ISBN: 978-3-030-03332-3.

[CCR91] Leonard S Charlap, Raymond Coley, and David P Robbins. *Enumeration of rational points on elliptic curves over finite fields*. Preprint. 1991.

[Cer04] Juan M. Cerviño. *On the Correspondence between Supersingular Elliptic Curves and maximal quaternionic Orders*. Apr. 2004. URL: http://arxiv.org/abs/math/0404538.

[CGL09] Denis X. Charles, Eyal Z. Goren, and Kristin E. Lauter. “Cryptographic Hash Functions from Expander Graphs”. In: *Journal of Cryptology* 22.1 (Jan. 2009), pp. 93–113. ISSN: 0933-2790. DOI: 10.1007/s00145-007-9002-x.

[CH13] Wouter Castryck and Hendrik Hubrechts. “The distribution of the number of points modulo an integer on elliptic curves over finite fields”. In: *The Ramanujan Journal* 30.2 (2013), pp. 223–242.

[CH17] Craig Costello and Huseyin Hisil. “A Simple and Compact Algorithm for SIDH with Arbitrary Degree Isogenies”. In: *Advances in Cryptology – ASIACRYPT 2017*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Springer International Publishing, 2017, pp. 303–329. ISBN: 978-3-319-70697-9.

[Chu89] Fan R.K. Chung. “Diameters and eigenvalues”. In: *Journal of the American Mathematical Society* 2.2 (1989), pp. 187–196.

[CJ05] Mathieu Ciet and Marc Joye. “Elliptic Curve Cryptosystems in the Presence of Permanent and Transient Faults”. In: *Designs, Codes and Cryptography* 36.1 (July 2005), pp. 33–43. ISSN: 1573-7586. DOI: 10.1007/s10623-003-1160-8.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. “Constructing elliptic curve isogenies in quantum subexponential time”. In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29.

[CK01] Ran Canetti and Hugo Krawczyk. “Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels”. In: *EUROCRYPT*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 453–474. ISBN: 3-540-42070-3.

[CK91] David G. Cantor and Erich Kaltofen. “On fast multiplication of polynomials over arbitrary algebras”. In: *Acta Informatica* 28.7 (July 1991), pp. 693–701. ISSN: 0001-5903. DOI: 10.1007/BF01178683.

[CKY89] John F. Canny, Eric Kaltofen, and Lakshman N. Yagati. “Solving systems of nonlinear polynomial equations faster”. In: *ISSAC ’89: Proceedings of the ACM-SIGSAM 1989 international symposium on Symbolic and algebraic computation*. New York, NY, USA: ACM, 1989, pp. 121–128. ISBN: 0-89791-325-6. DOI: 10.1145/74540.74556.

[CL08] Jean-Marc Couveignes and Reynald Lercier. “Galois invariant smoothness basis”. In: *Series on Number Theory and Its Applications* 5 (May 2008). World Scientific, pp. 142–167.

[CL13] Jean-Marc Couveignes and Reynald Lercier. “Fast construction of irreducible polynomials over finite fields”. In: *Israel Journal of Mathematics* 194.1 (2013), pp. 77–105.

[CL84] Henri Cohen and Hendrik W. Lenstra. “Heuristics on class groups of number fields”. In: *Number Theory Noordwijkerhout 1983*. Ed. by Hendrik Jager. Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 33–62. ISBN: 978-3-540-38906-4.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. “Efficient Algorithms for Supersingular Isogeny Diffie-Hellman”. In: *Advances in Cryptology – CRYPTO 2016: 36th Annual International Cryptology Conference*. Ed. by Matthew Robshaw and Jonathan Katz. Springer Berlin Heidelberg, 2016, pp. 572–601. ISBN: 978-3-662-53018-4. DOI: 10.1007/978-3-662-53018-4_21.

[CLO05] David A. Cox, John Little, and Donal O’Shea. *Using Algebraic Geometry*. Springer-Verlag, 2005. ISBN: 0387207066.

[CM94] Jean-Marc Couveignes and François Morain. “Schoof’s algorithm and isogeny cycles”. In: *ANTS-I: Proceedings of the First International Symposium on Algorithmic Number Theory*. Vol. 877. Lecture Notes in Computer Science. London, UK: Springer, 1994, pp. 43–58. ISBN: 3-540-58691-1.

[Coh93] Henri Cohen. *A Course in Computational Algebraic Number Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1993. ISBN: 0-387-55640-0.

[Con00] John H. Conway. *On Numbers and Games*. 2nd edition. AK Peters, Ltd., Dec. 2000. ISBN: 1568811276.

[Cos+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. “Efficient Compression of SIDH Public Keys”. In: *Advances in Cryptology – EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Springer International Publishing, 2017, pp. 679–706. ISBN: 978-3-319-56620-7. DOI: 10.1007/978-3-319-56620-7_24.

[Cos+18] Anamaria Costache, Brooke Feigon, Kristin Lauter, Maike Massierer, and Anna Puskas. *Ramanujan graphs in cryptography*. Cryptology ePrint Archive, Report 2018/593. 2018. URL: https://eprint.iacr.org/2018/593.

[Cou00] Jean-Marc Couveignes. “Isomorphisms between Artin-Schreier towers”. In: *Mathematics of Computation* 69.232 (2000), pp. 1625–1631. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-00-01193-5.

[Cou06] Jean-Marc Couveignes. *Hard Homogeneous Spaces*. Cryptology ePrint Archive, Report 2006/291. 2006. URL: https://eprint.iacr.org/2006/291.

[Cou94] Jean-Marc Couveignes. “Quelques calculs en théorie des nombres”. PhD thesis. Université de Bordeaux, 1994.

[Cou96] Jean-Marc Couveignes. “Computing ℓ-isogenies using the p-Torsion”. In: *ANTS-II: Proceedings of the Second International Symposium on Algorithmic Number Theory*. London, UK: Springer-Verlag, 1996, pp. 59–65. ISBN: 3-540-61581-4.

[CR15] Romain Cosset and Damien Robert. “Computing (ℓ, ℓ)-isogenies in polynomial time on Jacobians of genus 2 curves”. In: *Mathematics of Computation* 84.294 (2015), pp. 1953–1975.

[CS03] Ronald Cramer and Victor Shoup. “Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack”. In: *SIAM Journal on Computing* 33.1 (2003), pp. 167–226. DOI: 10.1137/S0097539702403773.

[CS17] Craig Costello and Benjamin Smith. “Montgomery curves and their arithmetic”. In: *Journal of Cryptographic Engineering*. Special issue on Montgomery arithmetic (2017). DOI: 10.1007/s13389-017-0157-6. URL: https://hal.inria.fr/hal-01483768.

[CW90] Don Coppersmith and Shmuel Winograd. “Matrix multiplication via arithmetic progressions”. In: *Journal of Symbolic Computation* 9.3 (1990), pp. 251–280. ISSN: 07477171. DOI: 10.1016/S0747-7171(08)80013-2.

[CZ81] David G Cantor and Hans Zassenhaus. “A New Algorithm for Factoring Polynomials over Finite Fields”. In: *Mathematics of Computation* (1981), pp. 587–592.

[DDS13] Luca De Feo, Javad Doliskani, and Éric Schost. “Fast Algorithms for ℓ-adic Towers over Finite Fields”. In: *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*. ISSAC ’13. New York, NY, USA: ACM, 2013, pp. 165–172. ISBN: 978-1-4503-2059-7. DOI: 10.1145/2465506.2465956.

[DDS14] Luca De Feo, Javad Doliskani, and Éric Schost. “Fast Arithmetic for the Algebraic Closure of Finite Fields”. In: *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*. ISSAC ’14. New York, NY, USA: ACM, 2014, pp. 122–129. ISBN: 978-1-4503-2501-1. DOI: 10.1145/2608628.2608672.

[De +16] Luca De Feo, Cyril Hugounenq, Jérôme Plût, and Éric Schost. “Explicit isogenies in quadratic time in any characteristic”. In: *LMS Journal of Computation and Mathematics* 19.A (2016), pp. 267–282.

[De 10] Luca De Feo. “Algorithmes Rapides pour les Tours de Corps Finis et les Isogénies”. PhD thesis. Ecole Polytechnique X, Dec. 2010. URL: http://tel.archives-ouvertes.fr/tel-00547034/en/.

[De 11] Luca De Feo. “Fast algorithms for computing isogenies between ordinary elliptic curves in small characteristic”. In: *Journal of Number Theory* 131.5 (May 2011), pp. 873–893. ISSN: 0022-314X. DOI: 10.1016/j.jnt.2010.07.003.

[De 17] Luca De Feo. *Mathematics of Isogeny Based Cryptography*. 2017. arXiv: 1711.04062. URL: http://arxiv.org/abs/1711.04062.

[Deu41] Max Deuring. “Die Typen der Multiplikatorenringe elliptischer Funktionenkörper”. In: *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 14.1 (Dec. 1941), pp. 197–272. ISSN: 0025-5858. DOI: 10.1007/BF02940746.

[DG16] Christina Delfs and Steven D. Galbraith. “Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$”. In: *Designs, Codes and Cryptography* 78.2 (Feb. 2016), pp. 425–440. ISSN: 1573-7586. DOI: 10.1007/s10623-014-0010-1.

[DG18] Luca De Feo and Steven D. Galbraith. *SeaSign: Compact isogeny signatures from class group actions*. Cryptology ePrint Archive, Report 2018/824. 2018. URL: https://eprint.iacr.org/2018/824.

[DJP14] Luca De Feo, David Jao, and Jérôme Plût. “Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies”. In: *Journal of Mathematical Cryptology* 8.3 (2014), pp. 209–247.

[DKS18] Luca De Feo, Jean Kieffer, and Benjamin Smith. “Towards Practical Key Exchange from Ordinary Isogeny Graphs”. In: *Advances in Cryptology – ASIACRYPT 2018*. Ed. by Thomas Peyrin and Steven D. Galbraith. Springer International Publishing, 2018, pp. 365–394. ISBN: 978-3-030-03332-3.

[DS09] Luca De Feo and Éric Schost. “Fast arithmetics in Artin-Schreier towers over finite fields”. In: *ISSAC ’09: Proceedings of the 2009 international symposium on Symbolic and algebraic computation*. New York, NY, USA: ACM, 2009, pp. 127–134. ISBN: 978-1-60558-609-0. DOI: 10.1145/1576702.1576722.

[DS12] Luca De Feo and Éric Schost. “Fast arithmetics in Artin-Schreier towers over finite fields”. In: *Journal of Symbolic Computation* 47.7 (2012), pp. 771–792. ISSN: 07477171. DOI: 10.1016/j.jsc.2011.12.008.

[DS14] Javad Doliskani and Éric Schost. “Taking roots over high extensions of finite fields”. In: *Mathematics of Computation* 83.285 (2014), pp. 435–446.

[DS15] Javad Doliskani and Éric Schost. “Computing in degree 2^k-extensions of finite fields of odd characteristic”. In: *Designs, Codes and Cryptography* 74.3 (2015), pp. 559–569.

[DSV03] Giuliana Davidoff, Peter Sarnak, and Alain Valette. *Elementary number theory, group theory, and Ramanujan graphs*. Vol. 55. London Mathematical Society Student Texts. Cambridge: Cambridge University Press, 2003. DOI: 10.1017/CBO9780511615825.

[Dup06] Régis Dupont. “Moyenne arithmético-géométrique, suites de Borchardt et applications”. PhD thesis. École Polytechnique, 2006.

[Eis+18] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. “Supersingular Isogeny Graphs and Endomorphism Rings: Reductions and Solutions”. In: *Advances in Cryptology – EUROCRYPT 2018*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Springer International Publishing, 2018, pp. 329–368. ISBN: 978-3-319-78372-7.

[ElG85] Taher ElGamal. “A public key cryptosystem and a signature scheme based on discrete logarithms”. In: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[Elk92] Noam D. Elkies. “Explicit isogenies”. Manuscript, Boston MA. 1992.

[Elk98] Noam D. Elkies. “Elliptic and modular curves over finite fields and related computational issues”. In: *Computational perspectives on number theory (Chicago, IL, 1995)*. Vol. 7. Studies in Advanced Mathematics. Providence, RI: AMS International Press, 1998, pp. 21–76. URL: http://www.ams.org/mathscinet-getitem?mr=1486831.

[EM03] Andreas Enge and François Morain. “Fast decomposition of polynomials with known Galois group”. In: *AAECC’03: Proceedings of the 15th international conference on Applied algebra, algebraic algorithms and error-correcting codes*. Berlin, Heidelberg: Springer-Verlag, 2003, pp. 254–264. ISBN: 3-540-40111-3.

[Eng09] Andreas Enge. “Computing modular polynomials in quasi-linear time”. In: *Mathematics of Computation* 78.267 (2009), pp. 1809–1824.

[Faz+17] Armanndo Faz-Hernández, Julio López, Eduardo Ochoa-Jiménez, and Francisco Rodríguez-Henríquez. “A Faster Software Implementation of the Supersingular Isogeny Diffie–Hellman Key Exchange Protocol”. In: *IEEE Transactions on Computers* 67 (11 Nov. 2017), pp. 1622–1636. ISSN: 0018-9340. DOI: 10.1109/TC.2017.2771535.

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. "Zero-knowledge proofs of identity". In: *Journal of Cryptology* 1.2 (June 1988), pp. 77–94. ISSN: 0933-2790. DOI: 10.1007/BF02351717.

[FGS99] Sandra Feisel, Joachim von zur Gathen, and M. Amin Shokrollahi. "Normal bases via general Gauss periods". In: *Mathematics of Computation* 68.225 (1999), pp. 271–290.

[Fie+17] Claus Fieker, William Hart, Tommy Hofmann, and Fredrik Johansson. "Nemo/Hecke: Computer Algebra and Number Theory Packages for the Julia Programming Language". In: *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*. ISSAC '17. New York, NY, USA: ACM, 2017, pp. 157–164. ISBN: 978-1-4503-5064-8. DOI: 10.1145/3087604.3087611. URL: http://nemocas.org/.

[FM02] Mireille Fouquet and François Morain. "Isogeny Volcanoes and the SEA Algorithm". In: *Algorithmic Number Theory Symposium*. Ed. by Claus Fieker and David R. Kohel. Vol. 2369. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2002. Chap. 23, pp. 47–62. ISBN: 978-3-540-43863-2. DOI: 10.1007/3-540-45455-1_23.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Advances in Cryptology — CRYPTO' 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 537–554. ISBN: 978-3-540-48405-9.

[Gal+16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. "On the security of supersingular isogeny cryptosystems". In: *Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2016, pp. 63–91.

[Gal12] Steven D. Galbraith. *Mathematics of public key cryptography*. https://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html. Cambridge University Press, 2012.

[Gal99] Steven D. Galbraith. "Constructing Isogenies between Elliptic Curves Over Finite Fields". In: *LMS Journal of Computation and Mathematics* 2 (1999), pp. 118–138. DOI: 10.1112/S1461157000000097.

[GAP] *GAP – Groups, Algorithms, and Programming, Version 4.9.2*. The GAP Group. 2018. URL: https://www.gap-system.org.

[Gau00] Pierrick Gaudry. "Algorithmique des courbes hyperelliptiques et applications à la cryptologie". PhD thesis. École Polytechnique, 2000. URL: https://hal-polytechnique.archives-ouvertes.fr/tel-00514848/.

[Gau86] Carl Friedrich Gauss. *Disquisitiones Arithmeticae*. Ed. by William C. Waterhouse. Springer-Verlag, 1986. ISBN: 9783540962540. URL: https://books.google.fr/books?id=Y-49PgAACAAJ.

[GG99] Joachim von zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. New York, NY, USA: Cambridge University Press, 1999. ISBN: 0-521-64176-4.

[GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. "Extending the GHS Weil descent attack". In: *Advances in cryptology–EUROCRYPT 2002 (Amsterdam)*. Vol. 2332. Lecture Notes in Computer Science. Berlin: Springer, 2002, pp. 29–44. ISBN: 3-540-43553-0.

[Giv] *Givaro – C++ library for arithmetic and algebraic computations*. The LinBox Team. URL: https://github.com/linbox-team/givaro.

[GLS01] Marc Giusti, Grégoire Lecerf, and Bruno Salvy. "A Gröbner free alternative for polynomial system solving". In: *Journal of Complexity* 17.1 (Mar. 2001), pp. 154–211. ISSN: 0885-064X. DOI: 10.1006/jcom.2000.0571.

[GLV01] Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone. "Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms". In: *CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 2001, pp. 190–200. ISBN: 3-540-42456-3. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.2004.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems". In: *Journal of the Association for Computing Machinery* 38.3 (July 1991), pp. 690–728. ISSN: 0004-5411. DOI: 10.1145/116825.116852.

[Gol11] Oded Goldreich. "Basic Facts about Expander Graphs". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. Ed. by Oded Goldreich. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 451–464. ISBN: 978-3-642-22670-0. DOI: 10.1007/978-3-642-22670-0_30.

[GP97] Shuhong Gao and Daniel Panario. "Tests and Constructions of Irreducible Polynomials over Finite Fields". In: *Foundations of Computational Mathematics*. Ed. by Felipe Cucker and Michael Shub. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 346–361. ISBN: 978-3-642-60539-0.

[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*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Springer International Publishing, 2017, pp. 3–33. ISBN: 978-3-319-70694-8.

[GS13] Steven D. Galbraith and Anton Stolbunov. "Improved algorithm for the isogeny problem for ordinary elliptic curves". In: *Applicable Algebra in Engineering, Communication and Computing* 24.2 (June 2013), pp. 107–131. ISSN: 1432-0622. DOI: 10.1007/s00200-013-0185-0.

[GS92] Joachim von zur Gathen and Victor Shoup. "Computing Frobenius Maps and Factoring Polynomials". In: *Computational Complexity* 2 (1992), pp. 187–224.

[GW17] Alexandre Gélin and Benjamin Wesolowski. "Loop-abort faults on supersingular isogeny cryptosystems". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2017, pp. 93–106.

[Har09] David Harvey. "Faster polynomial multiplication via multipoint Kronecker substitution". In: *Journal of Symbolic Computation* 44.10 (Oct. 2009), pp. 1502–1510. ISSN: 07477171. DOI: 10.1016/j.jsc.2009.05.004.

[Har10] William B. Hart. “Fast Library for Number Theory: An Introduction”. In: *Proceedings of the Third International Congress on Mathematical Software*. ICMS’10. Kobe, Japan: Springer-Verlag, 2010, pp. 88–91. URL: http://flintlib.org/.

[Har14] David Harvey. “Counting points on hyperelliptic curves in average polynomial time”. In: *Annals of Mathematics* 179.2 (2014), pp. 783–803. ISSN: 0003486X. URL: http://www.jstor.org/stable/24522768.

[Hea92] David R. Heath-Brown. “Zero-free regions for Dirichlet L-functions, and the least prime in an arithmetic progression”. In: *Proceedings of the London Mathematical Society*. Vol. 64. 2. 1992, pp. 265–338.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. “A Modular Analysis of the Fujisaki-Okamoto Transformation”. In: *Theory of Cryptography*. Ed. by Yael Kalai and Leonid Reyzin. Springer International Publishing, 2017, pp. 341–371. ISBN: 978-3-319-70500-2.

[HL99] Lenwood S. Heath and Nicholas A. Loehr. “New algorithms for generating Conway polynomials over finite fields”. In: *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*. SODA ’99. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1999, pp. 429–437. ISBN: 0-89871-434-6.

[HM00] Safuat Hamdy and Bodo Möller. “Security of Cryptosystems Based on Class Groups of Imaginary Quadratic Orders”. In: *Advances in Cryptology — ASIACRYPT 2000*. Ed. by Tatsuaki Okamoto. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 234–247. ISBN: 978-3-540-44448-0.

[Hoe04] Joris van der Hoeven. “The Truncated Fourier Transform and Applications”. In: *Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*. ISSAC ’04. New York, NY, USA: ACM, 2004, pp. 290–296. ISBN: 1-58113-827-X. DOI: 10.1145/1005285.1005327.

[HQZ04] Guillaume Hanrot, Michel Quercia, and Paul Zimmermann. “The Middle Product Algorithm I”. In: *Applicable Algebra in Engineering, Communication and Computing* 14.6 (2004), pp. 415–438. ISSN: 0938-1279. DOI: 10.1007/s00200-003-0144-2.

[Hug17] Cyril Hugounenq. “Volcanoes and isogeny computing”. PhD thesis. Université Paris-Saclay, Sept. 2017. URL: https://tel.archives-ouvertes.fr/tel-01635463.

[IJ13] Sorina Ionica and Antoine Joux. “Pairing the volcano”. In: *Mathematics of Computation* 82.281 (2013), pp. 581–603.

[IT14] Sorina Ionica and Emmanuel Thomé. “Isogeny graphs with maximal real multiplication”. In: *arXiv preprint arXiv:1407.6672* (2014).

[Jao+18] David Jao, Jason LeGrow, Christopher Leonardi, and Luiz Ruiz-Lopez. “A polynomial quantum space attack on CRS and CSIDH”. In: *MathCrypt 2018*. To appear. 2018.

[JD11] David Jao and Luca De Feo. “Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies”. In: *Post-Quantum Cryptography*. Ed. by Bo-Yin Yang. Vol. 7071. Lecture Notes in Computer Science. Taipei, Taiwan: Springer Berlin / Heidelberg, 2011. Chap. 2, pp. 19–34. ISBN: 978-3-642-25404-8. DOI: 10.1007/978-3-642-25405-5_2.

[JMV09] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. “Expander graphs based on GRH with an application to elliptic curve cryptography”. In: *Journal of Number Theory* 129.6 (June 2009), pp. 1491–1504. ISSN: 0022314X. DOI: 10.1016/j.jnt.2008.11.006.

[Jou02] Antoine Joux. *The Weil and Tate pairings as building blocks for public key cryptosystems*. Berlin, 2002. DOI: 10.1007/3-540-45455-1_3.

[JS10] David Jao and Vladimir Soukharev. “A Subexponential Algorithm for Evaluating Large Degree Isogenies”. In: *ANTS IX: Proceedings of the Algorithmic Number Theory 9th International Symposium*. Vol. 6197. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010. Chap. 19, pp. 219–233. ISBN: 978-3-642-14517-9. DOI: 10.1007/978-3-642-14518-6_19.

[JS14] David Jao and Vladimir Soukharev. “Isogeny-based quantum-resistant undeniable signatures”. In: *Post-Quantum Cryptography: 6th International Workshop, PQCrypto 2014*. Waterloo, ON, Canada: Springer International Publishing, 2014, pp. 160–179. ISBN: 978-3-319-11659-4. DOI: 10.1007/978-3-319-11659-4_10.

[Kal87] Erich Kaltofen. “Computer algebra algorithms”. In: *Annual Review in Computer Science* 2 (1987), pp. 91–118. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/87/Ka87_annrev.pdf.

[Kar+16] Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. “Efficient Finite Field Multiplication for Isogeny Based Post Quantum Cryptography”. In: *Proceedings of WAIFI 2016* (2016).

[Ked01] Kiran S. Kedlaya. “Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology”. In: *Journal of the Ramanujan Mathematical Society* 16.4 (2001), pp. 323–338.

[Ked04] Kiran S. Kedlaya. “Computing zeta functions via p-adic cohomology”. In: *Algorithmic number theory*. Vol. 3076. Lecture Notes in Comput. Sci. Berlin: Springer, 2004, pp. 1–17.

[Kie17] Jean Kieffer. “Étude et accélération du protocole d’échange de clés de Couveignes–Rostovtsev–Stolbunov”. MA thesis. Inria Saclay & Université Paris VI, 2017.

[Kit95] Alexey Yuri Kitaev. “Quantum measurements and the Abelian stabilizer problem”. In: *arXiv preprint quant-ph/9511026* (1995). URL: https://arxiv.org/abs/quant-ph/9511026.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential Power Analysis”. In: *Advances in Cryptology — CRYPTO’ 99*. Vol. 1666. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, Dec. 1999. Chap. 25, pp. 388–397. ISBN: 978-3-540-66347-8. DOI: 10.1007/3-540-48405-1_25.

[Ko+00] Ki Hyoung Ko, Sang Jin Lee, Jung Hee Cheon, Jae Woo Han, Ju-sung Kang, and Choonsik Park. “New Public-Key Cryptosystem Using Braid Groups”. In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 166–183. ISBN: 978-3-540-44598-2.

[Koh+14] David R. Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. "On the quaternion-isogeny path problem". In: *LMS Journal of Computation and Mathematics* 17.A (2014), pp. 418–432.

[Koh18] David R. Kohel. *Echidna databases*. 2018. URL: http://iml.univ-mrs.fr/~kohel/dbs/.

[Koh96] David R. Kohel. "Endomorphism rings of elliptic curves over finite fields". PhD thesis. University of California at Berkeley, 1996.

[KS97] Erich Kaltofen and Victor Shoup. "Fast polynomial factorization over high algebraic extensions of finite fields". In: *ISSAC '97: Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*. New York, NY, USA: ACM, 1997, pp. 184–188. ISBN: 0-89791-875-4. DOI: 10.1145/258726.258777.

[KS98] Erich Kaltofen and Victor Shoup. "Subquadratic-time factoring of polynomials over finite fields". In: *Mathematics of Computation* 67.223 (1998), pp. 1179–1197. ISSN: 0025-5718. DOI: 10.1090/S0025-5718-98-00944-2.

[KU11] Kiran S Kedlaya and Christopher Umans. "Fast polynomial factorization and modular composition". In: *SIAM Journal on Computing* 40.6 (2011), pp. 1767–1802.

[Kum46] Ernst Eduard Kummer. "Über die Divisoren gewisser Formen der Zahlen, welche aus der Theorie der Kreistheilung entstehen". German. In: *Journal für die reine und angewandte Mathematik* 30 (1846), pp. 107–116. URL: http://eudml.org/doc/147278.

[Kum47a] Ernst Eduard Kummer. "Sur les nombres complexes qui sont formés avec les nombres entiers réels et les racines de l'unité". French. In: *Journal de Mathématiques Pures et Appliquées* (1847), pp. 185–212. URL: http://eudml.org/doc/235075.

[Kum47b] Ernst Eduard Kummer. "Über die Zerlegung der aus Wurzeln der Einheit gebildeten complexen Zahlen in ihre Primfactoren". German. In: *Journal für die reine und angewandte Mathematik* 35 (1847), pp. 327–367. URL: http://eudml.org/doc/147394.

[Kum47c] Ernst Eduard Kummer. "Zur Theorie der complexen Zahlen". German. In: *Journal für die reine und angewandte Mathematik* 35 (1847), pp. 319–326. URL: http://eudml.org/doc/147393.

[Kum51] Ernst Eduard Kummer. "Mémoire sur la théorie des nombres complexes composés de racines de l'unité et de nombres entiers". French. In: *Journal de Mathématiques Pures et Appliquées* (1851), pp. 377–498. URL: http://eudml.org/doc/235621.

[Kum55] Ernst Eduard Kummer. "Über eine besondere Art, aus complexen Einheiten gebildeter Ausdrücke". German. In: *Journal für die reine und angewandte Mathematik* 50 (1855), pp. 212–232. URL: http://eudml.org/doc/147605.

[Kum57] Ernst Eduard Kummer. "Über die den Gaußschen Perioden der Kreistheilung entsprechenden Congruenzwurzeln". German. In: *Journal für die reine und angewandte Mathematik* 53 (1857), pp. 142–148. URL: http://eudml.org/doc/147659.

[Kun86] Ernst Kunz. *Kähler differentials*. Friedrich Vieweg & Sohn, 1986.

[Kup05] Greg Kuperberg. "A subexponential-time quantum algorithm for the dihedral hidden subgroup problem". In: *SIAM Journal of Computing* 35.1 (2005), pp. 170–188. eprint: quant-ph/0302112.

[Kup13] Greg Kuperberg. "Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem". In: *8th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2013)*. Ed. by Simone Severini and Fernando Brandao. Vol. 22. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013, pp. 20–34. ISBN: 978-3-939897-55-2. DOI: 10.4230/LIPIcs.TQC.2013.20. URL: http://drops.dagstuhl.de/opus/volltexte/2013/4321.

[Lan02] Serge Lang. *Algebra*. 3rd edition. Springer, Jan. 2002. ISBN: 038795385X.

[Lan87] Serge Lang. *Elliptic Functions*. Vol. 112. Graduate texts in mathematics. Springer, 1987. ISBN: 9780387965086.

[Lau04] Alan G. B. Lauder. "Computing zeta functions of Artin-Schreier curves over finite fields II". In: *Journal of Complexity* 20.2-3 (June 2004), pp. 331–349. ISSN: 0885064X. DOI: 10.1016/j.jco.2003.08.009.

[Le 14] François Le Gall. "Powers of Tensors and Fast Matrix Multiplication". In: *ISSAC'14*. ACM, 2014, pp. 296–303.

[Leb15] Romain Lebreton. "Relaxed Hensel lifting of triangular sets". In: *Journal of Symbolic Computation* 68 (2015). Effective Methods in Algebraic Geometry, pp. 230–258. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2014.09.012.

[Len77] Hendrik W. Lenstra. "On the algebraic closure of two". In: *Indagationes Mathematicae (Proceedings)* 80.5 (1977), pp. 389–396. ISSN: 1385-7258. DOI: 10.1016/1385-7258(77)90053-1.

[Len87] Hendrik W. Lenstra. "Factoring integers with elliptic curves". In: *Annals of Mathematics* 126 (1987), pp. 649–673.

[Len91] Hendrik W. Lenstra. "Finding isomorphisms between finite fields". In: *Mathematics of Computation* 56.193 (1991), pp. 329–347.

[Ler96] Reynald Lercier. "Computing Isogenies in GF(2,n)". In: *ANTS-II: Proceedings of the Second International Symposium on Algorithmic Number Theory*. London, UK: Springer-Verlag, 1996, pp. 197–212. ISBN: 3-540-61581-4.

[Lit28] John E Littlewood. "On the Class-Number of the Corpus $P(\sqrt{k})$". In: *Proceedings of the London Mathematical Society* 2.1 (1928), pp. 358–372.

[LL97] Chae Hoon Lim and Pil Joong Lee. "A key recovery attack on discrete log-based schemes using a prime order subgroup". In: *Advances in Cryptology — CRYPTO '97*. Ed. by Burton S. Kaliski. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 249–263. ISBN: 978-3-540-69528-8.

[LMS07] Xin Li, Marc Moreno Maza, and Éric Schost. "Fast Arithmetic for Triangular Sets: From Theory to Practice". In: *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*. ISSAC '07. New York, NY, USA: ACM, 2007, pp. 269–276. ISBN: 978-1-59593-743-8. DOI: 10.1145/1277548.1277585.

[LMS13] Romain Lebreton, Esmaeil Mehrabi, and Éric Schost. "On the Complexity of Solving Bivariate Systems: The Case of Non-singular Solutions". In: *ISSAC'13*. ACM, 2013, pp. 251–258.

[LO77] Jeffrey C. Lagarias and Andrew M. Odlyzko. "Effective versions of the Chebotarev density theorem". In: *Algebraic number fields: L-functions and Galois properties*. London: Academic Press, 1977, pp. 409–464.

[LPS88] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. "Ramanujan graphs". In: *Combinatorica* 8.3 (1988). DOI: 10.1007/BF02126799.

[LR12] David Lubicz and Damien Robert. "Computing isogenies between abelian varieties". In: *Compositio Mathematica* 148.5 (2012), pp. 1483–1515. DOI: 10.1112/S0010437X12000243.

[LR15] David Lubicz and Damien Robert. "Computing separable isogenies in quasi-optimal time". In: *LMS Journal of Computation and Mathematics* 18.1 (2015), pp. 198–216. DOI: 10.1112/S146115701400045X.

[LS08a] Hendrik W. Lenstra and Bart de Smit. *Standard models for finite fields: the definition*. 2008. URL: http://www.math.leidenuniv.nl/~desmit/papers/standard_models.pdf.

[LS08b] Reynald Lercier and Thomas Sirvent. "On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field". In: *Journal de théorie des nombres de Bordeaux* 20.3 (2008), pp. 783–797. URL: http://perso.univ-rennes1.fr/reynald.lercier/file/LS08.pdf.

[LS14] Patrick Longa and Francesco Sica. "Four-Dimensional Gallant–Lambert–Vanstone Scalar Multiplication". In: *Journal of Cryptology* 27.2 (2014), pp. 248–283. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9144-3.

[Lüb08] Frank Lübeck. *Conway polynomials for finite fields*. 2008. URL: http://www.math.rwth-aachen.de/~Frank.Luebeck/data/ConwayPol/.

[Lub94] Alexander Lubotzky. *Discrete groups, expanding graphs and invariant measures*. Vol. 125. Progress in Mathematics. Basel: Birkhäuser Verlag, 1994. ISBN: 978-3-0346-0332-4. DOI: 10.1007/978-3-0346-0332-4.

[LV16] Pierre Lairez and Tristan Vaccon. "On p-adic Differential Equations with Separation of Variables". In: *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*. ISSAC '16. New York, NY, USA: ACM, 2016, pp. 319–323. ISBN: 978-1-4503-4380-0. DOI: 10.1145/2930889.2930912.

[Mes86] Jean-François Mestre. "La méthode des graphes. Exemples et applications". In: *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata, 1986)*. Nagoya: Nagoya University, 1986. URL: http://boxen.math.washington.edu/msri06/refs/mestre-method-of-graphs/mestre-fr.pdf.

[Mil15] Enea Milio. "A quasi-linear time algorithm for computing modular polynomials in dimension 2". In: *LMS Journal of Computation and Mathematics* 18.1 (2015), pp. 603–632. DOI: 10.1112/S1461157015000170.

[Mir+05] Josep M. Miret, Ramiro Moreno, Ana Rio, and Magda Valls. "Determining the 2-Sylow subgroup of an elliptic curve over a finite field". In: *Mathematics of Computation* 74.249 (2005), pp. 411–427. DOI: 10.1090/S0025-5718-04-01640-0.

[Mir+06] Josep M. Miret, Ramiro Moreno, Daniel Sadornil, Juan Tena, and Magda Valls. “An algorithm to compute volcanoes of 2-isogenies of elliptic curves over finite fields”. In: *Applied Mathematics and Computation* 176.2 (2006), pp. 739–750. DOI: 10.1016/j.amc.2005.10.020.

[Mir+08] Josep M. Miret, Ramiro Moreno, Daniel Sadornil, Juan Tena, and Magda Valls. “Computing the height of volcanoes of ℓ-isogenies of elliptic curves over finite fields”. In: *Applied Mathematics and Computation* 196.1 (2008), pp. 67–76. ISSN: 0096-3003. DOI: 10.1016/j.amc.2007.05.037.

[MMR07] Gérard Maze, Chris Monico, and Joachim Rosenthal. “Public key cryptography based on semigroup actions”. In: *Advances in Mathematics of Communications* 1.4 (2007), pp. 489–507. ISSN: 1930-5346. DOI: 10.3934/amc.2007.1.489.

[MMT01] Markus Maurer, Alfred Menezes, and Edlyn Teske. “Analysis of the GHS Weil Descent Attack on the ECDLP over Characteristic Two Finite Fields of Composite Degree”. In: *INDOCRYPT ’01: Proceedings of the Second International Conference on Cryptology in India*. Berlin: Springer-Verlag, 2001, pp. 195–213. ISBN: 3-540-43010-5.

[Moe76] Robert T. Moenck. “Another polynomial homomorphism”. In: *Acta Informatica* 6.2 (June 1976), pp. 153–169. ISSN: 0001-5903. DOI: 10.1007/BF00268498.

[Mon87] Peter L. Montgomery. “Speeding the Pollard and Elliptic Curve Methods of Factorization”. In: *Mathematics of Computation* 48.177 (1987), pp. 243–264. ISSN: 00255718. DOI: 10.2307/2007888.

[Moo12] Dustin Moody. “Computing isogeny volcanoes of composite degree”. In: *Applied Mathematics and Computation* 218.9 (2012), pp. 5249–5258. ISSN: 0096-3003. DOI: 10.1016/j.amc.2011.11.008.

[Mor95] François Morain. “Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques”. In: *Journal de Théorie des Nombres Bordeaux* 7.1 (1995). Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993), pp. 255–282. ISSN: 1246-7405. URL: http://jtnb.cedram.org/item?id=JTNB_1995__7_1_255_0.

[MP13] Gary L. Mullen and Daniel Panario. *Handbook of finite fields*. CRC Press, 2013.

[MR17] Enea Milio and Damien Robert. “Modular polynomials on Hilbert surfaces”. Working paper or preprint. Sept. 2017. URL: https://hal.archives-ouvertes.fr/hal-01520262.

[MRC17] Michael Meyer, Steffen Reith, and Fabio Campos. *On hybrid SIDH schemes using Edwards and Montgomery curve arithmetic*. Cryptology ePrint Archive, Report 2017/1213. 2017. URL: https://eprint.iacr.org/2017/1213.

[MV10] Preda Mihailescu and Victor Vuletescu. “Elliptic Gauss sums and applications to point counting”. In: *Journal of Symbolic Computation* 45.8 (2010), pp. 825–836. ISSN: 0747-7171. DOI: 10.1016/j.jsc.2010.01.004.

[Nar18] Anand Kumar Narayanan. “Fast Computation of Isomorphisms Between Finite Fields Using Elliptic Curves”. In: *International Workshop on the Arithmetic of Finite Fields, WAIFI 2018*. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2018.

[Nat16] National Institute of Standards and Technology. *Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms*. Ed. by The Federal Register. National Institute of Standards and Technology. 2016. URL: https://www.federalregister.gov/documents/2016/12/20/2016-30615/announcing-request-for-nominations-for-public-key-post-quantum-cryptographic-algorithms.

[Nic88] Werner Nickel. *Endliche Körper in dem gruppentheoretischen Programmsystem GAP*. 1988. URL: https://www2.mathematik.tu-darmstadt.de/~nickel/.

[Noe32] Emmy Noether. "Normalbasis bei Körpern ohne höhere Verzweigung". In: *Journal für die reine und angewandte Mathematik* 167 (1932), pp. 147–152. URL: http://eudml.org/doc/149800.

[NTL] Victor Shoup. *NTL: A library for doing number theory*. URL: http://www.shoup.net/ntl.

[OEI12] OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences*. http://oeis.org/A130715. 2012.

[OKS00] Katsuyuki Okeya, Hiroyuki Kurumatani, and Kouichi Sakurai. "Elliptic Curves with the Montgomery-Form and Their Cryptographic Applications". In: *Public Key Cryptography — PKC 2000*. Ed. by Hideki Imai and Yuliang Zheng. Vol. 1751. Lecture Notes in Computer Science. Springer, 2000, pp. 238–257. ISBN: 3-540-66967-1. DOI: 10.1007/978-3-540-46588-1_17.

[PARI] *PARI/GP, version 2.8.0*. The PARI Group. Bordeaux, 2016. URL: https://pari.math.u-bordeaux.fr/.

[Pet17] Christophe Petit. "Faster Algorithms for Isogeny Problems Using Torsion Point Images". In: *Advances in Cryptology – ASIACRYPT 2017*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Springer International Publishing, 2017, pp. 330–353. ISBN: 978-3-319-70697-9.

[Pil90] Jonathan Pila. "Frobenius Maps of Abelian Varieties and Finding Roots of Unity in Finite Fields". In: *Mathematics of Computation* 55.192 (1990), pp. 745–763. ISSN: 00255718. DOI: 10.2307/2008445.

[Pin92] Richard G. E. Pinch. "Recognising Elements Of Finite Fields". In: *Cryptography and Coding II*. Oxford University Press, 1992, pp. 193–197.

[Piz90] Arnold K. Pizer. "Ramanujan graphs and Hecke operators". In: *Bulletin of the American Mathematical Society (N.S.)* 23.1 (1990). DOI: 10.1090/S0273-0979-1990-15918-X.

[Piz98] Arnold K. Pizer. "Ramanujan graphs". In: *Computational perspectives on number theory (Chicago, IL, 1995)*. Vol. 7. AMS/IP Stud. Adv. Math. Providence, RI: Amer. Math. Soc., 1998.

[PLQ08] Christophe Petit, Kristin Lauter, and Jean-Jacques Quisquater. "Full Cryptanalysis of LPS and Morgenstern Hash Functions". In: *Proceedings of the 6th international conference on Security and Cryptography for Networks*. SCN '08. Berlin, Heidelberg: Springer-Verlag, 2008. DOI: 10.1007/978-3-540-85855-3_18.

[Poi95] David Pointcheval. “A New Identification Scheme Based on the Perceptrons Problem”. In: *Advances in Cryptology — EUROCRYPT ’95*. Vol. 921. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1995. Chap. 26, pp. 319–328. ISBN: 978-3-540-59409-3. DOI: 10.1007/3-540-49264-X_26. URL: http://dx.doi.org/10.1007/3-540-49264-X%5C_26.

[PS06] Cyril Pascal and Éric Schost. “Change of order for bivariate triangular sets”. In: *ISSAC ’06: Proceedings of the 2006 international symposium on Symbolic and algebraic computation*. New York, NY, USA: ACM, 2006, pp. 277–284. ISBN: 1-59593-276-3. DOI: 10.1145/1145768.1145814.

[PS13a] Adrian Poteaux and Éric Schost. “Modular Composition Modulo Triangular Sets and Applications”. In: *Computational Complexity* 22.3 (2013), pp. 463–516.

[PS13b] Adrien Poteaux and Éric Schost. “On the complexity of computing with zero-dimensional triangular sets”. In: *Journal of Symbolic Computation* 50 (2013), pp. 110–138.

[PS73] Michael S. Paterson and Larry J. Stockmeyer. “On the Number of Nonscalar Multiplications Necessary to Evaluate Polynomials”. In: *SIAM Journal on Computing* 2.1 (1973), pp. 60–66. DOI: 10.1137/0202007.

[Rai96] Eric M. Rains. “Efficient Computation of Isomorphisms Between Finite Fields”. Personal communication. 1996.

[Reg04] Oded Regev. *A Subexponential Time Algorithm for the Dihedral Hidden Subgroup Problem with Polynomial Space*. arXiv:quant-ph/0406151. June 2004. URL: http://arxiv.org/abs/quant-ph/0406151.

[Ren18] Joost Renes. “Computing Isogenies Between Montgomery Curves Using the Action of (0, 0)”. In: *Post-Quantum Cryptography*. Ed. by Tanja Lange and Rainer Steinwandt. Springer International Publishing, 2018, pp. 229–247. ISBN: 978-3-319-79063-3.

[RFB13] David Roe, Jean-Pierre Flori, and Peter Bruin. *Implement pseudo-Conway polynomials*. Trac ticket #14958. Oct. 2013. URL: https://trac.sagemath.org/ticket/14958.

[Rou99] Fabrice Rouillier. “Solving Zero-Dimensional Systems Through the Rational Univariate Representation”. In: *Applicable Algebra in Engineering, Communication and Computing* 9.5 (May 1999), pp. 433–461. ISSN: 0938-1279. DOI: 10.1007/s002000050114.

[RS06] Alexander Rostovtsev and Anton Stolbunov. *Public-key cryptosystem based on isogenies*. Cryptology ePrint Archive, Report 2006/145. Apr. 2006. URL: http://eprint.iacr.org/2006/145/.

[Sage] *SageMath, the Sage Mathematics Software System (Version 8.0)*. The Sage Developers. 2018. URL: https://www.sagemath.org.

[Sar90] Peter Sarnak. *Some applications of modular forms*. Vol. 99. Cambridge Tracts in Mathematics. Cambridge: Cambridge University Press, 1990.

[Sat00] Takakazu Satoh. “The canonical lift of an ordinary elliptic curve over a finite field and its point counting”. In: *Journal of the Ramanujan Mathematical Society* 15.4 (2000), pp. 247–270.

[Sch85] René Schoof. "Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p". In: *Mathematics of Computation* 44.170 (1985), pp. 483–494. ISSN: 00255718. DOI: 10.2307/2007968.

[Sch95] René Schoof. "Counting points on elliptic curves over finite fields". In: *Journal de Théorie des Nombres de Bordeaux* 7.1 (1995), pp. 219–254. URL: http://www.ams.org/mathscinet-getitem?mr=1413578.

[Ser70] Jean-Pierre Serre. *Cours d'arithmétique*. Paris: Presses Universitaires de France, 1970.

[Ser77] Jean-Pierre Serre. *Arbres, amalgames, SL_2*. Vol. 46. Astérisque. Paris: Société Mathématique de France, 1977.

[Sha89] Adi Shamir. "An efficient identification scheme based on permuted kernels (extended abstract)". In: *Proceedings on Advances in cryptology*. CRYPTO '89. New York, NY, USA: Springer-Verlag New York, Inc., 1989, pp. 606–609. ISBN: 0-387-97317-6.

[Sho90] Victor Shoup. "New algorithms for finding irreducible polynomials over finite fields". In: *Mathematics of Computation* 54.189 (Jan. 1990), pp. 435–435. DOI: 10.1090/s0025-5718-1990-0993933-0.

[Sho93] Victor Shoup. "Fast construction of irreducible polynomials over finite fields". In: *SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1993, pp. 484–492. ISBN: 0-89871-313-7.

[Sho94a] Peter W Shor. "Algorithms for quantum computation: Discrete logarithms and factoring". In: *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*. IEEE. 1994, pp. 124–134.

[Sho94b] Victor Shoup. "Fast construction of irreducible polynomials over finite fields". In: *Journal of Symbolic Computation* 17.5 (1994), pp. 371–391. ISSN: 0747-7171. DOI: 10.1006/jsco.1994.1025.

[Sho95] Victor Shoup. "A New Polynomial Factorization Algorithm and its Implementation". In: *Journal of Symbolic Computation* 20.4 (1995), pp. 363–397. ISSN: 0747-7171. DOI: 10.1006/jsco.1995.1055.

[Sho99] Victor Shoup. "Efficient Computation of Minimal Polynomials in Algebraic Extensions of Finite Fields". In: *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*. ISSAC '99. New York, NY, USA: ACM, 1999, pp. 53–58. ISBN: 1-58113-073-2. DOI: 10.1145/309831.309859.

[SIKE] Reza Azarderakhsh, Brian Koziel, Matt Campagna, Brian LaMacchia, Craig Costello, Patrick Longa, Luca De Feo, Michael Naehrig, Basil Hess, Joost Renes, Amir Jalali, Vladimir Soukharev, David Jao, and David Urbanik. *Supersingular Isogeny Key Encapsulation*. Nov. 30, 2017. URL: http://sike.org.

[Sil92] Joseph H. Silverman. *The arithmetic of elliptic curves*. Vol. 106. Graduate Texts in Mathematics. Corrected reprint of the 1986 original. New York: Springer-Verlag, 1992, pp. xii+400. ISBN: 0-387-96203-4.

[Sil94] Joseph H. Silverman. *Advanced Topics in the Arithmetic of Elliptic Curves*. Vol. 151. Graduate Texts in Mathematics. Springer, Jan. 1994. ISBN: 0387943285.

[Sol01] Jerome A. Solinas. *Low-Weight Binary Representations for Pairs of Integers*. Tech. rep. National Security Agency, USA, 2001.

[SS14] Igor E. Shparlinski and Andrew V. Sutherland. "On the Distribution of Atkin and Elkies Primes". In: *Foundations of Computational Mathematics* 14.2 (Apr. 2014), pp. 285–297. ISSN: 1615-3383. DOI: 10.1007/s10208-013-9181-9.

[Ste94a] Jacques Stern. "A new identification scheme based on syndrome decoding". In: *Advances in Cryptology — CRYPTO' 93*. Vol. 773. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1994. Chap. 2, pp. 13–21. ISBN: 978-3-540-57766-9. DOI: 10.1007/3-540-48329-2_2. URL: http://dx.doi.org/10.1007/3-540-48329-2%5C_2.

[Ste94b] Jacques Stern. "Designing Identification Schemes with Keys of Short Size". In: *Advances in Cryptology — CRYPTO '94*. Vol. 839. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 1994. Chap. 18, pp. 164–173. ISBN: 978-3-540-58333-2. DOI: 10.1007/3-540-48658-5_18. URL: http://dx.doi.org/10.1007/3-540-48658-5%5C_18.

[Sto09] Anton Stolbunov. "Reductionist Security Arguments for Public-Key Cryptographic Schemes Based on Group Action". In: *Norsk informasjonssikkerhetskonferanse (NISK)*. Ed. by Stig F. Mjølsnes. 2009.

[Sto10] Anton Stolbunov. "Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves". In: *Advances in Mathematics of Communications* 4.2 (2010).

[Sto12] Anton Stolbunov. "Cryptographic schemes based on isogenies". PhD thesis. Norges teknisk-naturvitenskapelige universitet, Fakultet for informasjonsteknologi, matematikk og elektroteknikk, Institutt for telematikk, Jan. 2012.

[Sut12a] Andrew V. Sutherland. "Accelerating the CM method". In: *LMS Journal of Computational Mathematics* 15 (2012), pp. 172–204. ISSN: 1461-1570. DOI: 10.1112/S1461157012001015.

[Sut12b] Andrew V. Sutherland. "Constructing elliptic curves over finite fields with prescribed torsion". In: *Mathematics of Computation* 81 (2012), pp. 1131–1147.

[Sut13a] Andrew Sutherland. "Isogeny volcanoes". In: *ANTS X: Proceedings of the Algorithmic Number Theory 10th International Symposium*. Vol. 1. Berkeley: Mathematical Sciences Publishers, 2013, pp. 507–530.

[Sut13b] Andrew Sutherland. "On the evaluation of modular polynomials". In: *The Open Book Series* 1.1 (2013), pp. 531–555.

[Sut18] Andrew V. Sutherland. *Modular polynomials*. 2018. URL: https://math.mit.edu/~drew/ClassicalModPolys.html.

[Tan09] Seiichiro Tani. "Claw finding algorithms using quantum walk". In: *Theoretical Computer Science* 410.50 (2009), pp. 5285–5297.

[Tao11] Terence Tao. *Expansion in groups of Lie type – Basic theory of expander graphs*. 2011. URL: https://terrytao.wordpress.com/2011/12/02/245b-notes-1-basic-theory-of-expander-graphs/.

[Tat66] John Tate. "Endomorphisms of abelian varieties over finite fields". In: *Inventiones mathematicae* 2.2 (Apr. 1966), pp. 134–144. ISSN: 1432-1297. DOI: 10.1007/BF01404549.

[Tes06] Edlyn Teske. "An Elliptic Curve Trapdoor System". In: *Journal of Cryptology* 19.1 (Jan. 2006), pp. 115–133. ISSN: 0933-2790. DOI: 10.1007/s00145-004-0328-3.

[Tes99] Edlyn Teske. "The Pohlig-Hellman Method Generalized for Group Structure Computation". In: *Journal of Symbolic Computation* 27.6 (1999), pp. 521–534. ISSN: 0747-7171. DOI: 10.1006/jsco.1999.0279.

[Ti17] Yan Bo Ti. "Fault attack on supersingular isogeny cryptosystems". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2017, pp. 107–122.

[TZ08] Jean-Pierre Tillich and Gilles Zémor. "Collisions for the LPS expander graph hash function". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2008, pp. 254–269.

[UJ18] David Urbanik and David Jao. "SoK: The Problem Landscape of SIDH". In: *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop*. APKC '18. Incheon, Republic of Korea: ACM, 2018, pp. 53–60. ISBN: 978-1-4503-5756-2. DOI: 10.1145/3197507.3197516.

[Vas12] Virginia Vassilevska Williams. "Multiplying matrices faster than Coppersmith-Winograd". In: *STOC'12*. ACM, 2012, pp. 887–898.

[Vél71] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus de l'Académie des Sciences de Paris* 273 (1971), pp. 238–241.

[Voi18] John Voight. *Quaternion Algebras*. 2018. URL: https://math.dartmouth.edu/~jvoight/quat-book.pdf.

[Wil82] Hugh C. Williams. "A $p+1$ method of factoring". In: *Mathematics of Computation* 39.159 (1982), pp. 225–234.

[Yoo+17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. "A Post-quantum Digital Signature Scheme Based on Supersingular Isogenies". In: *Financial Cryptography and Data Security*. Ed. by Aggelos Kiayias. Cham: Springer International Publishing, 2017, pp. 163–181. ISBN: 978-3-319-70972-7.

[Zan+18] Gustavo H. M. Zanon, Marcos A. Simplicio, Geovandro C. C. F. Pereira, Javad Doliskani, and Paulo S. L. M. Barreto. "Faster Isogeny-Based Compressed Key Agreement". In: *Post-Quantum Cryptography*. Ed. by Tanja Lange and Rainer Steinwandt. Springer International Publishing, 2018, pp. 248–268. ISBN: 978-3-319-79063-3.

[ZD06] Paul Zimmermann and Bruce Dodson. "20 Years of ECM". In: *Algorithmic Number Theory, 7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings*. Ed. by Florian Hess, Sebastian Pauli, and Michael E. Pohst. Vol. 4076. Lecture Notes in Computer Science. Springer, 2006, pp. 525–542. ISBN: 3-540-36075-1. DOI: 10.1007/11792086_37.

[Zha05] Shengyu Zhang. “Promised and Distributed Quantum Search Computing and Combinatorics”. In: *Proceedings of the Eleventh Annual International Conference on Computing and Combinatorics*. Vol. 3595. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2005. Chap. 44, pp. 430–439. ISBN: 978-3-540-28061-3. DOI: 10.1007/11533719_44.

[Zim+18] Paul Zimmermann et al. *GMP-ECM software*. 2018. URL: http://ecm.gforge.inria.fr/.