Algorithmic aspects of elliptic bases in finite field discrete logarithm algorithms

Antoine Joux, Cécile Pierrot

Manuscript submitted to
AIMS' Journals
Volume **X**, Number **0X**, XX **200X**

doi:10.3934/xx.xx.xx.xx

pp. **X–XX**

ALGORITHMIC ASPECTS OF ELLIPTIC BASES IN FINITE FIELD DISCRETE LOGARITHM ALGORITHMS

ANTOINE JOUX AND CÉCILE PIERROT

Antoine Joux*

CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
and
Sorbonne Université, Institut de Mathématiques de Jussieu–Paris Rive Gauche
CNRS, INRIA, Univ Paris Diderot.
Campus Pierre et Marie Curie, F-75005, Paris, France

Cécile Pierrot

Université de Lorraine
CNRS, Inria, LORIA
F-54000 Nancy, France

(Communicated by the associate editor name)

Abstract. Elliptic bases, introduced by Couveignes and Lercier in 2009, give an elegant way of representing finite field extensions. A natural question which seems to have been considered independently by several groups is to use this representation as a starting point for discrete logarithm algorithms in small characteristic finite fields.

This idea has been recently proposed by two groups working on it, in order to achieve provable quasi-polynomial time for discrete logarithms in small characteristic finite fields.

In this paper, we do not try to achieve a provable algorithm but, instead, investigate the practicality of heuristic algorithms based on elliptic bases. Our key idea is to use a different model of the elliptic curve used for the elliptic basis that allows for a relatively simple adaptation of the techniques used with former Frobenius representation algorithms.

We have not performed any record computation with this new method but our experiments with the field $\mathbb{F}_{3^{1345}}$ indicate that switching to elliptic representations might be possible with performances comparable to the current best practical methods.

1. **Introduction.** The discrete logarithm problem (DLP) is a fundamental problem underlying the security of many cryptographic systems. Given G a finite cyclic group denoted multiplicatively and g a generator of the group, solving the discrete logarithm problem in G means being able, for any arbitrary element $h \in G$, to find an integer x such that:

$$g^x = h.$$

2010 *Mathematics Subject Classification.* 11Y16, 12Y05, 11G20.

Key words and phrases. Discrete logarithms, Finite fields, Elliptic bases.

This work has been supported in part by the European Union as H2020 Programme under grant agreement number ERC-669891.

The integer x is defined modulo $|G|$ and is called the discrete logarithm of h.

Among the groups considered for cryptographic use, we find the multiplicative group of finite fields. There is a long history of algorithms to address this problem that we do not recall here. In the case of "small" characteristic fields, tremendous progress was made in 2013 and the years after. They are surveyed in [10]. This led to extreme computational improvements and two flavors of heuristic quasi-polynomial time algorithms. One of the fundamental tools used to achieved this result is a special representation of a finite field extension above $\mathbb{F}_q$, called the Frobenius representation. It requires an element θ satisfying a relation of the form:

$$\theta^q = \frac{h_0(\theta)}{h_1(\theta)},$$

where h_0 and h_1 are co-prime polynomials with very low degree.

A widely believed heuristic assumption is that any finite field extension can be represented that way, unless one of the known obstructions applies. These known obstructions are that h_0 and h_1 can both have degree ≤ 1. Furthermore, it is clearly not possible to represent an extension of degree higher than $q+\deg(h_1)$. In practice, finding such a representation via an exhaustive search among suitable polynomials is a trivial matter. However, proving this assumption seems to be a difficult task. Micheli [17] gave a partial answer to explain why any finite field extension could be represented that way, but we are not aware of any complete proof of this fact.

Consequently, it is natural to turn to different field representations which can provably be constructed and try to adapt the discrete logarithm algorithms to work with them. Elliptic bases, also called elliptic periods [5], form a natural candidate for this purpose. Several groups independently considered their use for discrete logarithms. We are aware of two attempts which have been made public. In 2016, in his master's thesis [14], Lido proposed a discrete logarithm based on elliptic representations using a descent method made of two halves. His presentation states a theorem concerning one-half of the descent and a conjecture for the other half. On June 26th, 2019, Kleinjung and Wesolowski released a preprint [12] on the eprint archive announcing a fully provable quasi-polynomial time algorithm based on elliptic representation. The next day, Schoof gave a talk at the conference NutMiC 2019 conference presenting the work of Lido. He also sent us a not publicly available document [15] that extends [14] and contains a theorem announcing an algorithm to compute logarithms in a finite field $\mathbb{F}$ in provable time $(\log|\mathbb{F}|)^{O(\log\log|\mathbb{F}|)}$. Lido made public his work with a complete proof of its correctness in a preprint in 2022 [16].

Whereas Lido's approach mostly relies on some Galois theory over function fields, the result announced in [12], and published in 2021 in [13] has a different form. To prove the irreductibility of certains curves, Kleinjung and Wesolowski's approach is to describe it as components of some fibered products. They show that discrete logarithms in $\mathbb{F}_{p^n}$ can be computed in provable time $(pn)^{2\log_2 n+O(1)}$. Both forms affect a very large range of characteristic. Indeed, until now, the best provable discrete logarithm algorithms for finite fields had complexity $L(1/2)$. They are thus outperformed as soon as $p < L_{p^n}(1/2-\epsilon)$, for an arbitrary small $\epsilon > 0$.

In this paper, we present the work we independently performed on a similar idea and that was first made public at the beginning of July 2019. However, we do not consider the provable aspects. Instead, we focus more on the algorithmic aspects of heuristic variants of elliptic representation discrete logarithm methods.

Our formulation differs in many details. As such, it might shed a different light on the topic and help the reader to study the theoretical breakthrough on provable algorithms. As of now, our proposal remains slightly inferior to the method in [9], the fastest currently known (heuristic) method to compute discrete logarithms in small characteristic. However, it gets very close, while leaving room for improvement in the analysis.

1.0.1. *A tool from Pomerance.* To turn an algorithm in the Frobenius representation family into a provable algorithm, it is not only necessary to prove that the finite field representation can be constructed, but it is also required to change the following steps: relation generation and linear algebra. Let h be a random element and g the generator. Essentially, one follows the approach of [20], which consists in decomposing plenty of elements of the form $g^a h^b$ over the factor base, which is a small set of small elements. Linear algebra can then be used to combine plenty of equations which leads to a random identity of the form $g^A h^B = 1$. We can then deduce the discrete logarithm of h in base g.

Assuming that the field representation exists, this makes the computation of discrete logarithms provable. For Frobenius representation algorithms, assuming that the field representation is given and deducing a provable algorithm has been studied in [8, 11, 6].

Unfortunately, the full computation has to be restarted from scratch for every discrete logarithm computation in the same field. The aim of this work is to propose a practical algorithm with a representation that can be proven. This algorithm includes an "individual logarithm step" at the end, allowing, for instance, to reuse part of the work done for another discrete logarithm that was previously computed.

1.0.2. *Discrete logarithm algorithm with an elliptic representation.* Let $\mathbb{F}_{p^k}$ be the target finite field in which we want to compute discrete logarithms. To simplify exposition, we assume $p \geq 5$, since the equation of the curves needs to be chosen differently in characteristic 2 or 3. We summarize our construction as follows:

1. **Representation.** Create an elliptic curve $\mathscr{E}$ over $\mathbb{F}_q$ (q being a power of p) such that:
$$\#(\mathscr{E}/\mathbb{F}_q) = \mu k,$$
with μ a natural integer. If k is square, there necessarily exists a k-torsion point $P_1 \in \mathscr{E}$. Otherwise, for every prime ℓ dividing k, let ℓ^{e_ℓ} be the largest power of ℓ dividing k. For each ℓ with $e_\ell > 1$, it might be necessary to change $\mathscr{E}$ to an isogeneous curve by applying a sequence of ℓ-isogenies in order to guarantee that a point of order ℓ^{e_ℓ} exists. These changes to $\mathscr{E}$ suffice to guarantee the existence of P_1.
Finally, find a point $F \in \mathscr{E}$ such that:
$$\pi(F) = F + P_1,$$
where π is the Frobenius action in the field $\mathbb{F}_q$. In particular, if we note $F = (\theta, \tau)$ then we can write $\mathbb{F}_{q^k}$ as $\mathbb{F}_q[\theta, \tau]$.
This almost gives the desired representation of the target field. Indeed, $\mathbb{F}_q[\theta]$ is either $\mathbb{F}_{q^k}$ or $\mathbb{F}_{q^{k/2}}$. In the sequel, we assume that $\mathbb{F}_q[\theta] = \mathbb{F}_{q^k}$, multiplying k by 2 if necessary.
Then, since $\mathbb{F}_{p^k}$ is a subfield of both $\mathbb{F}_{q^k}$ or $\mathbb{F}_{q^{2k}}$, we see that computing discrete logarithms in $\mathbb{F}_q[\theta]$ is sufficient to achieve the desired goal.

2. **Commutative Diagram.** We now define the full representation we want to use from a curve $\mathscr{C}$ in 3 dimensions obtained as the image of the following rational map:

$$\begin{array}{rcl} \Phi : \mathscr{E} & \mapsto & \overline{\mathbb{F}}_q^3 \\ Q & \mapsto & (x_{Q-P_1}, x_Q, x_{Q+P_1}) \end{array}$$

At first, this might seem to be a strange model of an elliptic curve, with respect to Weierstrass equations, for instance. However, the intuition is that, with this model coming from Semaev polynomials, the image of F is a point with an advantageous property: taking the Frobenius of one of its coordinates leads to the following one. In other words, if $\Phi(F)$ is seen as a point of $\mathscr{C}$ in the affine space $\mathbb{F}_q[U, V, W]$ then it lies on the intersection of the surfaces defined by the two equations $U^q = V$ and $V^q = W$. This property is at the core of our method for creating relations, since it allows us to re-use the core standard technique from the Frobenius representation approach to discrete logarithms. Starting from A and B two polynomials in $\mathbb{F}_q[U, V]$, we construct two big polynomials:

$$A^q B - A B^q = \prod_{\alpha \in \mathbb{P}_1(\mathbb{F}_q)} (A - \alpha B)$$

in one hand, and:

$$A(V, W)B(U, V) - A(U, V)B(V, W)$$

in the other hand. Each polynomial can be considered as an element of the function field $\mathbb{F}_q(\mathscr{C})$. Writing the divisor associated to each side, we can write down an equality between the image of each divisor in $\mathbb{F}_{q^k}$. For polynomial themselves, the image is simply obtained by evaluation at $\Phi(F)$. The equality of the two sides comes from the Frobenius relations between the coordinates of $\Phi(F)$.

3. **Relation collection.** We sieve on pairs of polynomials (A, B) such that $A = g_1 + \alpha g_3$ and $B = g_1 + \beta g_2 + \gamma g_3$ where $\alpha, \beta, \gamma \in \mathbb{F}_q$ and g_1, g_2 and g_3 are given polynomials constructed by linear combination of the monomials U, V, UV and 1. Set the factor base $\mathcal{F}$ as all the divisors of $\mathscr{E}$ with height at most 3. On one side $\prod_{\alpha \in \mathbb{P}_1(\mathbb{F}_q)}(A - \alpha B)$ will always lead to divisors that can be written as sum of divisors of $\mathcal{F}$, and on the other side $A(V, W)B(U, V) - A(U, V)B(V, W)$ have a low enough height so that the probability that the related divisor $\mathcal{D}$ splits in factor base elements is high enough to get as many relations we want.

   Since we only need three degrees of freedom from the four monomials U, V, UV and 1, we choose g_1, g_2 and g_3 all going through a common point. This nicely reduces the degree of the divisors appearing in the decomposition of the terms $A - \alpha B$. This is essential in making the probability of success during the relation collection phase good enough.

4. **Linear algebra and individual logarithm.** Thanks to the action of Frobenius, we can reduce the size of the factor base by a factor k. In other words, this reduces the effective size of the factor base $\mathcal{F}$ to $O(q^3/k)$. As a consequence, the cost of the sparse linear algebra, with $O(q)$ entries per equation, is of $O(q^7/k^2)$ arithmetic operations. Note that when k is chosen close to q, this matches the $O(q^5)$ asymptotic complexity obtained for this step in [9].

5. **Descent Phase.** Finally, we need a descent phase to conclude. We give the necessary tools to adapt existing methods in this context.

1.0.3. *Outline.* Section 2 gives algebraic preliminaries for this work. In this sequel, we focus on the algorithmic aspects and describe our practical elliptic Frobenius algorithm. Being aimed at practicality, this algorithm is heuristic. From a performance analysis point of view, our heuristic approach almost achieves the same efficiency as the best pre-existing practical algorithm for DLP in small characteristic finite fields. *Almost*, because there is a glitch in the analysis of the fast computation of the extended factor base. However, despite this glitch, we were able to implement and use the elliptic representation approach to compute logarithms of an extended factor base for the finite field $\mathbb{F}_{3^{1345}}$. We highlight the heuristics we use as far as we can, in order to clarify the difference with the provable algorithm of [13] or [16].

In Section 3 we give our variation on the representation of the target finite field while Section 4 details how to get relations. Section 5 deals with factor base extension and with the individual logarithms phase. Finally Section 6 presents (part of) a practical discrete logarithm computation over $\mathbb{F}_{3^{1345}}$.

2. A Refresher on the Function Field Sieve Machinery.

Notations. Let $\mathbb{K} = \mathbb{F}_q$ denote a finite field. Let $\mathscr{C}$ be a non-singular curve in the n-dimensional projective space $\mathbb{P}_n(\overline{\mathbb{F}}_q)$ defined over $\mathbb{K}$ and π denote the Frobenius map on $\mathbb{P}_n(\overline{\mathbb{F}}_q)$. The set of places of $\mathbb{K}(\mathscr{C})$ is denoted by $\Sigma_{\mathbb{K}(\mathscr{C})}$. The group of degree-0 divisors of $\mathscr{C}$ is written $\mathrm{Div}_0(\mathscr{C})$ and the Picard group (or divisor class group) of $\mathscr{C}$ is $\mathrm{Pic}_0(\mathscr{C})$. Everything that is needed in this article about classical objects such as algebraic function fields, divisors, the Picard group is given in Appendix A.

A tool from FFS. Many concepts used here originate from the Function Field Sieve (FFS) algorithm [1]. The aim of this first section is not to describe FFS itself, but to describes these concepts in a slightly more general form than the original description of Adleman and Huang article. The main tool we need from FFS consists in sending a degree-0 divisor into a finite field.

Let $\mathbb{L} = \mathbb{F}_{q^k}$ be a finite extension of $\mathbb{K}$. Let F be a point of $\mathscr{C}/\mathbb{L}$ such that the coordinates of F generate $\mathbb{L}$ over $\mathbb{K}$. Given an arbitrary element $f \in \mathbb{K}(\mathscr{C})^*$ which does not have F as a pole, we can evaluate f at F and obtain a value in $\mathbb{L}$. As explained in [1], this process can be generalized from functions to a large subset of divisors of $\mathscr{C}$. Clearly, since αf and f have the same divisor for any $\alpha \in \mathbb{K}$, we need to proceed with care.

First, we define a map Ψ from $\mathrm{Princ}(\mathscr{C})$ to $\mathbb{L}/\mathbb{K}^*$ defined as follows:

$$\begin{array}{rccc} \Psi: & \mathrm{Princ}(\mathscr{C}) & \mapsto & \mathbb{L}/\mathbb{K}^* \\ & D & \mapsto & \Psi(D) = f(F), \end{array}$$

where f is an arbitrary function such that $div(f) = D$. Since the result is only considered up to multiplication by an arbitrary constant in $\mathbb{K}^*$, it is independent of the particular choice of f.

To generalize to more divisors, we now consider a degree-0 divisor D together with an integer $h \in \mathbb{N}^*$ such that hD is principal and h is coprime to the order of $\mathbb{L}^*/\mathbb{K}^* = (q^k - 1)/(q - 1)$. For such a divisor, we extend the definition by letting:

$$\Psi(D) = \Psi(hD)^{1/h}.$$

In the terminology of [1], we evaluate at F the "surrogate" function that we have associated to D thanks to the multiplication by h. Note that replacing h by any other h' satisfying the conditions does not change the value of $\Psi(D)$. Furthermore, remark that if Ψ is defined on D and D', it is also defined on $D+D'$ and $\Psi(D+D') = \Psi(D) \cdot \Psi(D')$. Similarly, if $\Psi(D)$ is defined and non-zero, then $\Psi(-D) = 1/\Psi(D)$.

From a computational point of view, if D has a small support, then $\Psi(D)$ can be efficiently computed by using Miller's algorithm to compute the evaluation at F of the function corresponding to hD. See Section 3.3 for more details.

In the Function Field Sieve, this tool is used as part of the commutative diagram that underlies the construction of multiplicative relations. However, it is not used directly in the algorithm, only in its correctness proof. Similar, in our elliptic representation algorithm, the map Ψ is not really necessary to perform computations. However, having it at our disposal gives a very useful tool for checking the correctness of relations, thus helping to remove undesirable implementation bugs.

3. Representation of the Target Finite Field.

Let q and k respectively be a prime power and the extension degree of the field. Our aim is to compute discrete logarithms in it. Let's write p its characteristic. Since we want to define the extension as $\mathbb{F}_q[x_P]$, with x_P the abcissa of some point on an elliptic curve, while the construction naturally writes it as $\mathbb{F}_q[x_P, y_P]$, there is a small risk of producing a subextension (missing a last quadratic extension) when k is even. If this happens, it suffices to replace k by $2k$ during the initial choice of the elliptic basis. As a consequence, we can safely ignore this point in the sequel.

3.1. Choosing the elliptic basis.

The representation step of our algorithm starts by forming an elliptic curve over a small extension of $\mathbb{F}_q$ with cardinality a multiple of k. The following result explicits bounds with respect to q and k for both the extension degree and the multiplying factor.

Theorem 1. *If q and k be a prime power and a positive integer, then there exist μ and ν two integers and an elliptic curve over $\mathbb{F}_{q^\mu}$ with cardinality νk such that $\mu \leq \lceil \log(k^2/4)/\log q \rceil + 1$.*

Proof. Let p denote the characteristic of $\mathbb{F}_q$. We can perform following case by case analysis:

1. First, let us assume that $k < 2\sqrt{q}$. Thus there exist at least two multiples νk and $(\nu+1)k$ of k in the Hasse interval $\left]q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}\right[$. Furthermore, we may assume that $\nu k < q + 1$.

   If p divides k then p cannot divide the trace $t = q + 1 - \nu k$. Let us recall now the following result:

   Corollary 1 (of Waterhouse's theorem [24])**.** *For each value of the characteristic p, for any extension degree n and for every integer t in $]-2\sqrt{p^n}, 2\sqrt{p^n}[$ such that $t \not\equiv 0 \mod p$, there exists an elliptic curve over $\mathbb{F}_{p^n}$ whose number of rational point is exactly $p^n + 1 - t$.*

   The reader can for example find a proof in [22]. Note that we can run into some cases where the characteristic does divide the trace and yet such that there exists such a curve. Theorem 4 in [22] gives the exhaustive list of these special cases. Back to our discussion of the case where p divides k, we see that, Waterhouse's theorem yields the existence of an elliptic curve over $\mathbb{F}_q$ with cardinality νk.

If p does not divide k then two sub-cases occur. Either $q+1-\nu k$ is not a multiple of p, and Waterhouse's theorem permits to conclude again that there exists an elliptic curve over $\mathbb{F}_q$ with cardinality νk, or it is. In the second case, p divides $q+1-\nu k$ and doesn't divide k, so p doesn't divide $t = q+1-\nu k-k$. From Waterhouse's theorem we obtain a curve over $\mathbb{F}_q$ with cardinality $(\nu+1)k$.

Either way, when $k < 2\sqrt{q}$ we always find a curve over the base field $\mathbb{F}_q$.

2. If $k \geq 2\sqrt{q}$ then there is no guarantee of the existence of two multiples of k in the Hasse interval. Thus, unless we are lucky, we need to increase the size of the finite field to get a larger interval. Let μ be the smaller integer such that $k < 2\sqrt{q^\mu}$. Applying the previous case on this extended field, we see that there always exists an elliptic curve over $\mathbb{F}_{q^\mu}$ with cardinality νk such that $\nu k < q^\mu + 1 + 2\sqrt{q^\mu}$. To conclude, notice that the additional extension degree μ that we need is (at most) equal to the ceiling of $\log(k^2/4)/\log q$.

□

Once we have found $\mathscr{E}$, it allows us to define the finite field $\mathbb{F}_{q^{\mu k}}$, where μ is the extra extension degree needed to find $\mathscr{E}$. To lighten notations, we assume without loss of generality that $\mu = 1$. Indeed, it suffices to redefine a new value for q equal to the previous value q^μ. Thanks to the upper bound on the extension degree, we see that it does not affect the quasi-polynomial time complexity of the algorithm.

We further assume that $\mathscr{E}$ contains a point P_1 in $\mathbb{F}_q$ of order k. If necessary, apply low-degree isogenies to the initial curve $\mathscr{E}$ until a suitable one is obtained. Then construct a point F whose coordinates in the algebraic closure satisfy:

$$\pi(F) = F + P_1,$$

where π is the q-th power Frobenius action. Write the coordinates of F as (θ, τ). From [5], we know that $\mathbb{F}_{q^k} = \mathbb{F}_q[\theta, \tau]$. Furthermore, from our assumption on k, we have $\mathbb{F}_{q^k} = \mathbb{F}_q[\theta]$.

Note that, when focusing on the practical variation of the algorithm, it is important to have k as large as possible compared to q. The above proof only guarantees that $q = O(k^2)$, however, in the best cases we can have $q = O(k)$.

3.2. **Representing the curve with a different model.** We introduce here a model that represents elliptic curves in the three-dimensional affine space $\mathbb{F}_q[U, V, W]$. To construct this new model, we start from a curve $\mathscr{E}$ together with a k-torsion point P_1 of $\mathscr{E}/\mathbb{F}_q$. Let $(x_1, y_1) \in (\mathbb{F}_q)^2$ denote the coordinates of P_1 and (x_ℓ, y_ℓ) the coordinates of $P_\ell = \ell\, P_1$ for $\ell \in [2, \cdots, k-1]$.

3.2.1. *Adding some structure.* The idea is to create a new model of $\mathscr{E}$ in which we artificially inject extra desirable properties. Namely, for any point $Q \in \mathscr{E}/\overline{\mathbb{F}}_q$, we represent it by the triple of abcissae of the points $Q - P_1$, Q and $Q + P_1$, on the one hand and $-Q, P_1, Q - P_1$ on the other hand. In this model, there is an easy way to add the $\pi(F) = F + P_1$ constraint of the Couveignes and Lercier [5] construction of elliptic bases. The Frobenius action on a special point called F to be defined later is shown on Figure 1.

Indeed, for point F we see that the triple of coordinates is $(x_{\pi^{-1}(F)}, x_F, x_{\pi(F)})$. Furthermore, for $\pi(F)$ the triple is $(x_F, x_{\pi(F)}, x_{\pi^2(F)})$. As a consequence, the first two coordinates can be obtained by a simple shift. Furthermore, it is possible to recover the missing coordinate of $\pi(F)$ from the first two, in a way similar to Montgomery's ladder technique.

3.2.2. *Formal definition and equations of $\mathscr{C}$.* Now that we have captured our intuition, we give a formal definition of this new curve. We assume for simplicity that $\mathscr{E}$ is given by a reduced Weirstrass equation:

$$\mathscr{E} : Y^2 = X^3 + a\,X + b,$$

but this can be generalized to include characteristic 2 and 3. With this notation, the third summation polynomial is given by:

$$S_3(X_1, X_2, X_3) = 4\sigma_1(\sigma_3 + b) - (\sigma_2 - a)^2,$$

where the σ_i are the symmetric polynomials $\sigma_1 = X_1 + X_2 + X_3$, $\sigma_2 = X_1\,X_2 + X_1\,X_3 + X_2\,X_3$ and $\sigma_3 = X_1\,X_2\,X_3$. By definition of the third Semaev polynomial, for any triple of points $Q_1 = (x_{Q_1}, y_{Q_1}), Q_2 = (x_{Q_2}, y_{Q_2})$, $Q_3 = (x_{Q_3}, y_{Q_3}) \in \mathscr{E}(\bar{\mathbb{F}}_q) \setminus \{\mathcal{O}\}$, we have:

$$S_3(x_{Q_1}, x_{Q_2}, x_{Q_3}) = 0 \Leftrightarrow \exists (e_1, e_2, e_3) \in \{-1, 1\}^3, e_1Q_1 + e_2Q_2 + e_3Q_3 = \mathcal{O}.$$

We use this polynomial to describe the image $\mathscr{C}$ of the rational map:

$$\begin{array}{rcc} \Phi : \mathscr{E} & \mapsto & \overline{\mathbb{F}}_q^3 \\ Q & \mapsto & (x_{Q-P_1}, x_Q, x_{Q+P_1}) \end{array}$$

For every point $Q \in \mathscr{E}/\bar{\mathbb{F}}_q$, we use S_3 to rewrite the three simple identities $(Q - P_1) - Q + P_1 = \mathcal{O}$, $Q - (Q + P_1) + P_1 = \mathcal{O}$ and $(Q - P_1) - (Q + P_1) + P_2 = \mathcal{O}$. This shows that the point $(x_{Q-P_1}, x_Q, x_{Q+P_1})$ is a common root of the polynomials $S_3(U, V, x_1)$, $S_3(V, W, x_1)$, and $S_3(U, W, x_2)$. Let us first consider the variety given by these three equations. To determine its components, let us consider its intersection with the hyperplane $U = W$. This intersection is described by $S_3(U, V, x_1) = 0$, $U = W$ and $S_3(U, U, x_2) = 0$. From Equation $S_3(U, W, x_2)$ that is a degree-4 polynomial in U, we know that U has finitely many values. Thus we want to remove the extraneous points lying in this hyperplane.

We look so at the components in the complement of this hyperplane and assume that $U \neq W$. In this case, since $S_3(U, V, x_1) - S_3(V, W, x_1)$ is divisible by $U - W$ we obtain a lower degree polynomial, namely:

$$S_\delta = \frac{S_3(U, V, x_1) - S_3(V, W, x_1)}{U - W}.$$

The variety defined thanks to the three equations $S_3(U, V, x_1) = 0$, $S_\delta = 0$ and $S_3(U, W, x_2) = 0$ is now irreducible. We call it $\mathscr{C}$ and prove that it is a genus 1 curve isomorphic to $\mathscr{E}$. To see that, let us give rational maps between $\mathscr{E}$ and $\mathscr{C}$.

3.2.3. *$\mathscr{E}$ and $\mathscr{C}$ are isomorphic.* Let us consider the rational map:

$$\begin{array}{rcc} \Phi : \mathscr{E} & \to & \mathscr{C} \\ Q & \mapsto & (x_{Q-P_1}, x_Q, x_{Q+P_1}) \end{array}$$

Every point P in $\mathscr{E}$ is such that $\Phi(P) \in \mathscr{C}$. Besides, the images of the three points $\mathcal{O}$, P_1 and $-P_1$ are at infinity on $\mathscr{C}$ and that, by homogenization, we may check that there are exactly three points at infinity on $\mathscr{C}$.

As usual, Φ induces a map Φ^* from the function field $\mathbb{F}_q(\mathscr{C})$ to $\mathbb{F}_q(\mathscr{E})$ (expressed with the two variables X and Y) using the following replacement:

$$\begin{array}{rccl} \Phi^* : & \mathbb{F}_q(\mathscr{C}) & \mapsto & \mathbb{F}_q(\mathscr{E}) \\ & U & \mapsto & \left(\frac{Y+y_1}{X-x_1}\right)^2 - X - x_1, \\ & V & \mapsto & X, \\ & W & \mapsto & \left(\frac{Y-y_1}{X-x_1}\right)^2 - X - x_1. \end{array}$$

where y_1 is the ordinate of the point P_1 in $\mathscr{E}$. Developing and using the curve equation, the images of U and W can be respectively simplified to:

$$\begin{aligned} U &\mapsto \frac{x_1 X^2 + (a + {x_1}^2)\, X + a\, x_1 + 2\, b + 2\, y_1\, Y}{(X - x_1)^2}, \\ W &\mapsto \frac{x_1 X^2 + (a + {x_1}^2)\, X + a\, x_1 + 2\, b - 2\, y_1\, Y}{(X - x_1)^2}. \end{aligned}$$

In this form, it is clear that the map can be easily inverted when $X \neq x_1$. Moreover, given a pair (U, V) values we can compute the value of W and similarly, from (V, W) we can compute U.

When $X = x_1$, we have two possibilities $\Phi(P_1) = (\infty, x_{P_1}, x_{P_2})$ and $\Phi(-P_1) = (x_{P_2}, x_{P_1}, \infty)$. These are distinct (unless P_1 has order 2), which means that Φ is a bijection and thus an isomorphism.

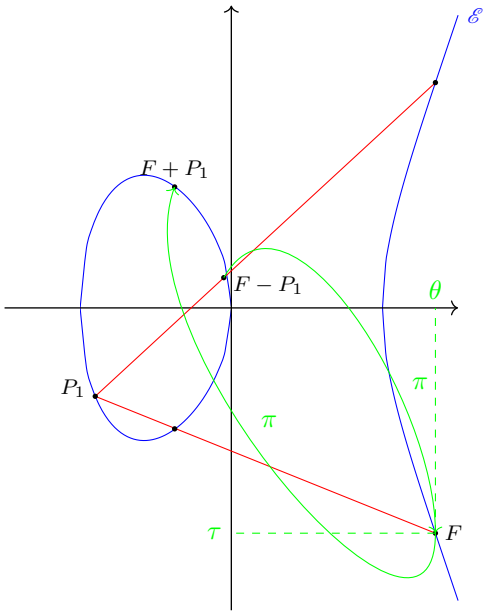

FIGURE 1. Frobenius action on the point $F \in \mathscr{E}/\mathbb{F}_{q^k}$.

3.2.4. *Point in $\mathscr{E}$ with coordinates in the target finite field.* Let $F \in \mathscr{E}$ be such that:

$$\pi(F) = F + P_1. \tag{1}$$

Lemma 1. *If $F \in \mathscr{E}/\bar{\mathbb{F}}_q$ is such that $\pi(F) = F + P_1$ then $F \in \mathscr{E}/\mathbb{F}_{q^k}$. Furthermore, letting (θ, τ) denote the coordinates of F, we have $\mathbb{F}_q[\theta, \tau] = \mathbb{F}_{q^k}$. In particular, there exist at least k rational points verifying the same property.*

Proof. Let us compute $\pi^k(F) = \pi^{k-1}(F + P_1) = \pi^{k-1}(F) + P_1 = \cdots = F + kP_1$. We know that P_1 has precisely order k hence $\pi^k(F) = F$. Besides, we note that any point $\pi^i(F)$ for $i = 1, \cdots, k-1$ satisfies Equation (1) too. □

As already mentioned, having possibly doubled the value of k in construction, we may assume that $\mathbb{F}_q[\theta] = \mathbb{F}_{q^k}$. In our model of curve, the point F is determined by the fact that $S_3(\theta, \theta^q, x_1) = 0$. The abscissa θ of F can thus be determined as a root of this polynomial. Note that the choice of the ordinate τ gives an orientation on the direction of the Frobenius action. We choose the letter F to name this point as a mnemonic to remind that it represents our target Finite Field and that it has a special relationship with the Frobenius map.

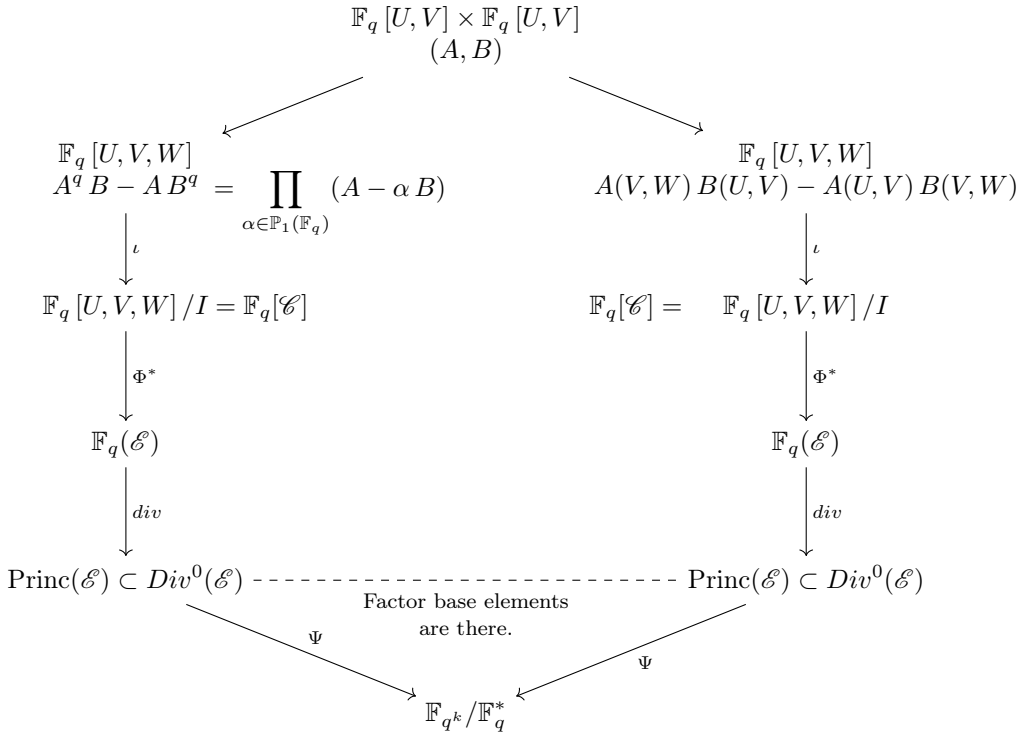

FIGURE 2. Commutative diagram of our algorithm.

3.3. **Commutative diagram.** From the above considerations, we derive the commutative diagram of Figure 2, which serves as the basis for our elliptic Frobenius representation algorithm. Note that the commutative diagram is above $\mathbb{F}_{q^k}/\mathbb{F}_q^*$. As a consequence, our algorithm doesn't compute the part of the discrete logarithm corresponding to $\mathbb{F}_q^*$. However, this field is so small that this missing part can be easily obtained.

Remark 1. *The diagram of Figure 2 could be simplified by removing references to the function field $\mathbb{F}_q(\mathscr{E})$ and computing divisors on $\mathscr{C}$ directly. However, when using standard computer algebra tools, it is much simpler to work on divisors with the Weirstrass equation of $\mathscr{E}$.*

3.3.1. *Explicit maps to $\mathbb{F}_{q^k}$ based on Miller algorithm.* The first two maps of the diagram are explicit and the two following ones are canonical injections. Φ^* is given in Appendix and *div* is as defined in Section 2. As the composition if frequent we note $\Xi = div \circ \Phi^*$ to lighten the ratings. In addition, we let Ψ denote a multiplicative group morphism that sends elements of $\mathrm{Princ}(\mathscr{E})$ to $\mathbb{F}_{q^k}$. Yet, only defining Ψ for principal divisors is not sufficient, since in the relation collection phase we need first to factor divisors in $\mathrm{Princ}(\mathscr{E})$ into elementary divisors before descending them into the finite field. Keep in mind that elementary divisors have no reason to be principal.

For any divisor in $\mathrm{Princ}(\mathscr{E})$ the first thing to do is to decompose it into elementary divisors in Div^0. Then, we note that since we wish to construct a group morphism that sends elements of Div^0 to $\mathbb{F}_{q^k}/\mathbb{F}_q^*$, it suffices to describe this morphism for any of these elementary divisors. Let us consider:

$$\mathscr{D}_e = \sum_{i=1}^{d} \pi^i(Q) - d(\mathcal{O}),$$

where $Q \in \mathscr{E}/\mathbb{F}_{q^d}$ is one of the conjugate points in the degree-d place. Fix a maximum degree D for the places we consider and let N_D be the least common multiple of the cardinalities of $\mathscr{E}$ over each finite field $\mathbb{F}_{q^d}$ with $1 \leq d \leq D$.

From this, we see that $N_D\mathscr{D}_e$ is a principal divisor, thus there exists a function $f_{\mathscr{D}_e}$ in the variables X and Y unique up to multiplication by a constant in $\mathbb{F}_q$ such that $\mathrm{div}(f_{\mathscr{D}_e}) = N_D\mathscr{D}_e$. We want to use the point F with coordinates in the target finite field to define it. Since θ and τ are respectively the abscissa and the ordinate of this point, it seems natural to send X to θ and Y to τ, or, in other words, to evaluate the function on the point F. However, since $f_{\mathscr{D}_e}$ is only defined modulo a constant in $\mathbb{F}_q$, the result in the finite field would change depending on the choice of the function. To annihilate this constant, we have to divide the evaluation on F by the evaluation on $\mathcal{O}$. Hence to have a well-defined application Ψ we set

$$\Psi(\mathscr{D}_e) = \left(f_{\mathscr{D}_e}(F - \mathcal{O})\right)^{1/N_D}.$$

However, evaluation at $\mathcal{O}$ isn't really necessary, since we are only interested in values in $\mathbb{F}_{q^k}/\mathbb{F}_q^*$. As done for bilinear pairings, this can be efficiently computed using Miller's algorithm [18].

3.3.2. *Analysis of the invertibility of N_D.* In order to raise to the power $1/N_D$ and get a uniquely defined value, we need to check that N_D is invertible modulo the order of $\mathbb{F}_{q^k}^*/\mathbb{F}_q^*$. This condition needs to be tested for all the orders of the curve $\mathscr{E}$ in the extension fields $\mathbb{F}_q, \mathbb{F}_{q^2}, \cdots, F_{q^D}$. We provide a replacement for Ψ in the case where N_D cannot be inverted.

This analysis requires us to follow standard practice and first decompose our target group $\mathbb{F}_{q^k}^*$, in order to apply Pohlig-Hellman algorithm [19]. Thanks to [19] it suffices to compute discrete logarithms in all prime order subgroups of $\mathbb{F}_{q^k}^*$. An important technicality is that we would need to first factor $q^k - 1$. Unfortunately, this would dominate the cost of computation. However, to study the invertibility of N_D, we do not need to factor $q^k - 1$, the existence of the factorization suffices.

Let γ be a prime factor dividing $q^k - 1$, it suffices to check that $N_D \neq 0 \mod \gamma$, for Ψ to be well-defined in subgroups of order a power of γ in $\mathbb{F}_{q^k}^*$. Since N_D is defined as the least common multiple of the cardinalities of $\mathscr{E}$ over each of the finite fields $\mathbb{F}_{q^d}$ with $1 \leq d \leq D$, where $D \leq k$ is the maximum degree of the places we want to consider, this gives us an extra condition on $\mathscr{E}$. Namely, it should satisfy the following property:

- For any $i = 1, \cdots, D$, $|\mathscr{E}/\mathbb{F}_{q^i}| \neq 0 \mod \gamma$.

Let us study this condition. We denote by t the trace of $\mathscr{E}$ over $\mathbb{F}_q$ and factor the characteristic polynomial of the Frobenius of $\mathscr{E}$:

$$X^2 - tX + q = (X - r)(X - s) \mod \gamma$$

with r and s in $\mathbb{F}_{\gamma^2}$. We know that the number of points of $\mathscr{E}/\mathbb{F}_{q^i}$ is equal to $(1 - r^i)(1 - s^i) \mod \gamma$. Thus, to ensure that the cardinalities of $\mathscr{E}$ over the field extensions $\mathbb{F}_{q^i}$, with ι in $[1, D]$, all differ from 0 modulo γ we just need to verify that both $r^i \neq 1 \mod \gamma$ and $s^i \neq 1 \mod \gamma$. In order to do that, let us first study the order of the product $rs = q \mod \gamma$.

By definition of γ, we have $q^k = 1 \mod \gamma$. Furthermore, the order of q is strictly smaller than k modulo γ if and only if γ already divides the order of the multiplicative group of a subfield of $\mathbb{F}_{q^k}$. In that case, we compute this part of the logarithm by applying our method to the smallest such subfield.

We now assume that the order of q is precisely $k \mod \gamma$. Thus, for any ι not a multiple of k, at most one of r^i or s^i can be equal to $1 \mod \gamma$. Exchanging r and s if necessary, we now study the case $r^i = 1 \mod \gamma$. In that case, we have $s^i = q^i \neq 1 \mod \gamma$. This implies that $\mathscr{E}(\mathbb{F}_{q^i})$ contains a γ-torsion point Q_γ but not the full γ-torsion $\mathscr{E}[\gamma]$. Thus, the Tate pairing provides a non-degenerate bilinear map to the γ roots of unity:

$$e_i :< Q_\gamma > \times \mathscr{E}(\mathbb{F}_{q^i})/\gamma\mathscr{E}(\mathbb{F}_{q^i}) \mapsto \mathbb{F}_q^*/\left(\mathbb{F}_q^*\right)^{(q^k-1)/\gamma}.$$

Fixing an arbitrary non-zero element from $\mathscr{E}(\mathbb{F}_{q^i})/\gamma\mathscr{E}(\mathbb{F}_{q^i})$, we obtain a linear map $\tilde{\Psi}_i$ from the subgroup generated by Q_γ to the γ-th roots of unity.

Possibly after renormalization, $\tilde{\Psi}_i$ gives a compatible replacement for Ψ that can be applied to the γ-torsion point. The renormalization consists in replacing $\tilde{\Psi}_i$ by $\tilde{\Psi}_i^{\beta_i}$, where β_i is the renormalization constant. As a consequence, it is mathematically possible to extend Ψ to all divisors. One computational caveat is that determining the value of β_i can be expressed as a discrete logarihm problem in the group of order γ. It does not affect the efficiency of the overall algorithm but prevents independent check on relations containing divisors of degree ι not compatible with the definition of Ψ during the precomputation phase.

To see how β_i can be determined, let us take a place p_i of degree ι and compute $\tilde{\Psi}_i(p_i)$. As usual this it the product of the value for all the conjugate points in p_i. Then apply one step of the descent algorithm to relate p_i to places of degrees $\neq i$ which are all compatible with the computation of Ψ. Multiplying these contributions gives the renormalized value $\tilde{\Psi}_i(p_i)^{\beta_i}$. Thus, if we wish to do so, we can compute β_i from the individual logarithms of these two values.

3.3.3. *Intuition about Ψ and commutativity of the diagram.* This definition matches with the following intuitive one. To unsure the commutativity of the diagram we need to verify that $\Psi(\Xi(\iota(A^qB - AB^q)))$ is equal in the finite field to the element

$\Psi(\Xi(\iota(A(V,W)B - AB(V,W))))$. Our intuition is that requiring in some sense:

$$U^q = V \qquad \text{and} \qquad V^q = W$$

would suffice to prove the commutativity. We point out that one point of the elliptic curve $\mathscr{C}$, namely $\Phi(F)$, precisely follows this restriction. Indeed, $\Phi(F)$ has abscissa $x_{F-P_1} = x_{\pi^{(k-1)}(\pi(F-P_1))} = x_{\pi^{(k-1)}(F)} = \pi^{(k-1)}(\theta)$, ordinate $x_F = \theta$ and applicate $x_{F+P_1} = x_{\pi(F)} = \pi(\theta)$. In a nutshell:

$$\Phi(F) = [\theta^{q^{k-1}}, \theta, \theta^q]$$

Thus, in the function field $\mathbb{F}_q(\mathscr{C})$, evaluation the functions at $\Phi(F)$ gives the expected relationship to the Frobenius map. As a consequence, evaluation at F after transporting back to the function field of $\mathscr{E}$ using Φ^* also gives the desired behavior.

To conclude about the commutativity of the diagram it suffices to note that we have the equalities in $\mathbb{F}_q^*$:

$$\Psi \circ div \circ \Phi^*(U^q) = (\theta^{q^{k-1}})^q = \theta = \Psi \circ div \circ \Phi^*(V)$$

$$\text{and} \quad \Psi \circ div \circ \Phi^*(V^q) = \theta^q = \Psi \circ div \circ \Phi^*(W).$$

Hopefully, since $\Psi \circ div \circ \Phi^* \circ \iota$ is a morphism the equality in the finite field between images of $A^qB - AB^q$ and $A(V,W)B - AB(V,W)$ holds too.

3.3.4. *Extension of the diagram to bigger fields.* As with classical Frobenius representation algorithm, the commutative diagram can also be used when the coefficients of A and B are taken in another extension $\mathbb{F}_{q^d}$. In that case, the commutative diagram ends in the compositum of $\mathbb{F}_{q^d}$ and $\mathbb{F}_{q^k}$. The only difference is that the identity in the finite field is between the images of $A^qB - AB^q$ and $A^\pi(V,W)B - AB^\pi(V,W)$, i.e. the coefficients of A and B need to be acted on by Frobenius.

4. **Harvesting Relations.** This section details the necessary tools to collect relations in our setting which is a mixture of the classical Function Field Sieve and of the Frobenius representation algorithms. As in the classical Function Field Sieve, our algorithm uses function fields instead of polynomials when writing down multiplicative relations. From Frobenius representation algorithms it inherits the use of a systematic form of relations.

4.1. **The usual systematic product.** We recall the systematic relation that we inherit from Frobenius representation algorithms:

$$A^q\, B - A\, B^q = \prod_{\alpha \in \mathbb{P}_1(\mathbb{F}_q)} (A - \alpha\, B), \tag{2}$$

where as in [9], when α is the point at infinity of $\mathbb{P}_1(\mathbb{F}_q)$, the term $A - \alpha\, B$ is used as a shorthand for B. For simplicity, we also use a bracket notation and define:

$$[A, B] = A(V,W)\, B(U,V) - A(U,V)\, B(V,W).$$

We underline that our bracket is $\mathbb{F}_q$-bilinear and antisymmetric, as in [9]. Yet, we warn the reader of the difference between the definition of our bracket and previous ones. Our bracket is equal to the entire fraction whereas the authors of [9] only consider the numerator of this rational fraction.

4.2. **Choice of A and B.** In the commutative diagram of Figure 2 and in the above discussion, we indicate that relations are obtained from a choice of two bivariate polynomials A and B in U and V. However, we need to specify how these polynomials are chosen and which monomials they should contain.

As a preliminary, let us notice that (A, B) and $(\alpha A, B)$ for $\alpha \in \mathbb{F}_q$ lead to the same relation. Indeed, $(\alpha A)^q\, B - (\alpha A)\, B^q = \alpha(A^q\, B - A\, B^q)$ so the two divisors associated to the two corresponding functions are equals. Thus A and B are chosen as some kind of monic polynomials: the coefficient of the higher monomial (in the lexicographic order for instance) must be equal to 1. Then we note that monomials divisible by $(UV)^2$ are not useful in A and B. Indeed, when going to $\mathbb{F}_q[\mathscr{C}]$, reduction modulo $S_3(U, V, x_1)$ transforms these monomials into *smaller* monomials $U^2V, UV^2, U^2, UV, V^2, U, V$ and 1. Moreover, it is natural to consider sets of monomials globally symmetric in U and V.

As a consequence for all those items, given a parameter $t \geq 1$ we construct A and B as linear combinations of monomials from:

$$\mathcal{M}_t = \left\{U^i, V^i, U^i\, V, U\, V^i | i \in [0 \cdots t]\right\}.$$

Each $\mathcal{M}_t$ contains $4t$ distinct monomials[1].

4.3. **Defining a naive factor base.** Let $\mathscr{D} = \sum_{Q_i \in \mathcal{V}} e_i Q_i$ be a divisor of $Div(\mathcal{V})$. The degree of the zero of $\mathscr{D}$ (or height of $\mathscr{D}$) is $h(\mathscr{D}) = \sum_{e_i > 0} e_i$. Note that, for any prime or elementary divisor, the height is equal to the degree of the corresponding place. This notion of size is needed to define the factor base, which is as a subset of $Div^0(\mathscr{E})$, as it is shown in Figure 2. We now explain how this subset is chosen.

4.3.1. *Left side.* Following the ideas of all Frobenius representation algorithms, we define the factor base such that the images of $A - \alpha B$ in this set are small. Doing this improve the relation collection phase compared to a classical sieving. Indeed, all elements in the left part of the diagram will belong to the factor base. The relation collection phase produces divisors of the form $\Xi(i(\prod(A - \alpha B)))$, so, thanks to the fact that we consider morphisms, it yields a sum of divisors $\sum(\Xi(i(A - \alpha B)))$. As explained, we require all the divisors noted by:

$$\Xi(i(A - \alpha B))$$

to be in the factor base.

Let us find the maximal height they can reach. To do so we set A and B two linear combinations of monomials in $\mathcal{M}_t$ and t an integer parameter to define later. All polynomials $A - \alpha B$ are so linear combinations of monomials in $\mathcal{M}_t$ too. In other words, we are considering divisors of functions of $\mathbb{F}_q(\mathscr{E})$ of the form $\Phi^*(\sum_{m \in \mathcal{M}_t} a_m m) = \sum_{m \in \mathcal{M}_t} a_m \Phi^*(m)$ where a_m are constants in the base field. We know that the height of the divisors we obtain in the left part of the diagram are dominated by the largest height achieve for any $\Xi(m)$, with $m \in \mathcal{M}_t$.

Considering $h(\Xi(U^t V^{t'})) \leq t\, h(\Xi(U)) + t'\, h(\Xi(V))$, we see that it suffices to determine both the height of the divisors associated to the images of U and V in the function field of $\mathscr{E}$. The most significant monomials will be $U^t V$ and UV^t.

[1] Not $4(t+1)$ since each of 1, U, V and UV are included twice.

Functions of $\mathscr{C}$	Height of the associated divisors in $\mathscr{E}$
1	0
U, V, W	2
UV, VW, UW	4
$U+V$, $V+W$, $U+W$	4
U^tV, UV^t	$2t+2$
U^tV+UV^t	at most $2t+2$
$U^tV^{t+1}W$, $UV^{t+1}W^t$, $U^tV^2W^t$, $UV^{2t}W$	$4t+4$

TABLE 1. Some functions of $\mathbb{F}_q[\mathscr{C}]$ and their height in $\mathscr{E}$.

On the one hand we have $\Xi(V) = div(X) = ([0,\sqrt{b},1]) + ([0,-\sqrt{b},1]) - 2(\mathcal{O})$ where $\sqrt{b}$ is an element[2] in $\bar{\mathbb{F}}_q$ such that its square is equal to b. Hence:

$$h(\Xi(V)) = 2. \tag{3}$$

On the other hand, $\Xi(U) = div((Y+y_1)^2 - (X-x_1)^3) - div((X-x_1)^2)$. From $div((Y+y_1)^2-(X-x_1)^3) = 2(-P_1)+(Q_1)+(Q_2)-4(\mathcal{O})$, where Q_1 and Q_2 are two conjugated points of a degree-2 place, and $div((X-x_1)^2) = 2(P_1)+2(-P_1)-4(\mathcal{O})$, it comes $\Xi(U) = (Q_1)+(Q_2)-2(P_1)$. We obtain:

$$h(\Xi(U)) = 2. \tag{4}$$

Putting Equations (3) and (4) together with the upper bound, we conclude that the most significant monomials U^tV and UV^t have both height $2t+2$. Yet $\Phi^*(U^tV)$ and $\Phi^*(UV^t)$ do not share the same denominator so to count their respective contribution in the height of divisors $\sum_{m\in\mathcal{M}_t} a_m\Phi^*(m)$ we need to add the contribution of the residual denominator. Namely, since $\Phi^*(U^t)$ brings the largest denominator, the height of the divisor of $\Phi^*(U^tV)$ does not change, but for the one of $\Phi^*(UV^t)$ we need to add the number of zeros corresponding to the denominator of $\Phi^*(U^t/U)$. We note that there is $2(t-1)$ such points. To put it in a nutshell, the most significant monomial is UV^t and **all divisors on the left are sum of divisors with a height upper-bounded by 4t** $= (2t+2)+2(t-1)$.

To conclude, starting the relation collection phase with $t=1$ it is natural to set the initial factor base as included in the set of divisors of Div^0 with height equal or lower than 4. We emphasize that in this case, all the divisors appearing in the left part belong to the factor base.

4.3.2. *Right side.* On the right part of the diagram, divisors are given through the extra variable W. We can compute the corresponding height exactly as for U. Again it gives:

$$h(\Xi(W)) = 2.$$

Let us consider the polynomials of $\mathbb{F}_q[U,V,W]$ given on this side and write this time $\mathcal{M}_t^{VW} = \{V^i, W^i, V^i\,W, V\,W^i | i\in[0\cdots t]\}$. Sorting the monomials in the lexicographic order, we recall that the leading coefficient for both A and B can be chosen equal to 1. Thus, keeping the leading monomial U^tV apart and calling a_m

[2]Be careful, here we assume that the characteristic differs from 2. If not, we just consider the corresponding degree-2 place.

Functions of $\mathscr{C}$		Height of the associated divisors in $\mathscr{E}$
$A(U,V) - \alpha B(U,V)$	where $\alpha \in \mathbb{F}_q$	at most $4t$
$A(V,W)B(U,V) - A(U,V)B(V,W)$		at most $8t$

TABLE 2. Functions appearing on both side of the diagram, and their corresponding heights in $\mathscr{E}$. A and B are linear combinations of monomials from $\mathcal{M}_t$.

(resp. b_m) the coefficients in $\mathbb{F}_q$ of A (resp. B) we obtain on the right side the polynomial:

$$\begin{aligned}[A,B] &= A(V,W)B(U,V) - A(U,V)B(V,W) \\ &= \left(V^tW + \sum_{m \in \mathcal{M}_t^{VW} \setminus \{V^tW\}} a_m m\right)\left(U^tV + \sum_{m \in \mathcal{M}_t \setminus \{U^tV\}} b_m m\right) \\ &\quad - \left(U^tV + \sum_{m \in \mathcal{M}_t \setminus \{U^tV\}} a_m m\right)\left(V^tW + \sum_{m \in \mathcal{M}_t^{VW} \setminus \{V^tW\}} b_m m\right)\end{aligned}$$

Since the monomial $U^tV^{t+1}W$ vanishes it yields a linear combination of monomials where the three that dominate the height of the associated divisor are $UV^{t+1}W^t$, $U^tV^2W^t$ and $UV^{2t}W$. Indeed, each variables U, V and W contributes the same way, thus, the most important monomials are those with the highest additive degree. We note then that we have $h(\Xi(UV^{t+1}W^t)) = h(\Xi(U^tV^2W^t)) = h(\Xi(UV^{2t}W)) = 4t + 4$. Yet, again, we need to carefully add the zeros raised by the residual denominator. The contribution of $\Phi^*(U^tV^2W^t)$ is left unchanged but we must add the number of poles of $\Phi^*(W^t/W)$ to the height of the divisor associated to $\Phi^*(UV^{t+1}W^t)$ and the one corresponding to the denominator of $\Phi^*((UW)^t(UW)^{-1})$ to the height of the divisor associated to $\Phi^*(UV^{2t}W)$. Since there are respectively $2(t-1)$ and $4(t-1)$ such points, we conclude that **all the divisors appearing on the right side are twice as large as factor base elements since they have height equal or lower than 8t** $= 4t + 4 + 4(t-1)$. In particular, when $t = 1$, this gives divisors of height 8 at most.

4.3.3. *Complexity of the linear algebra with a naive factor base.* A first and naive choice of factor base is made of all divisors of height ≤ 4.

However, this factor base is too large to be competitive when compared to the best Frobenius representation algorithms. To show this, let us briefly analyze the number of operations needed to perform linear algebra with this factor base. The number of divisors in this naive factor base is dominated by the number of degree 4 places on the curve $\mathscr{E}$, corresponding to polynomials of degree 4 with coefficients in $\mathbb{F}_q$. So the order of the factor base's size is dominated by q^4. Considering the Frobenius action of Section 4.4 that permits to divide the size of the factor base by $k \approx q$, we obtain a factor base of size $O(q^3)$. Since there are q terms in each linear equations, performing a sparse linear algebra step can be done in $O((q^3)^2 q) = O(q^7)$ operations.

As a comparison, we see that the first phase of the algorithm in [9] only has a $O(q^6)$ complexity. Thus, we need to improve our initial factor base.

4.4. **Action of Frobenius and translation by P_1.** Let d be the largest possible height of an elementary divisor of the factor base. We would like to explicit how the action of Frobenius on elements of the finite field $\mathbb{F}_{q^k}$ is related to addition of $-P_1$ on the elliptic curve $\mathscr{E}$. Considering the divisors:

$$\mathscr{D}_Q = \sum_{i=0}^{d-1}(\pi^i(Q)) - d(\mathcal{O})$$

related to the place given by any point $Q \in \mathscr{E}$ and:

$$\mathscr{D}_{Q-P_1} = \sum_{i=0}^{d-1}(\pi^i(Q-P_1)) - d(\mathcal{O})$$

related to the translation of Q by $-P_1$. We show that the two discrete logarithms satisfy a simple relation. More precisely, we have the following result:

Lemma 2. *Let $Q \in \mathscr{E}$ be a point with coordinates in $\mathbb{F}_{q^d}$. We consider the divisors*

$$\mathscr{D}_Q = \sum_{i=0}^{d-1}(\pi^i(Q)) - d(\mathcal{O}) \quad \textit{and} \quad \mathscr{D}_{Q-P_1} = \sum_{i=0}^{d-1}(\pi^i(Q-P_1)) - d(\mathcal{O})$$

respectively related to the place given by the point Q in $\mathscr{E}$ and the one given by the translation of Q by $-P_1$. Then:

$$\Psi(\mathscr{D}_{Q-P_1}) = \pi(\Psi(\mathscr{D}_Q)) \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d}.$$

where N_d is divisible by all the cardinalities of $\mathscr{E}$ over each finite field $\mathbb{F}_{q^i} \subset \mathbb{F}_{q^d}$.

Proof. Let us start from $\mathscr{D}_Q$ the degree-d divisor. We recall that to have a principal divisor we need to consider $N_d\mathscr{D}_Q$. Thanks to Miller algorithm we are able to recover a function f_Q with coefficients in the base field $\mathbb{F}_q$ such that $N_d\mathscr{D}_Q$ is the divisor of this function. By definition we obtain: $\Psi(\mathscr{D}_Q) = f_Q(F-\mathcal{O})^{1/N_d}$. To simplify the notation let us write $\alpha = f_Q(\mathcal{O})^{1/N_d}$ that is an element in $\mathbb{F}_q$. Hence on the one hand we have:

$$\begin{array}{rcll} \pi(\Psi(\mathscr{D}_Q)) & = & \pi(f_Q(F)^{1/N_d})/\pi(\alpha) & \\ & = & f_Q(\pi(F))^{1/N_d} \cdot \alpha^{-1} & \text{since } \alpha \in \mathbb{F}_q. \end{array} \tag{5}$$

To link this expression to the divisor of $\pi(Q)$, *i.e.* to the evaluation of f_Q in the point $\pi(F) = F + P_1$ we define the function g_Q such that, for all S in $\mathscr{E}$, $g_Q(S) = f_Q(S+P_1)$. Let us write the divisor of this new function. Since a zero S (resp. a pole) of g_Q is such that $S+P_1$ is a zero of f_Q (resp. a pole), we obtain:

$$\begin{array}{rcll} div(g_Q) & = & N_d(\sum_{i=0}^{d-1}(\pi^i(Q)-P_1) - d(-P_1)) & \\ & = & N_d(\sum_{i=0}^{d-1}(\pi^i(Q-P_1)) - d(-P_1)) & \text{since } P_1 \text{ is in } \mathscr{E}/\mathbb{F}_q \\ & = & div(f_{Q-P_1}) + N_d(d(\mathcal{O}) - d(-P_1)) & \\ & = & div(f_{Q-P_1}) - dN_d((-P_1) - (\mathcal{O})) & \end{array}$$

Hence on the other hand we have:

$$\begin{array}{rcl}
\Psi(\mathscr{D}_{Q-P_1}) & = & \Psi(div(f_{Q-P_1})) \\
& = & \Psi(div(g_Q) + dN_d((-P_1) - (\mathcal{O}))) \\
& = & \Psi(div(g_Q)) \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d} \\
& = & g_Q(F)^{1/N_d}\alpha^{-1} \cdot (\alpha/g_Q(\mathcal{O})^{1/N_d}) \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d} \\
& \underset{\text{mod } \mathbb{F}_q}{=} & g_Q(F)^{1/N_d}\alpha^{-1} \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d} \\
& \underset{\text{mod } \mathbb{F}_q}{=} & f_Q(F + P_1)^{1/N_d}\alpha^{-1} \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d} \\
& \underset{\text{mod } \mathbb{F}_q}{=} & \pi(\Psi(\mathscr{D}_Q)) \cdot \Psi((-P_1) - (\mathcal{O}))^{dN_d} \quad \text{from Equations (1) and (5).}
\end{array}$$

□

We emphasize that the green term is a constant term. Thanks to this action, we are able to reduce the size of the factor base by a factor k throughout the computations. Indeed, if we know the discrete logarithm of $\Psi(\mathscr{D}_Q)$ then we learn for free the discrete logarithms of $\Psi(\mathscr{D}_{Q-P_1}), \Psi(\mathscr{D}_{Q-P_2}), \cdots, \Psi(\mathscr{D}_{Q-P_{k-1}})$.

4.5. **Getting a smaller factor base.** To be able to reduce the size of the initial factor base, and thus to decrease the complexity of the linear algebra phase, we adapt the idea of systematic factors that was presented in [9] to the elliptic case. The idea was twofold: first extracting some systematic factors that appear in every equation, second, restrict the search to a sieving space that induce extra common factors. In this article, we choose to call these extra factors *compelled factors* to underline the difference with previous ones.

Left part of the diagram: making P_3 a compelled point. In our case, our aim is to consider a subgroup of the sieving space where A and B are polynomials such that the associated divisors always present a common (compelled) point. Here, choose to use the special point $P_3 = 3\,P_1$. As in [9] we select three generators g_1, g_2, g_3 in $\mathbb{F}_q[U, V]$ leading to divisors going through P_3. We propose to sieve on pairs of polynomials (A, B) such that $A = g_1 + \alpha g_3$ and $B = g_1 + \beta g_2 + \gamma g_3$ where $\alpha, \beta, \gamma \in \mathbb{F}_q$. Indeed, if A (resp. B) is a linear combination of those three generators and if P_3 is a zero of $\Phi^*(i(g_j))$ for $j = 1, 2$ and 3 then it is also a zero of the image of A (resp. B) in $\mathscr{E}$. As a consequence and for the same reason, P_3 is a zero of the image of $A - \alpha B$ too, where α belongs to the base field.

Lemma 3. *Let j be an integer in $[0, k-1]$ and assume that P_0 is a shorthand for $\mathcal{O}$. Then:*

$$\begin{array}{rl}
& \Xi(U - x_j) = (P_{j+1}) + (-P_{j-1}) - 2(P_1), \\
& \Xi(V - x_j) = (P_j) + (-P_j) - 2(\mathcal{O}), \\
\textit{and} \quad & \Xi(W - x_j) = (P_{j-1}) + (-P_{j+1}) - 2(-P_1).
\end{array}$$

Proof. Let j be an integer between 0 and $k-1$. We recall that x_j denotes the abscissa of $P_j \in \mathscr{E}$ and $x_0 = 0$. Then by definition:

$$\Phi(P_j) = [x_{j-1}, x_j, x_{j+1}]$$

for all possible values of j. It means that, over the curve $\mathscr{C}$, $\Phi(P_j)$ is a zero of $U - x_{j-1}$, $V - x_j$ and $W - x_{j+1}$. Going back to the curve $\mathscr{E}$ it yields that the point P_j is a zero of $\Phi^*(U - x_{j-1})$, $\Phi^*(V - x_j)$ and $\Phi^*(W - x_{j+1})$. Similarly, from:

$$\Phi(-P_j) = [x_{j+1}, x_j, x_{j-1}]$$

we get that $-P_j$ is a zero of $\Phi^*(U-x_{j+1})$, $\Phi^*(V-x_j)$ and $\Phi^*(W-x_{j-1})$. Besides, writing $\Phi^*(U-x_j)$ as $((Y+y_1)/(X-x_1))^2 - X - x_1 - x_j$ we see that P_1 is a pole of $\Phi^*(U-x_j)$ with multiplicity 2. Similarly $-P_1$ is a pole of $W-x_j$ with multiplicity 2. From $\Phi^*(V-x_j) = X - x_j$ we conclude that $\mathcal{O}$ is twice a pole too. □

Hence as generators we select:

$$\begin{aligned} & g_1 = U - x_2, \\ & g_2 = V - x_3, \\ \text{and} \quad & g_3 = (U-x_2)(V-x_3). \end{aligned} \tag{6}$$

From Lemma 3 we have $\Xi(g_1)) = (P_3) + (-P_1) - 2(P_1)$ and $\Xi(g_2)) = (P_3) + (-P_3) - 2(\mathcal{O})$. Thus $\Xi(g_3)) = 2(P_3) + (-P_1) + (-P_3) - 2(\mathcal{O}) - 2(P_1)$. Clearly the point P_3 is a positive point of each divisor.

Thus, if we start to sieve with a parameter t equal to 1, we obtain divisors on the left part of the diagram that have a height lower or equal to 4. Since P_3 is a positive point for all these divisors, we are left with divisors that have a height lower or equal to 3. We conclude with the definition of the reduced factor base:

$$\mathcal{F} = \{d \in Div(\mathscr{E}) \quad | \quad d \text{ is elementary and } h(d) \leq 3\}.$$

Remark 2. *The factor base is the same on both sides of the diagram.*

As previously, we can upper bound the cardinality of the factor base by the number of divisors with height lower than 3, so the number of monic degree-3 polynomials in $\mathbb{F}_q$, which is q^3. Thanks to the Frobenius action, the base is reduces by a factor of k, and the reduced factor base has size $O(q^3/k)$. At the end, assuming that we get enough equations (we discuss this issue in Section 4.6), linear algebra recovers the discrete logarithms of the initial factor base elements in $O((q^3/k)^2 q) = O(q^7/k^2)$ operations. When k is close to q, this matches the result of [9].

Remark 3. *Note that by contrast with polynomials, ideals of degree $4t$ are determined from $4t$ monomials, instead of $4t+1$. This is unfortunate because it forces us to increase the degrees to get enough degrees of freedom when generating relations. However, this drawback is counter-balanced by the reduction of the factor base size obtained by using the action of Frobenius.*

4.6. **How to get enough relations with the reduced factor base.** Thank to P_3 which is a compelled point we know that, on the left part of the diagram, we directly have divisors that split into factor base elements only. Now the questions are whereas we manage to easily obtain small height divisors on the right part or not, and how many relation we expect to write. We recall that, to be able to perform linear algebra, we need as many relations as unknowns.

We have around $q^3/3$ unknowns and we sieve on q^3 pairs of polynomials (A, B) consisting in linear combinations of g_1, g_2 and g_3 as given in (6). It means that we need a probability higher than $1/3$ to get a relation. Since a degree-d divisor is clearly linked with an irreducible polynomial of the same degree as seen in Section 2, this probability is assumed to be the same that a random polynomial of degree d to factor into terms of degree at most 3.

For degree $d = 8$, the probability is easy to compute. A polynomial fail to factor into terms of degree at most 3 when one factor as degree 4 or more. Since there can only be a single factor when the degree is 5, 6, 7 or 8 and at most two of degree 4. Thus, for large fields, the probability of success approaches $1 - (1/8 + 1/7 + 1/6 + 1/5 + 3/16 + 1/32) \approx 0.147$. Unfortunately, this is much smaller than $1/3$.

Right part of the diagram: the two compelled points P_2 and P_3. Thus we need to look at the right part of the diagram more carefully. Going back to the analysis made in Section 4.3, *Right part* we see that choosing A and B as monic (in some sense) does not reduce the height of the associated divisor. Hence, for (A, B) a pair of linear combinations of g_1, g_2 and g_3 as in (6), the divisor:

$$\Xi([A, B])$$

has still a height of 8. Because P_3 is a zero of $\Phi^*(A(U, V))$ and $\Phi^*(B(U, V))$, we note that P_3 is a zero of $\Phi^*([A, B])$ too. Yet it is not enough and we need to extract another compelled positive point of the associated divisor to the image of $A(V, W)$ (resp. $B(V, W)$) over $\mathscr{E}$. We start by underlining that g_1, g_2 and g_3 respectively becomes $V - x_2, W - x_3$ and $(V - x_2)(W - x_3)$, when sending V to W and U to V. Thus according to Lemma 3, the point $P_2 = 2\,P_2$ is a zero of all the generators, and so a zero of $[A, B]$ as P_3. We conclude that we are left with a divisor of height at most 6. The probability that it splits into a sum a divisors with height at most 3 is so roughly equals to:

$$1 - (1/6 + 1/5 + 1/4) \approx 0.383 > 1/3,$$

as q grows. As a consequence, we heuristically expect to get a linear system with enough equations to get the discrete logarithms of all elements of $\mathcal{F}$ in $O(q^5)$ operations.

Heuristic. *The heuristic in this linear algebra step and in all the following ones comes from the fact that we have no argument to prove we really get enough equations. We can count them and expect that when their number slightly exceeds the number of unknows, we are able to find a solution. Yet, nothing provably indicates whether the kernel of our matrix of relations has dimension* 1 *or not.*

5. Extended Factor Base and Individual Discrete Logarithm.

We only sketch here the last two main steps of our practical algorithm, the computation of an Extended Factor Base and the Individual Discrete Logarithm step. Indeed, they are an adaptation of the techniques that already exist for Frobenius representation algorithms to our setting.

5.1. From divisors of height 3 to divisors of height 4.

As done in Frobenius representation algorithms, we extend the initial factor base and now include all elementary divisors up to height 4. Thanks to the Frobenius action, there are approximately $q^4/4k \approx q^3/4$ unknowns. The naive approach we showed earlier gives the desired logarithms at a cost of $O(q^7)$ arithmetic operations (or $O(q^9/k^2)$ when k is away from q).

Practical speed up with regrouping. To speed up the computation of height 4 divisors, it is possible to decompose the height 4 factor base into small groups, in a way similar to [9], in order to perform several linear algebra steps on these small groups, instead of a single big linear algebra step. In Appendix B, we give details on how to produce relation in these groups. One technicality is the interaction of the groupings with the reduction of the factor base size given the action of Frobenius.

Once the height 4 divisors are obtained, it is a simple matter to continue extending the factor base to height 5 divisors. For that final step, no additional linear algebra is needed. It suffices to keep relations where a single height 5 divisors appears, the rest being of lower degree. See Appendix B for a detailed explanation.

However, for the height 4 extension, the expected number of produced relations seems to be slightly too low asymptotically to guarantee its success. Nevertheless, we tested the method on a practical example to check its viability. Namely, for the target finite field $\mathbb{F}_{3^{1345}}$, we were able to compute logarithms for an extended factor base comprising divisors of height up to 5. This was done by choosing a curve of (prime) cardinality 269 over $\mathbb{F}_{243}$. Studying the exact behavior of the height 4 extension to understand this gap is thus a matter of further research.

5.2. **Computing Individual Discrete Logarithms.** To solve the discrete logarithm problem in our target finite field, we need not only to know the logarithms of extended factor base elements but to be able to compute the discrete logarithm of any arbitrary element. This is the aim of this section. Various descent phases were previously proposed by various authors, the idea is to show how to adapt to our context. In practice, one can use the bilinear descent, the classical descent and the zig-zag descent or a mix of them. Indeed the quasi-polynomial descent of [3] is unlikely to be practical for currently accessible computations.

For the classical descent which simply consist in writing the target finite field element whose discrete logarithm is wanted as a product or quotient of relatively low-degree polynomials in θ, no adaptation is needed. We only need to check that any polynomial in $f(\theta)$ can be injected in the commutative diagram. This is simply done by written the divisor of $f(V)$ since $\Psi(V) = \theta$. When f is irreducible, the corresponding divisor is either the sum of two elementary divisors of height $\deg(f)$ or a single elementary divisor of height $2\deg(f)$.

We illustrate the adaptation with the bilinear and zigzag descents:

5.2.1. *Bilinear descent for our setting.* The bilinear descent step is easy to adapt from [9]. Remark that we usually need to unbalance the degrees of freedom in A and B, thus choosing different sets of generating polynomials. Instead of constructing the polynomials just from 1, U, V and UV we built them from higher degree polynomials in $\mathcal{M}_{t_a}$ and $\mathcal{M}_{t_b}$ respectively. We assume that $t_a \geq t_b$. Let us first analyze the case where we use all these monomials, remembering that there are $4t_a$ and $4t_b$ of them. As usual, we force A and B to be monic and remove the head monomial of B from A. All of the other coefficients are replaced by a corresponding formal unknown. Thus, the polynomial A contains $4t_a - 2$ unknowns. If $t_a \neq t_b$, B contains $4t_b - 1$ monomials. If $t_a = t_b$, we can remove an extra unknown from B. Furthermore, we know that the height of factors of the form $A - \alpha B$ is upper bounded by $4t_a$. We also know from Table 4.3.3 that the height of the bracket is at most $8t_a$.

If we want to adjust the values modulo 4 of the number of degrees of freedom, it is necessary to use compelled points. More precisely, we can force A and B to go through one, two or three compelled points. This reduces the degrees of freedom by the same amount on both sides. It also reduces height on the left by the same value and heights on the right by its double.

As in Frobenius representation algorithms, the coefficients of each monomials in $[A, B]$ are bilinear (or linear or constant) in the A and B unknowns. To force an elementary divisor of degree d to appear in $[A, B]$, it suffices to require that the bracket vanishes when evaluated at each of the d conjugate points corresponding to the associated prime divisor. This yields a bilinear system of d equations in the A and B unknowns. This equation can be solved using Grobner basis techniques exactly as in the case of Frobenius representation.

The only extra (and minor) restriction here is the relation between the number of A variables and B variables modulo 4.

5.3. **Zig-zag descent.** The zig-zag descent seems to be the best option to achieve provable quasi-polynomial complexity. In particular, it is used both in [12] and [15]. As a consequence, we also show how to adapt it to our setting. As it is more lenghty to describe than the bilinear descent, we assign a separate section to it.

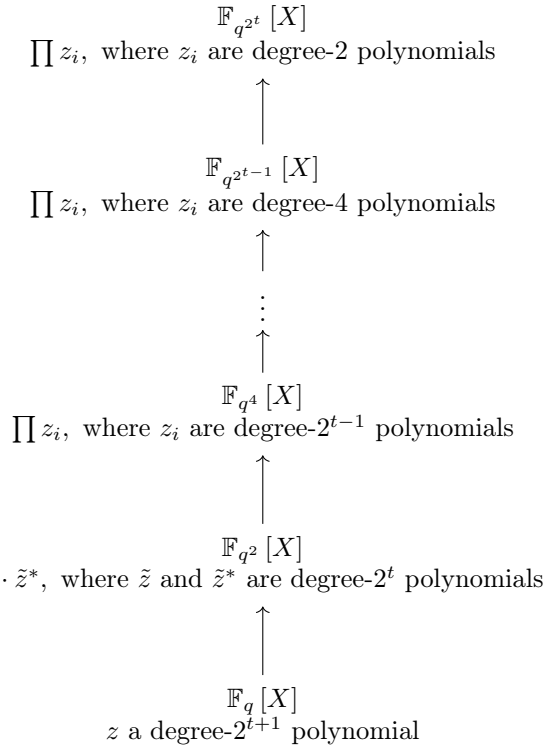

FIGURE 3. Tower of extensions over the base field $\mathbb{F}_q$ in the classical zig-zag descent.

Short recap on the zig-zag descent. First the main idea is to adapt the zig-zag descent presented in [7] to our setting. Let us give an insight of this descent in the classical settings. We call z our target, which is an irreducible polynomial in $\mathbb{F}_q[X]$ of degree [3] $2d = 2^{t+1}$. One crucial point of this method is that for any relation in $\mathbb{F}_{q^2}[X]$ implying degree-d polynomials, one can find a relation in the subfield $\mathbb{F}_q[X]$ at the price of having polynomials of degree twice as large. Thus, in order to make z appear in a polynomial relation of $\mathbb{F}_q[X]$, we write it as a product of two degree-d conjugated polynomials $\tilde{z}$ and $\tilde{z}^*$ over the extended field $\mathbb{F}_{q^2}[X]$ and we try to get one relation (in the extended field) involving one of this degree-d polynomials. Multiplying by the same conjugated relation we would obtain a relation (in the subfield) where z appears.

[3] Indeed, one can use Wan's theorem [23, Theo 5.1] to ensure that any field element is equivalent to an irreducible polynomial of degree a power of 2 only slightly larger than the extension degree k.

Recursively manipulating this trick on a tower of extensions as presented in Figure 3, we write in fact z as a product of conjugated degree-2 polynomials over $\mathbb{F}_{q^{2^t}}[X]$. Indeed, this descent method rests upon the existence of an extended field in which any degree-2 polynomials evaluated in θ can be written as product of linear evaluations in θ. Thus at the end, we get a relation of the form $z(\theta) = \prod_i L_i(\theta)$ where L_i are linear polynomials.

Elliptic zig-zag descent. To adapt the previous descent to our settings, a idealized method would be to exhibit a sufficiently large extension of the curve $\mathscr{C}$ in which any height-2 divisor can be written not as a degree-2 place but as a sum of points on this exact extension (and not the larger following one). This precisely would have translated the requirement that all degree-2 polynomials split in linear polynomials when the extension degree of the field is sufficiently large. Unfortunately, to the best of our knowledge, this ideal adaptation isn't possible.

On the technical side, we see that the method is much easier to describe when computing logarithm in $\mathbb{F}_{q^k}$ for an odd extension degree k. Indeed, in that case, the compositum of $\mathbb{F}_{q^k}$ and any extension $\mathbb{F}_{q^{2^i}}$ is simply $\mathbb{F}_{q^{k2^i}}$. Making this assumption is very convenient to describe the adaptation to the elliptic representation. We deal now with to the elliptic representation setting, with the additional restriction that the extension degree k is odd. Figure 4 illustrates our general technique.

5.3.1. *Points and divisors over extensions.* As mentioned in Section 3, the commutative diagram in Figure 2 can be used not only over $\mathbb{F}_q$ but also over extensions. We now give more details for $\mathbb{F}_{q^d}$, assuming that d and k are coprime.

This we now use polynomials A and B with coefficients in the larger field $\mathbb{F}_{q^d}$. Everything remains almost identical, except the definition and properties of the bracket. With a larger field, we use:

$$[A, B]_* = A^\pi(V, W)\, B(U, V) - A(U, V)\, B^\pi(V, W),$$

where A^π denotes the polynomial derived from A by raising each coefficient of A to the power q (while keeping the same monomials). This new bracket $[\cdot, \cdot]_*$ is $\mathbb{F}_q$-bilinear (but not $\mathbb{F}_{q^d}$-bilinear).

5.3.2. *Bootstrapping the descent.* Let $z \in \mathbb{F}_{q^k}$ be our target arbitrary element for which we want to find a discrete logarithm. Thanks to the diagram of Figure 2, we know that there exists a polynomial Pol in $\mathbb{F}_q[U, V]$ such that:

$$z = \Psi(\Xi(\mathrm{Pol}(U, V))).$$

In fact, there are many such polynomials. We choose ℓ such that $2^\ell > k$ and search a representation by a polynomial Pol in $\mathbb{F}_q[U, V]$ such that:

1. $z = \Psi(\Xi(\mathrm{Pol}(U, V)))$.
2. $h(\Xi(\mathrm{Pol}(U, V))) = 2^\ell$.
3. $\Xi(\mathrm{Pol}(U, V))$ exactly corresponds to a place of degree 2^ℓ.

Let us call p_z such a place in $\Sigma_{\mathbb{F}_q(\mathscr{C})}$. We could lift it to $\Sigma_{\mathbb{F}_{q^{2^\ell}}(\mathscr{C})}$ so that it corresponds to 2^ℓ points. However, for the rest of the method, it suffices to decompose it into degree-8 places. Theses places appear in $\Sigma_{\mathbb{F}_{q^{2^{\ell-3}}}(\mathscr{C})}$.

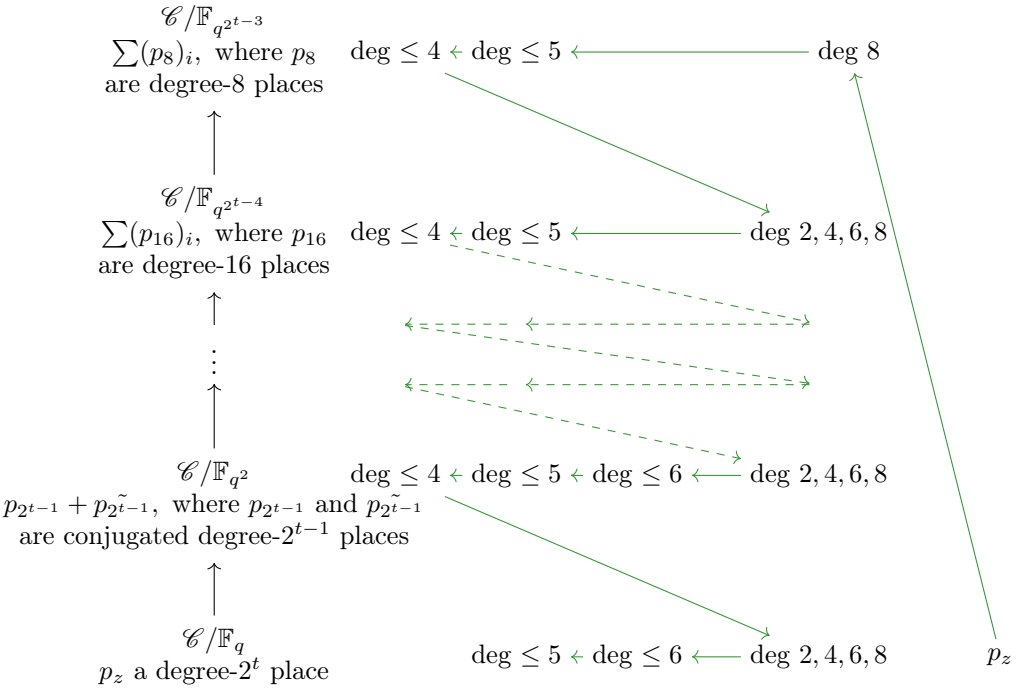

FIGURE 4. Tower of elliptic curves extensions in the elliptic zig-zag descent. The path in green represents how we decompose p_z in smaller degree places over higher extensions during the algorithm.

5.3.3. *Descending degree-*8 *places.* Using a series of relations based on the bracket $[A, B]_*$, there is a way to express the logarithm of the divisor corresponding to a degree-8 place as a sum of logarithms of divisors of degree at most 4. Once this is done, we can pair conjugates divisors and go one step down in the tower of quadratic extension. This at most doubles the height of divisors. Iterating the process, we now encounter places of degree 6 and 8 whose divisors need to be expressed as combination of divisors of degree at most 4. Finally, at the bottom of the tower, everything can be expressed using divisors of height at most 4, this in turn permit to compute the logarithm of z.

Keeping this strategy in mind we now describe the transformation of logarithm of divisors into sums of divisors of lower height. More precisely, we first transform degree 8 places as sums using divisors of height at most 6. Places of degree 6 can be

expressed using divisors of height up to 5. Finally place of degree 5 are transformed using divisors of height up to 4.

The exact degrees appearing in the descent strategy depend on the relative position in the tower of extension. Except at the lower levels, it is possible to descent directly from degree 8 to degree 5 and from degree 6 to degree 4. Except at the lower levels, it even possible to descend from degree 4 to degree 3.

Thus, from a practical point of view,there are two essentially equivalent options for the descent. Either one starts from a degree-8 place and encounters descent steps from 8 to 5 then 4 and descent steps from 6 to 4, except in the lower levels where longer chains from 8 to 6 then 5 and finally 4 appear. Or one starts from a degree 4 place and encounters descent steps from 6 to 4 then 3 and steps from 4 to 3. At the lower levels, this approach gets stuck.

In our presentation, we choose the approach that starts from a degree-8 place. Note than in the context of provable algorithms, using degree 8 possibly leads to a more difficult proof.

5.3.4. *Degree-5 places.* We start with degree 5 places since it is slightly simpler and illustrates the general idea. We let d be the power of 2 corresponding to our current position in the tower of extensions.

Again, we create somehow relations from:

$$\prod_{\alpha\in\mathbb{P}_1(\mathbb{F}_q)} (A-\alpha\, B) = [A,B]_*\,,$$

where (A,B) is a pair of polynomials with coefficients in $\mathbb{F}_{q^d}$. For degree 5, the polynomials are built from monomials in $\mathcal{M}_1$, i.e. from 1, U, V and UV. To check whether there are enough degrees of freedom to force a place too appear, we need to consider how many (non equivalent) pairs of candidates relations we can try.

Since we use the new bracket instead of $[\cdot,\cdot]$, the counting changes slightly. Before considering the property of the bracket, there is a total of 8 coefficients in $\mathbb{F}_{q^d}$, four in each of A and B. Remark that, for any $\Lambda\in\mathbb{F}_{q^d}$, we have $[\Lambda A,\Lambda B]_* = \Lambda^{q+1}\,[A,B]_*$. Simultaneously, the left side corresponding to $(\Lambda A,\Lambda B)$ is $\Lambda^{q+1}\prod_{\alpha\in\mathbb{P}_1(\mathbb{F}_q)}(A-\alpha B)$. Since Λ^{q+1} appears on both sides, we see that (A,B) and $(\Lambda A,\Lambda B)$ generate the same equation. Thus, we can set the leading coefficient of A to 1. This removes one of the coefficients.

In addition, because of the $\mathbb{F}_q$-linearity of the bracket, for any $\lambda\in\mathbb{F}_q$, we have $[A,B-\lambda A]_* = [A,B]_*$. Thus we can fix one component of the leading coefficient of B to 0. Then, using $[A,\lambda B]_* = \lambda\,[A,B]_*$ we can fix one component of another coefficient of B to 1. Finally, thanks to the relation $[A-\lambda B,B]_* = [A,B]_*$, we can set the corresponding component in A to 0.

This decreases the numbers of degrees of freedom to $7-3/d>5$, when $d>1$. Thus, we have enough degrees of freedom available. In the case $d=1$, we are in the base field where the logarithms of the degree-5 places have already been precomputed as part of the extended factor base.

Let p_5 be a place of degree 5 in $\Sigma_{\mathbb{F}_{q^d}(\mathscr{C})}$. Using a variation on bilinear descent and solving a bilinear system of equations in 6 unknowns over the extension field $\mathbb{F}_{q^d}$, we can obtained a relation involving p_5. Since the number of variables is a small constant, this is a very efficient computation.

The relation can be written in the form:

$$\sum_{D \in Div \,|\, h(d) \leq 4} D = (p_5) + D_3$$

where D_3 is a divisor of height 3. This shows that we can descend any divisors of height 5 as a sum of divisors of height at most 4. Note that D_3 and the divisors on the left may not be elementary, however, in that case they decompose into elementary divisors of lower degrees.

Note that we do not prove here that such a decomposition always exists. Instead out counting of the degrees of freedom gives heuristic support to this fact. It might be possible to adapt the proofs of [15] or [12] to our specific setting.

5.3.5. *Degree-6 places.* For degree 6, there are two options depending of the extension degree d.

When $d \geq 4$, we can again build relations using only the monomials 1, U, V and UV. In this case, it gives $7 - 3/d > 6$ degrees of freedom. Thus, we can directly descend to a sum of divisors of height at most 4.

For the remaining cases, $d = 2$ or $d = 1$, we need to use monomials from $\mathcal{M}_2$ to provide more degrees of freedom. However, if we use them all, the height of the left-hand factor become 8 and the height of the bracket is 16. To control this explosion, it suffices to fix three (essentially arbitrary) compelled points and keep a basis of all functions going through these 3 points. This basis contains 5 polynomials, say $g_1, \ldots, g_5$. Forming A and B as linear combinations of the g_is induces a systematic factor of total height 3 in every term $A - \alpha B$ (corresponding to the compelled points). Furthermore, this systematic factor also appears in the decomposition of the bracket together with an extra systematic factor also of height 3. This extra factor corresponds to the compelled points translated by $-P_1$. Thanks to the systematic factors, the height of the left becomes 5 while the height of the right becomes 10. There is a total of 10 coefficients in A and B, which corresponds to $9 - 3/d \geq 6$ degrees of freedom when removing identical relations as in the previous case. More precisely, we can fix the coefficient of g_1 in A to 1, one component of the coefficient of g_1 in B to 0, and one component of g_2 to 1 in B and 0 in A.

Solving a bilinear system, we can find coefficients that lead to an equation:

$$\sum_{D \in Div \,|\, h(d) \leq 5} D = (p_6) + D_4$$

where D_4 is a divisor of height 4. This expresses the logarithm p_6 as a sum of logarithms of divisors of height at most 5.

5.3.6. *Degree-8 places.* For degree 8 places, we proceed as in the second method for degree 6. We use monomials from $\mathcal{M}_2$. With three compelled points as in degree 6, we have $9 - 3/d$ degree of freedom. This is more than 8 as soon as $d \geq 4$. In this case, we can write the logarithm p_8 as a sum of logarithms of divisors of height at most 5.

When d is 1 or 2, we use only two compelled points. We thus have a basis of 6 polynomials and $9 - 3/d \geq 8$ degrees of freedom. The height after removing the systematic factors become 6 for the left factors and 12 for the bracket. Thus, in the lower levels of the tower of extension, we can write the logarithm p_8 as a sum of logarithms of divisors of height at most 6.

5.3.7. *Practical (un)efficiency of the approach.* In the Frobenius representation zigzag, every step down the tower was based on the creation of one relation. As a consequence, at every level, the total number of elements under consideration was multiplied by $O(q)$.

By contrast, here, we need two levels of relations for each of the middle levels of the tower. As a consequence, the total number at each level is multiplied by $O(q^2)$, which makes this approach much less appealing in practice.

6. **Practical discrete experiment over $\mathbb{F}_{3^{1345}}$.** To study the practicality of the elliptic representation in discrete logarithm computations, we have performed an experiment using $q = 3^5 = 243$. We did run our experiment on a server. The most time consuming steps where implemented in pure C, with calls to the magma software [2] for verification purposes and complex Gröbner basis computations.

We define $\mathbb{F}_{243}$ as $\mathbb{F}_3[a]$ where a is a root of the irreducible polynomial $x^5 - x + 1$. To define the full extension, we use the elliptic curve $\mathscr{E}$ with equation:

$$y^2 = x^3 + a\,x^2 + x + a^{35}.$$

This curve has $k = 269$ points over $\mathbb{F}_{243}$, thus allowing us to define the extension $\mathbb{F}_{243^{269}} = \mathbb{F}_{3^{1345}} = \mathbb{F}_{243}[v]$. Note that this k is a prime close to the top of the Hasse interval for a field with 3^5 elements. We also choose the point $P_1 = (1, a^{195})$ to describe the action of Frobenius.

The trivariate description, i.e. the curve $\mathscr{C}$ obtained from the Semaev polynomial is then defined by the three equations:

$$\begin{aligned}
U\,V + a^3\,U\,W + a^9\,U + V\,W + V + a^9\,W + a^{193} &= 0\\
U\,W^2 + a^{13}\,U\,W + a^{26}\,U + a^{239}\,V\,W^2 + a^{239}\,V\,W + a^{239}\,V + a^6\,W^2 + a^{59}\,W + a^{134} &= 0\\
V^2\,W^2 + V^2\,W + V^2 + V\,W^2 + a^{190}\,V\,W + a^{38}\,V + W^2 + a^{38}\,W + a^{110} &= 0
\end{aligned}$$

Adding the two extra Frobenius relations $V = U^q$ and $W = V^q$ we can use a Gröbner basis algorithm to find the minimal polynomial F_v of our finite field generator v.

For simplicity, we use the image of the point P_1 by the pairing to the field as a basis of the discrete logarithms in the subgroup of large order:

$$\frac{3^{1345} - 1}{648196409762}.$$

Of course, $648196409762 = 2 \times 11^2 \times 10223 \times 262007$ is small enough so that logarithms modulo this number can be computed by generic algorithms.

6.1. **First set of discrete logarithms.** In an initial step, we constructed equations from a pair of polynomials (A, B) of the form:

$$A(U,V) = (U - U(P_1))(V - V(P_1)) + C_A\,(U - U(P_1), B(U,V) = (V - V(P_1)) + D_B\,(U - U(P_1),$$

where C_A and D_B take all possible values in $\mathbb{F}_{243}$. Because of the compelled point P_1, every polynomial of the form $A - \alpha B$ factors into places of degree at most 3. Going through all the possible pairs took 90s on a single core of our server and created 22 575 equations in 17 887 variables corresponding to the places of degree 2 and 3.

This system of equations was then solved using Wiedeman algorithm, the standard four phases respectively took:

- **First matrix multiplication sequence.** 286 minutes on four cores,
- **Fast Berlekamp-Massey.** 14 minutes on a single core,

- **Second matrix multiplication sequence.** 146 minutes on four cores,
- **Final sum and verification.** 3 minutes on a single core.

The resulting 17 887 logarithms were tested by using magma to transfer places to the finite field through pairings and verifying the exponentiation.

6.2. **Discrete logarithms of degree-4 places.** For places of degree 4, we used the grouping technique described in Appendix B and worked with 244 groups. For a typical group[4], the cost of computations of the corresponding logarithms are distributed as follow:

- **Generating equations.** Under a minute on a single core, about 13 500 unknowns per group.
- **First matrix multiplication sequence.** 8 minutes on four cores,
- **Fast Berlekamp-Massey.** 10 minutes on a single core,
- **Second matrix multiplication sequence.** 5 minutes on four cores,
- **Final sum and verification.** Under a second.

In total, 3 253 919 logarithms were computed for degree 4 places.

6.3. **Discrete logarithms of some degree-5 places.** For place of degree 5, a fraction of the logarithms can be obtained by descending to degree 4. Unfortunately, this is not enough to get all of them. To be able to go the descent phase, we divided the places of degree 5 into 242×244 groups of approximately equal size. Each full set of 244 groups in turn permits a "descent" from degree 5 to degree 5 (inside the specific set of group) for a fraction of places. In total, 20 full sets of 244 groups were enough to enable the descent phase. The timings for each set of groups were as follow:

- **Generating equations.** About 750 minutes for most sets (less for the first three because of the cost of descent for later sets).
- **Linear algebra.** About 4 minutes on four cores, for a single group. I.e. roughly 1000 minutes for a set.
- **Final verification.** About 15 000 minutes to recompute all equations, check all logarithms and perform eventual corrections. The corrections are necessary when the initial (truncated) systems of equations do not have full rank. In that case, a small number of logarithms are possibly wrong and corrected in this step.

In total, 1 753 982 logarithms were computed for the first set of degree 5 places. Then, 754 224 for the second set and 323 074 for the third. Finally, the remaining 17 sets of roughly equal size (because of the choice to only use the first three for degree-5 to degree-5 descent here) containing altogether 2 319 092 logarithms.

6.4. **Descent.** To illustrate the descent, we start from a challenge generated from the expansion of the real constant Π. The exact formula is given in a magma verification script available on request, or given in Appendix C. Let us denote this challenge value by Z.

In a first step, we consider many elements of the form $Z_{i,j} = Z \cdot (u+1)^i \cdot (u+a)^j$. Since the logarithms of $v+1$ and $v+a$ are known, finding the logarithm of one of those is equivalent to finding the logarithm of Z. We consider many such elements to have a better chance to find a smooth expression of one of them.

[4]There are a couple of groups with slightly different performances but this does not affect the overall performance.

For each such element, we view it as a polynomial in v and using a continued fraction algorithm to find an expression of $Z_{i,j}$ as a quotient of two half-degree polynomials in v. We kept the $Z_{i,j}$ with the lowest possible degree obtained after factoring the numerator and denominator. More precisely, we do not consider the degree of the factorization into polynomials but into places of the curve $\mathscr{C}$.

We considered 240 values of j and 30 000 values of i for each j. We run 48 instances in parallel on our server for a wall-clock time of 4200 minutes and a CPU-time of approximately 140 days.

The best $Z_{i,j}$ contained places up to degree 36, it is given in the magma verification script.

By using the bilinear descent, we were able to compute the logarithms of places up to degree 18. As an illustration, we give in the magma file the logarithm of a degree 18 polynomial in v, which was computated as the sum of logarithms of two places of degree 18. Each of those two computations took about 4 days, using a single core[5]. We also computed the logarithms of lower degree places.

However, our decomposition still contains three polynomials of respective degree 25, 34 and 36 (each corresponding to two places of the same degree) for which we could not compute the logarithms. The reason for this aborted computation is that the Gröbner basis descent did not terminate for theses cases. We considered using the alternative approaches but did not find an option that would allow us to finish the computation without requiring excessive development and/or running times.

REFERENCES

[1] Leonard M. Adleman and Ming-Deh A. Huang. Function field sieve method for discrete logarithms over finite fields. *Inf. Comput.*, 151(1-2):5–16, 1999.

[2] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[3] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 1–16, 2014.

[4] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren, editors. *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Chapman and Hall/CRC, 2005.

[5] Jean Marc Couveignes and Reynald Lercier. Elliptic periods for finite fields. *Finite Fields and Their Applications*, 15(1):1–22, 2009.

[6] Faruk Gölo̧glu and Antoine Joux. A simplified approach to rigorous degree 2 elimination in discrete logarithm algorithms. *IACR Cryptology ePrint Archive*, page 430, 2018.

[7] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. On the powers of 2. *IACR Cryptology ePrint Archive*, 2014:300, 2014.

[8] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. On the discrete logarithm problem in finite fields of fixed characteristic. *Trans. Amer. Math. Soc.*, 270:3129–3145, 2018.

[9] Antoine Joux and Cécile Pierrot. Improving the polynomial time precomputation of frobenius representation discrete logarithm algorithms - simplified setting for small characteristic finite fields. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, pages 378–397, 2014.

[5]Since magma was not installed on our main server, the computation of Gröbner basis were delegated to a smaller server. However, during that time the main server was simply waiting for the result.

[10] Antoine Joux and Cécile Pierrot. Technical history of discrete logarithms in small characteristic finite fields - the road from subexponential to quasi-polynomial complexity. *Des. Codes Cryptogr.*, 78(1):73–85, 2016.
[11] Thorsten Kleinjung and Benjamin Wesolowski. A new perspective on the powers of two descent for discrete logarithms in finite fields. *IACR Cryptology ePrint Archive*, page 647, 2018.
[12] Thorsten Kleinjung and Benjamin Wesolowski. Discrete logarithms in quasi-polynomial time in finite fields of fixed characteristic. Cryptology ePrint Archive, Report 2019/751, 2019. https://eprint.iacr.org/2019/751.
[13] Thorsten Kleinjung and Benjamin Wesolowski. Discrete logarithms in quasi-polynomial time in finite fields of fixed characteristic. *Journal of the American Mathematical Society*, 2021.
[14] Guido Lido. Discrete logarithm over finite fields of small characteristic. Master's thesis, Universita di Pisa, September 2016. Available from https://etd.adm.unipi.it/t/etd-08312016-225452.
[15] Guido Lido. Discrete logarithm over finite fields of small characteristic. Unpublished (personal communication), 2019.
[16] Guido Lido. A provably quasi-polynomial algorithm for the discrete logarithm problem in finite fields of small characteristic, 2022.
[17] Giacomo Micheli. On the selection of polynomials for the dlp quasi-polynomial time algorithm for finite fields of small characteristic. *SIAM Journal on Applied Algebra and Geometry*, 3(2):256–265, 2019.
[18] Victor S. Miller. The Weil pairing, and its efficient calculation. *J. Cryptology*, 17(4):235–261, September 2004.
[19] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance (corresp.). *IEEE Transactions on Information Theory*, 24(1):106–110, 1978.
[20] Carl Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. In *Discrete algorithms and complexity*, pages 119–143, 1987.
[21] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Springer Publishing Company, Incorporated, 2nd edition, 2008.
[22] Emanuela Ughi. On the number of points of elliptic curves over a finite field and a problem of B. Segre. *European Journal of Combinatorics*, 4(3):263–270, 1983.
[23] Daqing Wan. Generators and irreducible polynomials over finite fields. *Mathematics of Computation*, 66:1195–1212, 1997.
[24] William C. Waterhouse. Abelian varieties over finite fields. *Annales scientifiques de l'Ecole Normale Supérieure*, 2(4):521–560, 1969.

Appendix A. **Notations and reminders about algebraic function fields and divisors.** Let $\mathbb{K} = \mathbb{F}_q$ denote a finite field. Let $\mathscr{C}$ be a non-singular curve in the n-dimensional projective space $\mathbb{P}_n(\overline{\mathbb{F}}_q)$ defined over $\mathbb{K}$ and π denote the Frobenius map on $\mathbb{P}_n(\overline{\mathbb{F}}_q)$. We let $\mathbb{K}(\mathscr{C})$ denote the function field of $\mathscr{C}$ over $\mathbb{K}$. More details can be found in [4, 21] if needed. A *discrete valuation* on $\mathbb{K}(\mathscr{C})$ is a map υ from $\mathbb{K}(\mathscr{C})$ to $\mathbb{Z}$ such that for all $x, y \in \mathbb{K}(\mathscr{C})$ we have:

1. $\upsilon(xy) = \upsilon(x)\,\upsilon(y)$;
2. $\upsilon(x+y) \geq \min(\upsilon(x), \upsilon(y))$;
3. $\upsilon(x+y) = \min(\upsilon(x), \upsilon(y))$ when $\upsilon(x) \neq \upsilon(y)$.

We define an equivalence relation between valuations by saying that two valuations υ and υ' are equivalent whenever there exists a non zero rational constant α such that for all $x \in \mathbb{K}(\mathscr{C})$, we have $\upsilon'(x) = \alpha\,\upsilon(x)$. We recall that a *place* of $\mathbb{K}(\mathscr{C})$ is an equivalence class of discrete valuations of $\mathbb{K}(\mathscr{C})$ which are trivial on $\mathbb{K}$. The set of places of $\mathbb{K}(\mathscr{C})$ is denoted by $\Sigma_{\mathbb{K}(\mathscr{C})}$. In every place $\mathfrak{p}$, there exists a unique valuation whose value group is $\mathbb{Z}$, it is called the *normalized valuation* of $\mathfrak{p}$ and denoted $\upsilon_{\mathfrak{p}}$.

We recall that, for a non-singular curve $\mathscr{C}$, there is a one-to-one correspondance between places of $\mathbb{K}(\mathscr{C})$ and Galois orbit of points on $\mathscr{C}$. The *degree* of a place $\mathfrak{p}$ is the number of points in the corresponding orbit, we denote it by $\mathrm{Deg}(\mathfrak{p})$.

The *divisor group* $\mathrm{Div}(\mathscr{C})$ of $\mathscr{C}$ (over $\mathbb{K}$) is defined as the free abelian group over $\Sigma_{\mathbb{K}(\mathscr{C})}$. An element D of $\mathrm{Div}(\mathscr{C})$ is expressed as:

$$D = \sum_{\mathfrak{p} \in \Sigma_{\mathbb{K}(\mathscr{C})}} n_{\mathfrak{p}}(\mathfrak{p}),$$

where each $n_{\mathfrak{p}} \in \mathbb{Z}$ and $n_{\mathfrak{p}} = 0$ for all but finitely many places $\mathfrak{p}$. Since each place corresponds to a Galois orbit of points, a divisor D can also be given in the alternative form:

$$D = \sum_{P \in \mathscr{C}/\overline{\mathbb{F}}_q} n_P(P),$$

where each $n_P \in \mathbb{Z}$, $n_P = 0$ for all but finitely many points and $n_P = n_Q$ if P and Q belong to the same Galois orbit of points. A divisor D is said to be *prime* when $D = (\mathfrak{p})$ for a place $\mathfrak{p} \in \Sigma_{\mathbb{K}(\mathscr{C})}$.

The *degree* of a divisor D is defined as:

$$\mathrm{Deg}(D) = \sum_{\mathfrak{p} \in \Sigma_{\mathbb{K}(\mathscr{C})}} n_{\mathfrak{p}} \mathrm{Deg}(\mathfrak{p}) = \sum_{P \in \mathscr{C}/\overline{\mathbb{F}}_q} n_P.$$

In this paper, a degree-0 divisor that is the difference between a prime divisor and the right number of times the point at infinity $\mathcal{O}$ is defined as an *elementary divisor*. In particular, any elementary divisor associated to a point $Q \in \mathscr{C}/\mathbb{F}_{q^d}$ is a divisor of the form:

$$\sum_{i=0}^{d-1} \pi^i(Q) - d(\mathcal{O}).$$

A divisor D is called *effective* when $n_{\mathfrak{p}} \geq 0$ for all $\mathfrak{p}$. Any divisor D can be uniquely written as a difference of two effective divisors in the form $D = D_0 - D_\infty$, where:

$$D_0 = \sum_{\substack{\mathfrak{p} \in \Sigma_{\mathbb{K}(\mathscr{C})} \\ n_{\mathfrak{p}} \geq 0}} n_{\mathfrak{p}}(\mathfrak{p}) \quad \text{and} \quad D_\infty = \sum_{\substack{\mathfrak{p} \in \Sigma_{\mathbb{K}(\mathscr{C})} \\ n_{\mathfrak{p}} < 0}} -n_{\mathfrak{p}}(\mathfrak{p}).$$

The degree map from $\mathrm{Div}(\mathscr{C})$ to $\mathbb{Z}$ is a group morphism. Its kernel is denoted $\mathrm{Div}_0(\mathscr{C})$ and called the group of degree-0 divisors of $\mathscr{C}$, it is a subgroup of $\mathrm{Div}(\mathscr{C})$.

We define the map div that sends an element $f \in \mathbb{K}(\mathscr{C})^*$ to a divisor in the following way:

$$\begin{array}{rccc} div: & \mathbb{K}(\mathscr{C})^* & \mapsto & \mathrm{Div}(\mathscr{C}) \\ & f & \mapsto & div(f) = \displaystyle\sum_{\mathfrak{p}\in\Sigma_{\mathbb{K}(\mathscr{C})}} \upsilon_{\mathfrak{p}}(f)\,\mathfrak{p}. \end{array}$$

A divisor associated to a function in the above way is called a *principal divisor*. The image of div, i.e. the set of all principal divisors, is denoted $\mathrm{Princ}(\mathscr{C})$. All principal divisors have degree 0 and $\mathrm{Princ}(\mathscr{C})$ is a subgroup of $\mathrm{Div}_0(\mathscr{C})$. Every principal divisor can also be written as a difference of effective divisors as:

$$div(f) = div(f)_0 - div(f)_\infty.$$

The places (or points) that occur in $div(f)_0$ or $div(f)_\infty$ are respectively called the *zeroes* or *poles* of f. Note that for two functions f and g of $\mathbb{K}(\mathscr{C})^*$, we have $div(f) = div(g)$ if and only if there exists an element $\alpha \in \mathbb{K}^*$ such that $g = \alpha\, f$.

Since $\mathrm{Princ}(\mathscr{C})$ is a subgroup of $\mathrm{Div}_0(\mathscr{C})$, we can form the quotient group, which is called the *Picard group* (or divisor class group) of $\mathscr{C}$ and denoted $\mathrm{Pic}_0(\mathscr{C})$. Two divisors have the same representative in the Picard group if and only if their difference is principal.

Appendix B. **Relations for factor base extension.** Let us describe our decomposition in groups to extend the factor base to all elementary divisors of height equal or lower than 4. The idea is to write a partition of q groups with q^2 elements in each and to be able to decrease the height of the divisor associated to the bracket on the right again. To illustrate the process, we define a first group with the monomials:

$$\begin{aligned} g_1 &= UV \\ g_2 &= U + V \\ g_3 &= 1. \end{aligned}$$

Defining then G as all the linear combinations of these three monomials with coefficients in $\mathbb{F}_q$ permits to set our first (special) group as:

$$\mathcal{G} = \{div(\Psi^*(g)) \,|\, g \in G\}.$$

All the divisors in the special group have height lower than 4. We now sieve on pairs of polynomials (A, B) such that $A = g_1 + \alpha g_2$ and $B = g_1 + \beta g_3$ where $\alpha, \beta \in \mathbb{F}_q$. On the left side it is clear that all polynomials raised in the product belong to G. So all the divisors in the corresponding sum on the left side have height lower than 4 (see Table 4.3.1) and belong to $\mathcal{G}$. On the right side, we are left with a bracket $[A, B]$ leading to a height lower than 8. Again, the probability that it splits into divisors with a height lower than 3 is too low. Yet, computing the brackets:

$$\begin{aligned} [g_1, g_2] &= VW(U + V) - (V + W)UV = V^2(W - U) \\ [g_1, g_3] &= V(W - U) \\ [g_2, g_3] &= W - U \end{aligned}$$

and thanks to bilinearity we obtain that $W-U$ is a common factor of $[A, B]$. Besides we note that $h(\Xi(W - U)) = 4$.

Removing this constant contribution, we are left with a residual height of 4 on the right side. If it decomposes into lower height divisors, this gives us a linear equation involving the logarithms a subset of the divisors in $\mathcal{G}$. With enough such

equations, we again use linear algebra to compute the logarithms of the elements of $\mathcal{G}$.

Note that the probability to find a good relation is $3/4$ when q grows. We thus expect $3q^2/4$ equations in $q^2/4$.

Note that the pairs (A, B) that fail to give an equation are nonetheless useful! Indeed, a pair (A, B) of sieving polynomials fails if $[A, B]$ leads to a divisor with height precisely 4. It means that after obtain the logarithm of elements of $\mathcal{G}$ we can derive the logarithm of these extra divisors for free.

B.1. **Construction of groups with one compelled point.** Following the idea of the special group $\mathcal{G}$, we would like to construct small other groups of divisors that have two properties. First, for each group, we need to be able to create relations involving only heigh-4 divisors from this group on the left, possibly with divisors of lower height. Second, we need to control the splitting probability of the bracket on the right of the equation. We proceed using compelled points.

How to choose our generators g_1, g_2 and g_3 in this case ? We recall that the naive height-4 sieving is based on the monomials $1, U, V, UV$. Since there is no reason to favor nor U neither V, we propose to preserve symmetry between the two variables, writing:

$$\begin{aligned} g_1 &= UV + k_1 U \\ g_2 &= UV + k_2 V \\ g_3 &= 1. \end{aligned}$$

where k_1 and k_2 are in the base field $\mathbb{F}_q$. Defining again groups:

$$\mathcal{G}_{k_1,k_2} = \{\Xi(g_1 + \alpha g_2 + \beta g_3) \,|\, \alpha, \beta \in \mathbb{F}_q\}$$

with q^2 divisors each, we sieve on pairs of polynomials (A, B) such that $A = g_1 + \alpha g_2$ and $B = g_1 + \beta g_3$ where $\alpha, \beta \in \mathbb{F}_q$. On the left side all divisors have height lower than 4 and belong to $\mathcal{G}_{k_1,k_2}$. On the right side, we are left with a bracket $[A, B]$ leading to a height lower than 8. To decrease this height we consider the brackets:

$$\begin{aligned} [g_1, g_2] &= k_1 [U, UV] + k_2 [UV, V] + k_1 k_2 [U, V] \\ [g_1, g_3] &= VW + k_1 V - UV + k_1 U \\ [g_2, g_3] &= VW + k_2 W - UV + k_2 V \end{aligned}$$

Note that $[A, B]$ is a linear combination of these brackets and that the last two ones are associated to divisors of height lower than 4. Thus, removing a point in $div(\Phi^*([g_1, g_2]))$ will suffice. Let us look at $k_1 [U, UV] + k_2 [UV, V] + k_1 k_2 [U, V]$ in details. Calling c_f the coefficient in $\mathbb{F}_q$ of the leading monomial[6] of $[g_1, g_2]$ for the denominator of any fraction f, we see that we can force $k_1 c_{[U,UV]} + k_2 c_{[UV,V]} + k_1 k_2 c_{[U,V]} = 0$ in $\mathbb{F}_q$. We underline that for any fixed constant $k_1 \neq -c_{[UV,V]} c_{[U,V]}^{-1}$ there exists a unique k_2 such that the previous equality is verified. It means that we create $q - 1$ such groups. Besides, this annihilates the leading monomial so decreases the weighted degree of $[g_1, g_2]$ and leads to remove a point in the associated divisor of $[A, B]$. Hence, we are left with a residual height of 7. We want the corresponding divisor to be written as a sum of divisors of height 3 at most. The heuristic probability to get a good relation is so equal to:

$$1 - (1/7 + 1/6 + 1/5 + 1/4) \approx 0.2405$$

as q tends to infinity.

[6] Considering the weighted degree in X and Y of each monomial.

This is slightly too low for the purpose. As a consequence, we need either to improve the group construction or to make good use of the equations with a single height-4 divisor in the bracket. Nevertheless, since 0.24 is close to 1/4, it is conceivable that we can find enough relations in practice. We decided to test it and we computed all the discrete logarithms up to height-5 for the target finite field $\mathbb{F}_{3^{1345}} = \mathbb{F}_{243^{269}}$ with this method.

Remark 4. *It is useful to know that:*

$$\begin{aligned} [U, UV] &= UV(V - W) \\ [UV, V] &= VW(V - W) \\ [U, V] &= V^2 - UW \end{aligned}$$

B.2. **Interaction with the action of Frobenius.** Looking at our groupings, we see that we have built a total of q different ones (including the special group $\mathcal{G}$. Since each grouping contains about $q^2/4$, the computations (if successful) gives us about $q^3/4$ logarithms of height 4. This is much less than the total expected number which is close to $q^4/4$. However, the action of the Frobenius potentially multiply these logarithms by a factor of k. For practical, we heuristically assume that this is the case. The fact that we were able to compute the logarithms of all height-4 divisors for $\mathbb{F}_{3^{1345}}$ supports this assumption.

B.3. **Going to height** 5**.** We continue the extension to height 5 in similar fashion. Since, this requires more degree of freedom, we no longer need to use compelled points. Instead, we sieve on more general polynomials of the forms $A = UV + a_U U + a_1$ and $B = UV + b_V V + b_1$. On the left-hand side, all factors of the form $A - \alpha B$ have height 4. Thus they decompose in divisors of height at most 4 and their logarithms can be directly obtained. On the right-hand side, the bracket has height at most 8. We expect that it contains an elementary divisor of height 5 with probability close to 1/5. As a consequence, we obtain about $q^4/5$ divisors of height 5 without performing any linear algebra.

Again, thanks to the action of Frobenius, we expect to recover an overwhelming fraction of divisors of height 5. This turn out to work for our example $\mathbb{F}_{3^{1345}}$.

Appendix C. **Verification of discrete logarithms for the target field $\mathbb{F}_{3^{1345}}$.** Here is a Magma code that constructs the target finite field of our example thanks to an elliptic curve and gives some discrete logarithms.

```
gf3pol<x>:=PolynomialRing(GF(3));
gfext<a>:=ext<GF(3)|x^5-x+1>;
gf2pol<x2>:=PolynomialRing(gfext);
q:=3^5;

A:=a;B:=gfext!1; C:=a^35;
E:=EllipticCurve([0,A,0,B,C]);
P1:=E![1,a^195];

k:=#E;

cof:=648196409762;
M:=((q^k)-1) div cof;

```

```
15    R1<U,W,V>:=PolynomialRing(gfext,3);
16
17    // Use Semaev Polynomials to create C
18    // U, V, W are abcissae of 3 points P_U, P_V=P_U+P1 and P_W=P_V+P1
19    // First equation S3(U,V,x_P1)=0
20    S1:=P1[1]+U+V;
21    S2:=P1[1]*(U+V)+U*V;
22    S3:=P1[1]*U*V;
23    f1 :=(-(S1+A)*(S3+C)+(S2-B)^2);
24
25    // Second equation S3(V,W,x_P1)=0
26    shiftvar:=hom<R1->R1|V,0,W>;
27    f2:=shiftvar(f1);
28
29    // Third equation S3(U,W,x_(2*P1))=0
30    P2:=2*P1;
31    Sb1:=P2[1]+U+W;
32    Sb2:=P2[1]*(U+W)+U*W;
33    Sb3:=P2[1]*U*W;
34    f3 :=(-(Sb1+A)*(Sb3+C)+(Sb2-B)^2);
35    // Remove degenerate sub-varieties to get C
36    IdealC:=RadicalDecomposition(Ideal([f1,f2,f3]))[1];
37
38    // Add Frobenius constraint to get field definition (in V because of
          the chosen ordering of the polynomial ring)
39    FieldDef:=GroebnerBasis(Basis(IdealC) cat [V-U^q,W-V^q])[3];
40    BigField<v>:=ext<gfext|UnivariatePolynomial(FieldDef)>;
41    Eext:=BaseChange(E,BigField);
42    bool,Q:=IsPoint(Eext,v);
43    Q2:=Q+Eext!P1;
44    if Q2[1] ne Q[1]^q then
45      Q:=-Q;
46    end if;
47
48    u:=(Q-Eext!P1)[1];
49    w:=(Q+Eext!P1)[1];
50
51    // Check the Frobenius relations
52    assert u^q eq v;
53    assert v^q eq w;
54
55    // Mappings points to fields [can be extended to places, see paper]
56    FF<XX,YY>:=FunctionField(E);
57    HH:=hom<FF->BigField|Q[1],Q[2]>;
58
59       MyTatePairing:= function (P, ord, cof)
60       prod:=BigField!1;
61       pow:=BigField!1;
62       ZeroPt:=Zero(E);
63       curPsum:=ZeroPt;
64       curPdoub:=P;
65       e:=ord;
66       while (e ne 0) do
67         if  (e mod 2) eq 1 then
68           tmpD:=Divisor(curPdoub)+Divisor(curPsum)-Divisor(curPdoub+
           curPsum)-Divisor(ZeroPt);
69           bool,fn:=IsPrincipal(tmpD);
70           prod:=pow*prod*HH(fn);
71           curPsum:=curPdoub+curPsum;
```

```
72          end if;
73          tmpD:=2*Divisor(curPdoub)-Divisor(2*curPdoub)-Divisor(ZeroPt);
74          bool,fn:=IsPrincipal(tmpD);
75          pow:=pow^2*HH(fn);
76          curPdoub:=curPdoub+curPdoub;
77          e:=e div 2;
78        end while;
79        return prod^cof;
80      end function;
81
82
83      P0:=-P1;
84
85      // Sanity check:  Frobenius correctness after mapping to BigField
86      L0:=MyTatePairing(P0, k, cof);
87      Ltest:=MyTatePairing(14*P0, k, cof);
88      assert Ltest eq L0^(&+[q^i:i in [0..13]]);
89
90
91      // Challenge Log from Pi
92
93      R:=RealField(2000);
94      pival:=Pi(R);
95
96      Z:=&+[(Floor(pival*3^(val+1)) mod 3)*v^(val div 5)*a^(val mod 5): val
            in [0..1344]];
97
98      val:=(Z*(v+1)^9586*(v+a)^134);
99
```

```
100   tmp1:=(1)*v^134+(-1+a^3)*v^133+(a-a^4)*v^132+(-1-a+a^2+a^4)*v^131+(a
      ^3-a^4)*v^130+(-a-a^3-a^4)*v^129+(a+a^2+a^3-a^4)*v^128+(a-a^3+a^4)
      *v^127+(-1+a^2)*v^126+(-1-a^2+a^3)*v^125+(1-a^2-a^3-a^4)*v^124+(-a
      -a^3-a^4)*v^123+(-a+a^2+a^3-a^4)*v^122+(-a-a^2+a^3)*v^121+(a-a^2+a
      ^3-a^4)*v^120+(-1+a-a^2-a^4)*v^119+(a^2+a^3+a^4)*v^118+(-a-a^2-a
      ^4)*v^117+(-1-a^2-a^3-a^4)*v^116+(-a^2+a^3)*v^115+(a-a^2+a^3-a^4)*
      v^114+(1+a-a^2+a^3)*v^113+(-1-a+a^3)*v^112+(1+a+a^2+a^3+a^4)*v
      ^111+(a-a^3)*v^110+(-1-a^2-a^3+a^4)*v^109+(a^2-a^3+a^4)*v^108+(1-a
      -a^2+a^3)*v^107+(1-a+a^2+a^3)*v^106+(a^2-a^3)*v^105+(-1+a^2+a^3-a
      ^4)*v^104+(-a-a^4)*v^103+(1+a+a^3-a^4)*v^102+(-1+a-a^2+a^3+a^4)*v
      ^101+(a+a^3+a^4)*v^100+(-1+a+a^2-a^3+a^4)*v^99+(1-a+a^2-a^3)*v
      ^98+(1+a^3-a^4)*v^97+(-1+a+a^4)*v^96+(-1-a-a^2-a^3)*v^95+(-1+a-a
      ^2+a^3+a^4)*v^94+(a^2+a^4)*v^93+(-1+a-a^2-a^3+a^4)*v^92+(-a-a^2-a
      ^3-a^4)*v^91+(-a+a^2-a^3+a^4)*v^90+(a^3)*v^89+(-1-a^2)*v^88+(a-a
      ^2+a^3+a^4)*v^87+(-1-a+a^2)*v^86+(1+a^2+a^3-a^4)*v^85+(1+a+a^2+a
      ^3-a^4)*v^84+(a-a^2+a^4)*v^83+(-1-a^4)*v^82+(a^3)*v^81+(-1-a-a^3)*
      v^80+(-1+a+a^2-a^4)*v^79+(a^4)*v^78+(1-a+a^2+a^3-a^4)*v^77+(1+a+a
      ^2+a^3-a^4)*v^76+(-1-a+a^2-a^3-a^4)*v^75+(-1+a^3+a^4)*v^74+(1-a^2-
      a^3+a^4)*v^73+(1+a-a^3+a^4)*v^72+(a-a^3)*v^71+(1+a^3+a^4)*v^70+(a-
      a^2+a^3-a^4)*v^69+(-1+a-a^2-a^4)*v^68+(1+a-a^4)*v^67+(-a^2+a^4)*v
      ^66+(1+a+a^2-a^3+a^4)*v^65+(a-a^2+a^4)*v^64+(-1-a+a^2+a^3-a^4)*v
      ^63+(-1+a^2+a^4)*v^62+(1-a+a^3-a^4)*v^61+(-1+a+a^2-a^3+a^4)*v
      ^60+(1-a+a^3-a^4)*v^59+(-1+a-a^2-a^3+a^4)*v^58+(-1-a+a^3+a^4)*v
      ^57+(-a^2-a^4)*v^56+(a^2-a^4)*v^55+(1)*v^54+(-1+a^2-a^3+a^4)*v
      ^53+(a+a^2-a^3-a^4)*v^52+(a+a^2-a^3-a^4)*v^51+(a^3-a^4)*v^50+(1-a
      ^4)*v^49+(1-a-a^2+a^3)*v^48+(1+a+a^3)*v^47+(-1+a^2-a^3+a^4)*v
      ^46+(1-a^2+a^3)*v^45+(1-a+a^2+a^3+a^4)*v^44+(a-a^2)*v^43+(1+a^2+a
      ^3)*v^42+(-1+a-a^2+a^4)*v^41+(1+a^2-a^3+a^4)*v^40+(-1+a^2+a^3)*v
      ^39+(1-a-a^2+a^3+a^4)*v^38+(-1+a^2-a^3)*v^37+(-a-a^3-a^4)*v^36+(a
      ^2+a^4)*v^35+(a^3-a^4)*v^34+(1+a+a^3-a^4)*v^33+(-a^2)*v^32+(1-a+a
      ^2+a^3+a^4)*v^31+(1+a-a^3-a^4)*v^30+(-1-a-a^3)*v^29+(1-a^2-a^3-a
      ^4)*v^28+(-1+a^3)*v^27+(-1+a^2-a^3)*v^26+(1-a+a^2-a^3-a^4)*v
      ^25+(-1-a-a^2)*v^24+(-1+a-a^3+a^4)*v^23+(1+a^2-a^3)*v^22+(-1+a-a
      ^2-a^3+a^4)*v^21+(-1+a-a^3)*v^20+(-a-a^3)*v^19+(1+a^3+a^4)*v
      ^18+(1+a-a^2+a^4)*v^17+(a-a^2-a^3)*v^16+(1)*v^15+(-a^2+a^3)*v
      ^14+(1+a-a^3)*v^13+(1-a+a^2+a^4)*v^12+(-1-a^3)*v^11+(-1-a-a^2+a^3-
      a^4)*v^10+(-a^4)*v^9+(1+a)*v^8+(-1+a^2)*v^7+(-1+a+a^2+a^3+a^4)*v
      ^6+(1-a-a^2-a^3)*v^5+(a-a^3-a^4)*v^4+(-a+a^2+a^3-a^4)*v^3+(1-a^2-a
      ^3)*v^2+(-1+a+a^2-a^3-a^4)*v+(-1);
101
```

```
102   tmp2:=(1)*v^134+(-a-a^3)*v^133+(1+a^3+a^4)*v^132+(1-a+a^2-a^4)*v
          ^131+(1+a-a^4)*v^130+(1-a+a^2+a^3)*v^129+(-1+a-a^2-a^3)*v^128+(-1+
          a-a^2-a^3+a^4)*v^127+(a^2)*v^126+(-1+a-a^2)*v^125+(-a-a^3)*v^124+(
          a^2-a^3-a^4)*v^123+(1+a^2+a^4)*v^122+(-1-a^2+a^4)*v^121+(-a-a^2-a
          ^4)*v^120+(-1-a^2-a^3)*v^119+(a+a^3)*v^118+(1+a+a^2+a^4)*v^117+(1+
          a+a^2+a^3+a^4)*v^116+(1-a+a^2-a^3-a^4)*v^115+(-1-a-a^3+a^4)*v
          ^114+(1-a+a^2+a^3+a^4)*v^113+(-1+a+a^2+a^3+a^4)*v^112+(a-a^3)*v
          ^111+(a+a^2+a^3-a^4)*v^110+(-a+a^2+a^3)*v^109+(1-a-a^2)*v^108+(-a)
          *v^107+(-1+a-a^3-a^4)*v^106+(1+a^3+a^4)*v^105+(-1+a^2)*v^104+(-1+a
          ^4)*v^103+(a+a^3)*v^102+(-a+a^3+a^4)*v^101+(-1-a-a^4)*v^100+(-1+a+
          a^3+a^4)*v^99+(-1-a^3+a^4)*v^98+(1-a+a^3-a^4)*v^97+(1-a-a^2+a^4)*v
          ^96+(-1+a-a^2-a^3-a^4)*v^95+(a^4)*v^94+(1-a^2-a^4)*v^93+(a)*v
          ^92+(-1+a-a^2+a^3+a^4)*v^91+(-1-a+a^2+a^3+a^4)*v^90+(-1-a^3)*v
          ^89+(-1-a-a^3)*v^88+(1+a-a^2+a^3+a^4)*v^87+(1+a^2)*v^86+(1-a^2-a
          ^3-a^4)*v^85+(a-a^2+a^3+a^4)*v^84+(1-a-a^2-a^4)*v^83+(-1-a+a^2-a
          ^3+a^4)*v^82+(-a)*v^81+(1-a+a^3)*v^80+(-1+a-a^2)*v^79+(a-a^2+a^3)*
          v^78+(-1+a^2-a^3+a^4)*v^77+(0)*v^76+(1-a-a^2)*v^75+(1-a+a^2+a^3)*v
          ^74+(-1-a-a^2-a^3+a^4)*v^73+(1-a-a^2+a^4)*v^72+(a-a^2+a^4)*v^71+(-
          a+a^2+a^4)*v^70+(-a-a^2-a^3+a^4)*v^69+(1-a^2-a^4)*v^68+(-1-a-a^4)*
          v^67+(-a^2-a^3+a^4)*v^66+(1-a-a^3-a^4)*v^65+(-1+a+a^2-a^3-a^4)*v
          ^64+(1+a-a^2-a^3)*v^63+(-a-a^4)*v^62+(-1-a-a^3-a^4)*v^61+(-1+a-a
          ^2+a^3-a^4)*v^60+(1-a-a^2+a^4)*v^59+(-1+a-a^4)*v^58+(1+a-a^2+a^3)*
          v^57+(1-a^2)*v^56+(1-a-a^3+a^4)*v^55+(1+a)*v^54+(-a+a^3+a^4)*v
          ^53+(-1-a^2-a^3+a^4)*v^52+(-a-a^2+a^3)*v^51+(1+a+a^3+a^4)*v
          ^50+(-1-a-a^2-a^4)*v^49+(a-a^3-a^4)*v^48+(-a+a^2)*v^47+(-a+a^2+a
          ^4)*v^46+(1-a-a^2)*v^45+(1+a^3)*v^44+(-a^2+a^3)*v^43+(a+a^3)*v
          ^42+(-a+a^4)*v^41+(1-a-a^2-a^3+a^4)*v^40+(a-a^2-a^3)*v^39+(1+a+a
          ^2+a^3)*v^38+(1+a-a^2+a^3+a^4)*v^37+(-1-a^2+a^3+a^4)*v^36+(-1-a+a
          ^2-a^4)*v^35+(-1-a-a^2+a^3-a^4)*v^34+(1-a^2)*v^33+(a-a^2)*v^32+(1+
          a^2-a^4)*v^31+(-1-a^2-a^3+a^4)*v^30+(a-a^2-a^3)*v^29+(-1-a^2+a^3-a
          ^4)*v^28+(-1+a-a^3)*v^27+(1-a^3+a^4)*v^26+(-1-a-a^2+a^3+a^4)*v
          ^25+(-1-a^2-a^3)*v^24+(-a-a^2+a^4)*v^23+(-1+a+a^3+a^4)*v^22+(1-a
          ^2+a^3-a^4)*v^21+(-a+a^2)*v^20+(a+a^2+a^3+a^4)*v^19+(1-a+a^2+a^4)*
          v^18+(a-a^2+a^3)*v^17+(a^2-a^4)*v^16+(-1+a-a^2+a^3+a^4)*v^15+(1-a
          ^2-a^4)*v^14+(-a+a^3)*v^13+(1+a-a^2+a^3)*v^12+(1+a-a^2-a^3)*v
          ^11+(-1+a^2-a^3)*v^10+(1-a^3)*v^9+(-a^2-a^3-a^4)*v^8+(a+a^3-a^4)*v
          ^7+(-1-a+a^2+a^3+a^4)*v^6+(-1+a-a^4)*v^5+(-1-a-a^4)*v^4+(1+a-a^2-a
          ^3+a^4)*v^3+(-1+a+a^3-a^4)*v^2+(1-a+a^2+a^3)*v+(-a-a^2);
103
104   // Check that the transformed challenge is indeed tmp1/tmp2
105   assert val eq a^170*tmp1/tmp2;
106
107   gf2pol<x2>:=PolynomialRing(gfext);
108   coef1:=Eltseq(tmp1);
109   tmp1pol:=&+[coef1[I]*x2^(i-1): i in [1..135]];
110   coef2:=Eltseq(tmp2);
111   tmp2pol:=&+[coef2[I]*x2^(i-1): i in [1..135]];
112
113   // See that tmp1 and tmp2 indeed factor into low degree polynomials
114   Factorization(tmp1pol);
115   Factorization(tmp2pol);
116
117   // Illustration that Log of (v+1) and log of (v+a) are known:
118   logvplus1:=&+[q^i:i in [0..102]]+&+[q^i:i in [0..165]];
119   assert (v+1)^(k*cof) eq L0^logvplus1;
120
121   logvplusa:=
122   1150291103300623318480725612702655127268535464829713319181341981174024 5
```

```
123  190963697726645171950956909763178911701859180739031364682689567405338 52
124  624365100922034626861751010415619212994214310424591906719300764705918 06
125  642757944685681167526153023142115522631940703743142812325146128130300 08
126  444884591730766488116179755184373227443936427093745050548824173473302 28
127  404697047951809200515878863752612298867746453248962880650721279340168 07
128  908410190538575552872120699842390223067425412775613995629934148121811 72
129  790627629945548647854990203880789128441261335141017675097867844928815 82
130  961256600048992950331313814096591316491983698974803579316710574;
131
132  assert (v+a)^(k*cof) eq L0^logvplusa;
133
134  // Testing a degree 18 log example (fifth polynomial in the
         factorisation of tmp1pol)
135  img:=Evaluate(Factorization(tmp1pol)[5][1],v);
136  testlog:=
137  371363955803006631297819576651521722487051783961640982770176918718796 01
138  626744300524754636567467294781016057552660430394732533606094231001189 31
139  022280984089443435228277940164125042427431264990322203952576386992612 07
140  611078774291513315792324688252138341478066839898198639198731362481244 30
141  884409972737351098836477341723361898981075689768617097657105906354167 66
142  970251914605653114639999373276530373380705560894240103551307439250466 92
143  551199291164230524095459096849388759750872239935396789815852443663164 08
144  671886745213650355770063377378747339210686383014542601681926351052942 82
145  942217524010594800996863809226384714979532235158841503542483 7;
146
147  // Verify correctness of the log
148  assert img^cof eq L0^testlog;
```

E-mail address: antoine.joux@m4x.org
E-mail address: cecile.pierrot@inria.fr