Discrete Logarithm Factory

Haetham Al Aswad*, Cécile Pierrot, and Emmanuel Thomé

Université de Lorraine, CNRS, INRIA Nancy, France.

Abstract. The Number Field Sieve and its variants are the best algorithms to solve the discrete logarithm problem in finite fields. The Factory variant accelerates the computation when several prime fields are targeted. This article adapts the Factory variant to non-prime finite fields of medium and large characteristic. We combine this idea with two other variants of NFS, namely the tower and special variant. This combination leads to improvements in the asymptotic complexity. Besides, we lay out estimates of the practicality of this method for 1024-bit targets and extension degree 6.

1 Introduction

Context. The discrete logarithm problem in a cyclic group $\mathbb{G}$ with a generator $g \in \mathbb{G}$ is the computational problem of finding an integer x modulo $|\mathbb{G}|$ for a given target $T \in \mathbb{G}$, such that $T = g^x$. Despite the growing interest in post-quantum cryptography, the discrete logarithm problem is still at the basis of many currently-deployed public key protocols. This article deals with the discrete logarithm problem in the group of invertible elements of a finite field, $\mathbb{G} = \mathbb{F}_{p^n}^*$, excluding small characteristic finite fields due to the existence of quasi-polynomial time algorithms [BGJT14, GKZ14, KW22]. Therefore, our attention is restricted here to medium and large characteristic finite fields. We recall the usual notation[1] $L_Q(\alpha, c) = \exp((c + o(1)) \cdot (\log Q)^{\alpha} (\log \log Q)^{1-\alpha})$ where $o(1)$ tends to 0 as $Q = p^n$ tends to infinity. With this notation, a family of finite fields of size Q and characteristic p is said to be of medium characteristic if $p = L_Q(\alpha)$ with $1/3 < \alpha < 2/3$, and of large characteristic if this statement holds with $2/3 < \alpha$. This latter case includes prime fields where $n = 1$ and $p = L_Q(1)$.

The Number Field Sieve. Initially proposed as an integer factoring algorithm in the 90's [LLMP90, BLP93], the Number Field Sieve (NFS) was later adapted to the discrete logarithm problem in prime fields [Gor93], and medium and large characteristic finite fields [JLSV06]. The NFS family includes numerous variants to compute discrete logarithms in finite fields in time $L_{p^n}(1/3, c)$ for some constant $0 < c < 2.3$ that depends on the precise sub-case. For medium and large characteristic finite fields, the most efficient algorithm to compute discrete logarithm is some variant of NFS. We mention a few variants of interest. The special

* Funded by French Ministry of Army - AID Agence de l'Innovation de Défense.

[1] We use $L_Q(\alpha)$ instead of $L_Q(\alpha, c)$ when the value of c does not matter.

variant, SNFS [JP14] applies when the characteristic p is sparse, i.e., is the evaluation of a polynomial of relatively small degree and small coefficients, resulting in a more efficient algorithm than NFS, in both medium and large characteristic finite fields. The multiple variant, MNFS [Mat03, BP14, Pie15, SS16b] has a lower complexity than NFS in medium and large characteristic. The Tower variant, TNFS[2] [KB16, KJ17, SS19] is more efficient than NFS in medium characteristic finite fields when the extension degree is composite. When the characteristic is sparse and of medium size, and when the extension degree is composite, TNFS can be coupled with the SNFS resulting in the STNFS algorithm [KB16, KJ17]. Table 1 summarizes the asymptotic complexities of NFS and its variants. The boundary case between medium and large characteristic area is not represented in this table as complexities are functions of p and not constant.

Algorithm	Characteristic area	
	Medium	Large
Every finite field		
NFS	$(96/9)^{1/3} \approx 2.20$	$(64/9)^{1/3} \approx 1.92$
Multiple NFS	$((72+32\sqrt{6})/15)^{1/3} \approx 2.16$	$((92+26\sqrt{13})/27)^{1/3} \approx 1.90$
Composite extension degree		
Tower NFS	$\geq (48/9)^{1/3} \approx 1.75$	$(64/9)^{1/3} \approx 1.92$
Multiple Tower NFS	$\geq ((3+4\sqrt{(2/3)})/10)^{1/3} \approx 1.71$	$((92+26\sqrt{13})/27)^{1/3} \approx 1.90$
Sparse characteristic		
Special NFS	$\geq (64/9)^{1/3} \approx 1.92$	$(32/9)^{1/3} \approx 1.53$
Sparse characteristic and composite extension degree		
Special Tower NFS	$\geq (32/9)^{1/3} \approx 1.53$	$(32/9)^{1/3} \approx 1.53$

Table 1. Summary of the variants of NFS. All the asymptotic complexities are in $L_Q(1/3, c)$. This table indicates the exact value and then an approximation of c in each case. Each algorithm applies to all finite fields that satisfy the constraint expressed in bold above it. The complexities of SNFS and STNFS for medium characteristic are functions of on another parameter λ that is not represented in this table.

The general framework that is common to all variants of NFS is abundantly described in the literature, and we will briefly recall it in this article. The NFS framework sets up an algebraic context within which the target finite field $\mathbb{F}_{p^n}$ is presented in two or more distinct ways as quotient rings of number fields, bound together in a commutative diagram. Setting up this algebraic context is referred to as the *polynomial selection*, and to a large extent the polynomial selection is the main differentiating point between most variants mentioned above. Then smooth elements are found in a relation collection step, that permits afterwards to solve a linear system and get the logarithm of some particular elements. Arbitrary discrete logarithms are reconstructed in the last step called the individual logarithm step.

[2] Sometimes referred to as the extended Tower Number Field Sieve (exTNFS).

From a practical standpoint, the state of the art for the computation of discrete logarithms in finite fields of small extension degree has been regularly updated. In particular, recent work has shown that the TNFS variant is practical. De Micheli, Gaudry and Pierrot [MGP21] reported in 2021 the first implementation of TNFS and performed a record computation on a 521-bit finite field with extension degree $n = 6$. One year later, Robinson [Rob22] reported a record computation using TNFS on a 512-bit finite field of extension degree $n = 4$. On the "usual" NFS side, the latest record on a prime field $\mathbb{F}_p$ was done with NFS in 2019 in a 795-bit finite field [BGG$^+$20], although that computation was a lot more massive than the one in [MGP21]. Table 2 lists some of these recent computations. SNFS is also very practical as well, and is able to target finite fields of much larger sizes, such as a 1024-bit prime field in [FGHT17].

Finite field	Bitsize of p^n	Year	Team
$\mathbb{F}_p$	795	2019	Boudot, Gaudry, Guillevic, Heninger, Thomé, Zimmermann
$\mathbb{F}_{p^2}$	595	2015	Barbulescu, Gaudry, Guillevic, Morain
$\mathbb{F}_{p^3}$	593	2019	Gaudry, Guillevic, Morain
$\mathbb{F}_{p^4}$	512	2022	Robinson
$\mathbb{F}_{p^5}$	324	2017	Grémy, Guillevic, Morain
$\mathbb{F}_{p^6}$	521	2021	De Micheli, Gaudry, Pierrot
$\mathbb{F}_{p^{12}}$	203	2013	Hayasaka, Aoki, Kobayashi,Takagi

Table 2. Discrete logarithm records [Gré17] in finite fields of various extension degrees, performed with the Number Field Sieve. TNFS is only implemented for the $\mathbb{F}_{p^4}$ and the $\mathbb{F}_{p^6}$ records.

Attacking one key versus attacking many keys. This article studies how the cryptanalysis cost for several public keys evolves with the number of targeted keys. We identify two distinct situations. When the finite field is fixed, an adversary willing to compute several discrete logarithms at the same time can take advantage of the fact that the first steps of NFS only depend on the group under consideration, not on the specific target whose logarithm is desired. This is how the Logjam attack [ABD$^+$15] was carried out, by precomputing a data depending on the finite field only, and useful afterwards for all the individual logarithm computations.

In this work, we look at the problem from a different angle. A certain finite field bitsize is fixed, for example following a given cryptographic recommendation. Is there a more efficient way to solve the discrete logarithm problem in several different finite fields of this given bitsize, rather than using NFS (or its variants) on each field separately? In particular, is there a scheme where some kind of precomputation is beneficial? A precise answer depends on whether the set of target finite fields is known before the attack begins, which impacts the possible uses of the attack. In both cases though, such an attack scenario is referred to as a Factory-like computation, owing to the state-of-the-art algorithms described below.

Factoring Factory and discrete logarithm Factory. In 1993, Coppersmith presented the *Factorization Factory* algorithm [Cop93] to factor many numbers in a more efficient way than applying NFS on each of the numbers. The idea is to amortize the cost of a precomputation over many factorizations, by finding smooth elements in a relation collection phase that is only half done but that can be used for each of the different factorizations. With a reduction of the overall factoring effort by more than 50%, Kleinjung, Bos and Lenstra used this idea and managed to factor 17 Mersenne numbers [KBL14]. Coppersmith's idea was adapted to the computation of discrete logarithm in several prime finite fields by Barbulescu in his PhD Thesis [Bar13].

Non-prime finite fields arise in the wild. The relevance of the existing Factory-like methods that we just mentioned is lessened by their applicability to prime fields only. The purpose of this article is to address this issue. Discrete logarithms in cryptography are not restricted to prime fields. Several cryptographic protocols rely on the hardness of the discrete logarithm problem in non-prime fields. For instance, pairing-based protocols entail considering families of finite fields of fixed extension degree. Even if we can work with prime extensions, most often extension degrees are composite (e.g. $n = 12$). As an illustration, we find non prime fields in the Elliptic Curve Direct Anonymous Attestation protocol that is embedded in the current version of the Trusted Platform Module [TCG19]. The emergence of SNARKs [Gro16, GWC19, CHM$^+$20], which also require pairing friendly curves accentuates the interest for these non-prime fields.

Our work. In this article, we generalize the *discrete logarithm Factory* algorithm to finite fields of any extension degree. Several difficulties arise. The primary challenge in this generalization lies in the need to adapt the algebraic framework of NFS: the goal is to construct several branches of a diagram landing in several different finite fields, but starting from the same shared branch. The way in which this diagram is created depends very much on the polynomial selection, and thus on the considered variant. We manage to combine the Factory idea with several variants: NFS, TNFS, SNFS and STNFS. The second difficulty appears in the characterization of the primes for which a given Factory algorithm can apply. We show that this can be quantified based on the Frobenius density theorem.

For each variant that we can combine with Factory, we provide, based on usual NFS heuristics, a new lower asymptotic complexity for the discrete logarithm problem, with the requirement of a one-time precomputation that is solely dependent on the bitsize of the finite fields. This complexity analysis is clearly another difficult point of our work because of the accumulated technicalities. Let us give the example of TNFS when we target several finite fields of size close to Q. With a single computation that approximately costs $L_Q(1/3, 1.94)$, we lower the complexity of TNFS *per field* from roughly $L_Q(1/3, 1.75)$ to $L_Q(1/3, 1.37)$. Our work obtains several results of this kind for various sub-cases: Table 3 recapitulates the asymptotic complexities that we obtain.

Besides, we employ an analytic approach in order to assess the crossover point above which our Factory approach for TNFS is likely to be profitable.

Algorithm	Range	Usual approach	Multiple variant	**Our work** (Factory) Precomputation	Computation in each field
NFS	Prime fields	1.92	1.90	2.01 [Bar13]	1.64 [Bar13]
	Large p	1.92	1.90	**2.01**	**1.64**
	$p = L_Q(2/3)$	**Figure 8**			
	Medium p	2.20	2.16	**2.45**	**1.73**
TNFS	Medium p	1.75	1.71	**1.94**	**1.37**
SNFS	Large p	1.53	-	**1.85**	**1.39**
	Medium p	**Table 9**			
STNFS	Medium p	**Table 10**			

Table 3. Approximation of asymptotic complexities of NFS, MNFS, NFS Factory and their variants, expressed as $L_Q(1/3, c)$. This table indicates an approximation of c in each case. When the characteristic p is expressed as $p = L_Q(2/3, c_p)$, it represents the boundary case between medium and large characteristic. At this boundary, the complexities are given as a function of c_p. For this reason we give a figure and not a formula. Besides, in medium characteristic finite fields, both the complexities of SNFS and STNFS depend on an integer parameter λ. Tables 9 and 10 give the complexities for various values of λ. Moreover, the Multiple variant does not couple with the Special variants SNFS and STNFS.

When applied to the case of 1024-bit finite fields of extension degree $n = 6$, our estimates suggest that TNFS Factory is computationally more efficient than applying TNFS on each finite field separately when solving discrete logarithms in several tens of such finite fields.

Possible impact. One of the scenarios we had in mind during the course of this study involves the potential risk of compromising the security of standardized key sizes. Recommended key sizes correspond to the sizes of finite fields considered secure against the most efficient algorithms for attacking the discrete logarithm problem, namely NFS and its variants. Each previously recommended or current key size (e.g. 1024 bits, 2048 bits, 4096 bits, etc.) is associated with a specific level of security. As a result, the distribution of finite fields used in practical applications is not uniform across all possible sizes, but rather organized into groups or packages. Consequently, an attacker seeking to compromise multiple keys potentially across different finite fields, can leverage the idea of Factory. By adjusting the parameters and finding the most advantageous trade-off in terms of the number of compromised finite fields and the cost they are willing to invest in precomputation, they can minimize the overall expense. In any case, the aggregation of finite fields within packages resulting from protocol standardization has the potential to weaken a significant proportion of the public keys generated according to these standards.

Outline of the article. We start with a short refresher concerning NFS and its variants in Section 2. Section 3 presents the Factory idea adapted to non prime finite fields. Section 4 details then the asymptotic complexity results of this algorithm, while in Section 5 we discuss the feasibility and impact of this method on moderate key sizes, for instance targets elements living in several 1024-bit finite fields.

2 Background

Notations. From now on, p always denotes a prime number. When the extension degree n of the finite field $\mathbb{F}_{p^n}$ is composite, η and κ denote non trivial factors of n and such that $n = \eta\kappa$. Asymptotic estimates use the classical $O()$ and $o()$ notations, as well as the soft-O notation $f = \widetilde{O}(g)$ which means that there exists a constant c such that $f(x) = O(g(x)\log^c(x))$, as x tends to infinity. We recall that an integer is said to be *x-smooth* if we can write it as product of integers that are all smaller than x.

2.1 The (Tower) Number Field Sieve

We start with a short refresher on the Tower variant of the Number Field Sieve, of which the "usual" NFS can be considered a special case.

Commutative diagram. We target the finite field $\mathbb{F}_{p^n}$. Let η be a divisor of n. The classical TNFS setup considers the intermediate number field $\mathcal{K}_h = \mathbb{Q}(\iota)$ where ι is a root of h, a polynomial of degree η over $\mathbb{Z}$ that remains irreducible modulo p. We let $\mathcal{R}$ be the ring of integers of $\mathcal{K}_h$. For simplicity, we assume throughout this article that h is monic and furthermore that it is chosen so that $\mathcal{R} = \mathbb{Z}[\iota]/h$. (For the usual NFS, we rather let $\eta = 1$, $\mathcal{K}_h = \mathbb{Q}$, and $\mathcal{R} = \mathbb{Z}$; in particular there is no requirement that n be composite.) Above $\mathcal{K}_h$, define two number fields $\mathcal{K}_0 = \mathcal{K}_h[x]/f_0(x)$ and $\mathcal{K}_1 = \mathcal{K}_h[x]/f_1(x)$ where f_0, f_1 are irreducible polynomials over $\mathcal{R}$ that share an irreducible factor φ of degree κ modulo the unique ideal $\mathfrak{p}$ over p in $\mathcal{K}_h$ (in particular, f_0 and f_1 have degree at least κ). We write $\mathcal{O}_i$ the ring of integers of $\mathcal{K}_i$ and α_i a root of f_i in $\mathcal{K}_i$ for $i = 0, 1$. Because of the conditions on the polynomials h, f_0 and f_1, there exist two ring homomorphisms from $\mathcal{R}[x]$ to the target finite field $\mathbb{F}_{p^n}$ through the number fields $\mathcal{K}_0$ and $\mathcal{K}_1$. This allows to build a commutative diagram as shown in Figure 4. For simplicity, we will assume that f_0 and f_1 are defined over $\mathbb{Z}$, although this is only possible when κ and η are coprime.

Based on the diagram above, we briefly comment the steps of NFS in the following paragraphs. The *polynomial selection* step is the way the diagram of Figure 4 is built. For an appropriate notion of size that is defined in the intermediate number fields, the *relation collection* step accumulates relations between "small" elements in the number fields. Their images in the target finite field are then recovered by the *linear algebra* step, and the process is made more general by the *individual logarithm* step which leverages the acquired information to compute logarithms of arbitrary elements of the target number field.

$$\begin{tikzcd}[column sep=small]
 & \mathcal{R}\left[X\right] \arrow[dl] \arrow[dr] & \\
\mathcal{K}_0 \supset \mathcal{R}\left[X\right]/f_0(X) \arrow[dr, "\text{mod } \varphi,\text{ mod } p"'] & & \mathcal{K}_1 \supset \mathcal{R}\left[X\right]/f_1(X) \arrow[dl, "\text{mod } \varphi,\text{ mod } p"] \\
 & \mathcal{R}/\mathfrak{p}[X]/\varphi(X) \cong \mathbb{F}_{p^n} &
\end{tikzcd}$$

Fig. 4. Commutative diagram of Tower NFS. $\mathcal{R}$ is the ring of integers of $\mathcal{K}_h = \mathbb{Q}[x]/h$.

Polynomial selection. Several methods to do NFS polynomial selection are known. For example, the Conjugation, JLSV or Sarkar-Singh's methods [BGGM15, JLSV06, SS16b] can be used. Each polynomial selection method yields different degrees and coefficient sizes. A table summing up all the parameters for f_0 and f_1 output by various polynomial selections for NFS and its variants (Multiple, Tower, Special and composition of two of it) is given in [DM21, Section 3.4.2]. In this work we do not deal with all the polynomial selections.

Relation collection. The goal of the relation collection step is to select among the set of polynomials $\phi(x, \iota) \in \mathcal{R}[x]$ at the top of the diagram the candidates that yield a relation. A relation is found if the polynomial $\phi(x, \iota)$ mapped to $\mathcal{K}_0$ and $\mathcal{K}_1$ factors is *smooth* on both sides, meaning that it factors into products of ideals in $\mathcal{O}_0$ and $\mathcal{O}_1$ whose norm is below some smoothness bounds B_0 and B_1. Most often the search space for relation collection consists of linear polynomials $\phi(x, \iota) = a(\iota) - b(\iota)x \in \mathcal{R}[x]$, and for usual NFS this simplifies to searching for polynomials $a - bx$ with integers coefficients a, b, since $\mathcal{R} = \mathbb{Z}$ in that case. The ideals that occur in the factorizations in $\mathcal{O}_0$ and $\mathcal{O}_1$ constitute the factor basis $\mathcal{F}$. More precisely, we define it as the disjoint union $\mathcal{F} = \mathcal{F}_0 \sqcup \mathcal{F}_1$ with, for $i = 0, 1$:

$$\mathcal{F}_i(B_i) = \{\text{prime ideals of } \mathcal{O}_i \text{ of norm} \leq B_i, \text{ whose inertia degree over } \mathbb{Q}(\iota) \text{ is } 1\}.$$

To verify the B_i-smoothness on each side, one needs to evaluate the norms $N_i(a(\iota) - b(\iota)\alpha_i)$ for $i = 0, 1$. To do so, we can write:

$$N_i(a(\iota) - b(\iota)\alpha_i) \stackrel{*}{=} \mathrm{Res}_t(\mathrm{Res}_x(a(t) - b(t)x, f_i(x)), h(t)). \tag{1}$$

where the equality $\stackrel{*}{=}$ holds up to sign and up to powers of the leading coefficients of h and f_i. Since resultants are integers, this allows to verify the B_i-smoothness over integer values. (When h and f_i are not monic, we include in the factor basis the few ideals that divide their leading coefficients, but this is an unimportant technicality). The relation collection step stops when we have enough relations to construct a system of linear equations that may be full rank. The unknowns of these equations are the *virtual* logarithms of the ideals of the factor basis.

Linear algebra. A good feature of the linear system created is that the number of non-zero coefficients per line is very small. This allows to use sparse linear algebra algorithms such as Coppersmith's block Wiedemann algorithm [Cop94], for which parallelization is partly possible. The output of this step is a kernel vector corresponding to the virtual logarithms of the ideals in the factor basis.

Individual discrete logarithm. The final step consists in finding the discrete logarithm of the target element. This step is subdivided into two substeps: a smoothing step and a descent step. The smoothing step is an iterative process where the target element t is randomized until the randomized value lifted back to one of the number fields $\mathcal{K}_i$ is B'_i-smooth for a smoothness bound $B'_i > B_i$. The second step consists in decomposing every factor of the lifted value, in our case prime ideals with norms less than a smoothness bound B'_i, into elements of the factor basis for which we now know the virtual logarithms. This eventually makes it possible to reconstruct the discrete logarithm of the target element.

TNFS differs from NFS in this step as there exist improvements for the smoothing step when the target finite field has proper subfields [Gui19, AP22].

2.2 Other variants of NFS

Special NFS. When the characteristic is sparse, both NFS and TNFS can be adapted so that the polynomials in the sieving step have lower norms, resulting in better asymptotic complexities. This is called the Special variant of NFS and written SNFS or STNFS. The key idea as explained in [JP14] lies in a dedicated polynomial selection that takes advantage of the sparsity of the characteristic.

Multiple NFS. NFS and TNFS can be coupled with a multiple variant too [Mat03, BP14, Pie15, SS16b], the main idea being to have a lot of different intermediate number fields where a polynomial from the sieving can be smooth. MNFS and MTNFS give the best asymptotic complexities. However we don't detail this variant as we do not see a way to adapt the Factory algorithm to it. Similarly, the special variant and multiple variant cannot work together.

2.3 Smoothness probability

A key heuristic assumption in the analysis is that the probability of a norm being smooth is the same as that of a random integer of the same size. This allows us to apply the theorem by Canfield-Erdős-Pomerance [CEP83], which we choose to state as follows.

Corollary 1. *Let $(\alpha_1, \alpha_2, c_1, c_2)$ be four real numbers such that $1 > \alpha_1 > \alpha_2 > 0$ and $c_1, c_2 > 0$. As Q tends to infinity, the probability that a random positive integer below $L_Q(\alpha_1, c_1)$ splits into primes less than $L_Q(\alpha_2, c_2)$ is*

$$L_Q\left(\alpha_1 - \alpha_2, (\alpha_1 - \alpha_2)\, c_1\, c_2^{-1}\right)^{-1}.$$

The norms are estimated based on Equation (1). In the classical (non-Tower) NFS, the definition of the resultant as the determinant of the Sylvester matrix gives a bound that follows from Hadamard's inequality:

$$|\operatorname{Res}(\phi, f_i)| \leq ||\phi||_\infty^{\deg f_i} \cdot ||f_i||_\infty^{\deg \phi} \cdot (\deg f_i + 1)^{\deg \phi/2}(\deg \phi + 1)^{\deg f_i/2}.$$

When analyzing tower variants, the degree of h appears in the resultant. Since we assumed that $\mathcal{R} = \mathbb{Z}[\iota]$, we can assume that all coefficients of $\phi(x,t)$ are integers. If these integer coefficients are bounded in absolute value by E, we obtain the following bound:

$$|\operatorname{Res}_t(\operatorname{Res}_x(\phi, f_i), h)| \leq E^{\deg h \cdot \deg f_i} \cdot ||f_i||_\infty^{\deg h \cdot \deg_x \phi} \cdot ||h||_\infty^{\deg f_i \cdot \deg_t \phi} \cdot c$$

where the factor c is a combinatorial contribution that can be uniformly bounded depending on $\deg f_i$ and $\deg h$, and is negligible compared to the other factors in all cases we consider in this article. Note also that in all cases of interest, we have $\deg_x \phi = 1$ and $\deg_t \phi < \deg h$.

3 Discrete logarithm Factory

Whether it is deployed for integer factorization, for discrete logarithm in prime finite fields or in medium or large characteristic finite fields as in this article, the *Factory* algorithm revolves around the same idea. Given a chosen bitsize, the primary objective is to share a portion of the *relation collection* step in NFS (or one of its variants), to serve the factorization of several numbers, or to solve discrete logarithm in many finite fields, all of the same bitsize. Specifically, the *Factory* algorithm consists of two steps, which we detail here.

The common setting is a given order of magnitude Q as well as a fixed extension degree n. The goal, ultimately, is to compute discrete logarithms in many finite fields $\mathbb{F}_{p_1^n}, \mathbb{F}_{p_2^n}, \ldots$, with $p_1^n \approx p_2^n \approx Q$.

The "one-off" step. We construct *half* of the diagram of Figure 4 by computing $\mathcal{K}_h$ and $\mathcal{K}_0$. Afterward, a first search aims to identify (and store for later usage) elements ϕ in the search space that are B_0-smooth when mapped to $\mathcal{K}_0$, for a fixed smoothness bound B_0. All parameters of this one-off step, including of course the bound B_0 as well as the number of elements ϕ to test are tuned according to Q and n.

The "per-field" step. After the one-off step, challenges are considered as a collection of primes $p_1, p_2, \ldots$. For each of theses prime numbers (say p_i), the goal of the per-field step is to solve the discrete logarithm in the finite field $\mathbb{F}_{p_i^n}$, where p_i^n is close to Q. Once the diagram is complete, the *relation collection* step proceeds by testing the stored elements to determine which are B_*-smooth when mapped to $\mathcal{K}_i$, where B_* is another fixed smoothness bound. Because this per-field step is designed to work in a similar way for several primes p_i of similar size, parameters such as B_* are going to be identical for all of them. The remaining steps of NFS (or one of its variants) are unchanged.

3.1 A baseline: Factory algorithm for prime fields

The factorization Factory algorithm, introduced by Coppersmith [Cop93], and its adaptation to the discrete logarithm problem in prime finite fields, proposed by Barbulescu [Bar13], both use a *base-m method* for the polynomial selection. We describe this method for the discrete logarithm problem in prime fields.

Given an order of magnitude Q and an integer d, one chooses an integer m below $Q^{1/d}$, and let $f_0(X) = X - m$. Then one sets $\mathcal{K}_h = \mathbb{Q}$ and $\mathcal{K}_0 = \mathbb{Q}[X]/f_0 \simeq \mathbb{Q}$ for the precomputation step. To solve the discrete logarithm in a prime field $\mathbb{F}_{p_i}$, where $Q \leq p_i < m \times Q$, one computes the base-m expansion of p_i as $p_i = \sum_{k=0}^{d} a_k m^k$ and defines $f_i(X) = \sum_{k=0}^{d} a_k X^k$. Then, f_0 and f_i share a common root modulo p_i, which is m. We define $\mathcal{K}_i$ as $\mathbb{Q}[X]/f_i$ (the polynomial f_i is generically irreducible). This completes Diagram 4, and the elements stored at the *precomputation* step can be used to solve the discrete logarithm in $\mathbb{F}_{p_i}$.

3.2 Generalization to non prime finite fields

The novelty of this article is the generalization of the discrete logarithm Factory to finite fields of any extension degree, instead of only extension degree $n = 1$. Specifically, we aim to compute a table that can be used to efficiently solve the discrete logarithm problem in several finite fields $\mathbb{F}_{p_i^n}$ (recall that all p_i^n are of the same size, and n is the same for all). Since $n > 1$, both number fields $\mathcal{K}_0$ and $\mathcal{K}_i$ must be of degree greater than one over $\mathbb{Q}$, hence, the *base-m* polynomial selection used in the discrete logarithm Factory for prime fields does not allow this generalization. We make use of other polynomial selection methods, depending on the size of the characteristic and the variant of NFS. However, not every polynomial selection is adapted for a Factory variant of NFS: since we want to precompute a table of smooth elements in one number field, and share this table for several finite fields, it means that we need to be able to draw a diagram as Diagram 5:

Given a polynomial f_0, constructing a compatible polynomial f_i for any target finite field is not an easy task.

Let $n = \eta\kappa$ be a factorization of the extension degree, where η is a non trivial divisor of n in the tower variants TNFS and STNFS, and $\eta = 1$ otherwise. The polynomial $h \in \mathbb{Z}[X]$ of degree η defining $\mathcal{K}_h$ has to be irreducible and can be selected before searching for f_0 and f_i. Define $\mathcal{R} = \mathbb{Z}[\iota]$ where ι is a root of h in $\mathcal{K}_h$. In order to simplify the exposition, we require that η and κ are coprime, which allows us to search for f_0 and f_i with integer coefficients instead of coefficients in $\mathbb{Z}[\iota]$. Both f_0 and f_i must be coprime and irreducible over $\mathbb{Q}$, and share an irreducible factor φ_i of degree κ modulo p_i. Then $\mathbb{F}_{p_i^n}$ is represented as $(\mathcal{R}/p_i\mathcal{R})[X]/(\varphi_i)$. Other properties, such as small coefficient sizes and small degrees for h, f_0 and f_i are desired for the efficiency of NFS and NFS Factory, or their variants.

$$\begin{tikzcd}
 & \mathcal{R}[X] \arrow[rr] \arrow[dl] \arrow[dr] \arrow[dd] & & \mathcal{K}_3 \arrow[dd] \\
\mathcal{R}[X]/(f_1(X)) \subset \mathcal{K}_1 \arrow[dd, "\text{mod } \varphi_1"' pos=0.3, "\text{mod } p_1"' pos=0.7] & & \mathcal{K}_2 \arrow[dd, "\text{mod } \varphi_2"' pos=0.3, "\text{mod } p_2"' pos=0.7] & \\
 & \mathcal{R}[X]/(f_0(X)) \subset \mathcal{K}_0 \arrow[rr] \arrow[dl, "\text{mod } \varphi_1" pos=0.3, "\text{mod } p_1" pos=0.7] \arrow[dr, "\text{mod } \varphi_2"' pos=0.3, "\text{mod } p_2"' pos=0.7] & & \mathbb{F}_{{p_3}^n} \\
\mathcal{R}/\mathfrak{p}_1[X]/(\varphi_1(X)) \cong \mathbb{F}_{{p_1}^n} & & \mathbb{F}_{{p_2}^n} &
\end{tikzcd}$$

Fig. 5. Example of a commutative diagram for Factory for three target finite fields. The blue branch is the shared one.

Polynomial selections for Factory. We review the different polynomial selection methods that we can use for NFS Factory. When working with a Tower variant, the irreducible polynomial $h \in \mathbb{Z}[X]$ of degree η is assumed to be already fixed. In each of the cases below, the construction works only if specific requirements on the primes p_i are met. Those requirements are studied in Section 3.3.

Generalized-Joux-Lercier [BGGM15] Factory. Choose $f_0 \in \mathbb{Z}[X]$ irreducible, of degree $d+1 > \kappa$ for some integer d, and with small integer coefficients. Following the description at the beginning of Section 3, the *precomputation* step can be carried out based on f_0.

Let p_i be a prime number such that h is irreducible modulo p_i, and f_0 admits an irreducible factor modulo p_i of degree κ, which we lift to an integer polynomial as $\varphi_i(X) = X^\kappa + \sum_{j=0}^{\kappa-1} \varphi_j X^j$ with $-p_i/2 < \varphi_j \le p_i/2$ for $0 \le j \le \kappa - 1$. Build the lattice of dimension $(d+1) \times (d+1)$ whose basis is given by:

$$M_{p_i} = \begin{pmatrix} p_i & & & & & \\ & \ddots & & & & \\ & & p_i & & & \\ \varphi_0 & \varphi_1 & \dots & 1 & & \\ & \ddots & \ddots & & \ddots & \\ & & \varphi_0 & \varphi_1 & \dots & 1 \end{pmatrix} \begin{array}{l} \left.\vphantom{\begin{matrix} p_i \\ \ddots \\ p_i \end{matrix}}\right\} \kappa \text{ rows} \\ \left.\vphantom{\begin{matrix} \varphi_0 \\ \ddots \\ \varphi_0 \end{matrix}}\right\} d+1-\kappa \text{ rows} \end{array}$$

The shortest vector output by the LLL algorithm [LLL82] when applied to M_{p_i} gives a polynomial f_i that is an integer linear combination of $\left(p_i X^\ell\right)$ and $\left(X^k \varphi_i\right)$ for $0 \le \ell \le \kappa - 1$ and $0 \le k \le d - \kappa$. Thus φ_i divides f_i modulo p_i. We can safely assume that f_0 is irreducible over $\mathbb{Z}$; in the unlikely event that it is not, we can replace it with the appropriate irreducible factor that reduces modulo p_i to a multiple of φ_i. Alongside with h, the polynomial f_i defines $\mathcal{K}_i$, and φ_i defines $\mathbb{F}_{p_i^n}$, as in Diagram 5. The *computation per field* step can proceed by using the stored table (which does not depend on p_i) to solve the discrete logarithm problem in $\mathbb{F}_{p_i^n}$. Moreover, as the dimension of M_{p_i} is $d+1$, and its determinant is p_i^κ, LLL guarantees that the degree of f_i is at most d, and its coefficients have sizes in $\widetilde{O}\left({p_i}^{\kappa/(d+1)}\right)$.

Conjugation [BGGM15] Factory. Select g_0 and g_1 two polynomials with small integer coefficients with $\deg g_1 < \deg g_0 = \kappa$. Select μ a quadratic, monic, irreducible polynomial over $\mathbb{Z}$ with small coefficients. Define the polynomial f_0 as $\mathrm{Res}_Y\left(\mu(Y), g_0 + Y g_1\right)$. The degree of f_0 is 2κ with coefficients in $O(1)$.

Let p_i be a prime number such that h is irreducible modulo p_i, and μ_i has a root λ_i in $\mathbb{F}_{p_i}$ such that $\varphi_i := g_0 + \lambda_i g_1$ is irreducible modulo p_i. Define $f_i = v g_0 + u g_1$, where (u, v) is a rational reconstruction of λ_i. Then $f_0 = 0 \mod \varphi_i \mod p_i$ and $f_i = v\varphi_i \mod p_i$. Thus both polynomials share φ_i as an irreducible factor modulo p_i, and f_0 is irreducible over $\mathbb{Q}$. Moreover, f_i is of degree κ with coefficient sizes in $O(\sqrt{p_i})$. Alongside with h, the polynomial f_i defines $\mathcal{K}_i$, and φ_i defines $\mathbb{F}_{p_i^n}$. Then, the stored table is used to compute discrete logarithms in $\mathbb{F}_{p_i^n}$.

Joux-Pierrot [JP14] Factory, first approach: starting from a fixed integer u. The original SNFS algorithm proposes only one polynomial selection, that is used for sparse characteristics in both medium and large characteristics finite fields. However, if we want to combine SNFS with Factory, two different approaches are possible.

For the first approach we choose two integers $\lambda > 1$ and $u \approx Q^{1/(\lambda n)}$, as well as a polynomial R of degree at most $\kappa - 1$ with coefficients 0, 1, or -1, until $f_0(X) := X^\kappa + R(X) - u$ is irreducible over $\mathbb{Q}$. We perform the *precomputation* step based on f_0 with parameters depending on Q, n and λ.

Let P_i be a polynomial of degree d_i close to λ and with small coefficients. Assume that $p_i := P_i(u)$ is prime and h and f_0 are irreducible modulo p_i. Define $f_i(X) = P_i\left(X^\kappa + R(X)\right)$. Then f_0 divides f_i modulo p_i since $X^\kappa + R(X) = u \mod f_0$ and $P_i(u) = p_i$. Thus f_0 and f_i share $f_0 \bmod p_i$ as an irreducible factor of degree κ modulo p_i. (As above, we may safely assume that f_0 is irreducible over $\mathbb{Z}$.) Moreover, as explained in [JP14], R can be chosen of degree $O(\log(\kappa))$, resulting in f_i of degree $d_i\kappa$ and coefficient sizes in $\widetilde{O}(\log(\kappa)^{d_i})$. Again alongside with h, the polynomial f_i defines $\mathcal{K}_i$, and $f_i \mod p_i$ defines $\mathbb{F}_{p_i^n}$, which allows to complete the discrete logarithm computation for $\mathbb{F}_{p_i^n}$.

Joux-Pierrot [JP14] Factory, second approach: starting from a fixed P. Choose an integer $\lambda > 1$ and a polynomial P of degree λ with small coefficients, as well as a polynomial R of degree at most $\kappa - 1$ with coefficients 0, 1, or -1, until $f_0(X) := P\left(X^\kappa + R(X)\right)$ is irreducible over $\mathbb{Q}$. In fact, as explained in [JP14], R can be chosen of degree $O(\log(\kappa))$, resulting in f_0 of degree $\lambda\kappa$ and coefficient sizes in $\widetilde{O}(\log(\kappa)^\lambda)$.

Let u_i be an integer such that $u_i \approx Q^{1/(\lambda n)}$ and $p_i := P(u_i)$ is prime and h and $X^\kappa + R(X) - u_i$ are irreducible modulo p_i. Define $f_i(X) = X^\kappa + R(X) - u_i$.

Then f_i is an irreducible factor of f_0 modulo p_i, and is irreducible over $\mathbb{Q}$. Again we complete the diagram and compute a discrete logarithm in $\mathbb{F}_{p_i^n}$.

Table 6 summarizes the polynomial selections to use in the Factory variant of NFS. Table 7 gives the degrees and infinite norms of these polynomials.

Something special? \ Characteristic	Medium	Large	Algorithm
No	Conjugation	GJL	NFS Factory
Composite extension	Conjugation	-	TNFS Factory
Sparse characteristic	Joux-Pierrot	Joux-Pierrot	SNFS Factory
Composite extension And sparse characteristic	Joux-Pierrot	-	STNFS Factory

Table 6. Polynomial selections and algorithm for Factory, according to the size of the characteristic, and the potential feature of the target finite field (sparse characteristic or composite extension).

Polynomial selection \ Properties of f_0 and f_i	$\deg(f)$	$\deg(f_i)$	$\Vert f\Vert_\infty$	$\Vert f_i\Vert_\infty$
GJL	$d+1>\kappa$	d	$\widetilde{O}(1)$	$\widetilde{O}\left(p^{n/(d+1)}\right)$
Conjugation	2κ	κ	$\widetilde{O}(1)$	$\widetilde{O}\left(\sqrt{p}\right)$
Joux-Pierrot, first approach	κ	$\lambda d,\ d\approx\lambda$	$\widetilde{O}\left(Q^{1/(\lambda n)}\right)$	$\widetilde{O}\left(\log(\kappa)^d\right)$
Joux-Pierrot, second approach	$\lambda\kappa$	κ	$\widetilde{O}\left(\log(\kappa)^\lambda\right)$	$\widetilde{O}\left(p^{1/\lambda}\right)$

Table 7. Degrees and infinite norms of the polynomials given by three different polynomial selections used for the Factory variant of NFS. In all these polynomial selections, when considering a Tower variant that needs an extra polynomial h of degree η and with small coefficients. The *one-off* step is done with h and f_0.

3.3 Fantastic primes and how many are they?

Now that we know several polynomial selections that are compatible with the factory idea, we characterize the prime numbers for which a given one-off setup (consisting of an order of magnitude Q, a fixed extension degree n, the number fields $\mathcal{K}_h$, $\mathcal{K}_0$, and a precomputed table), can be compatible with the per-field step. For this to hold for one of the challenge primes p_i, we must make sure that there exists an irreducible polynomial f_i that shares an irreducible factor of degree κ with f_0 modulo p_i.

Frobenius density Theorem [SL96]. If a degree n polynomial f splits modulo a prime number p into a product of k irreducible factors of degrees $n_1, \ldots, n_k$, we say that f has a decomposition type $(n_1, \ldots, n_k)$ over p. Similarly, if a permutation σ in the group of permutations of n points $\mathfrak{S}_n$ splits into a product of k disjoint cycles of orders $n_1, \ldots, n_k$, we say σ has type $(n_1, \ldots, n_k)$. Moreover, the density of a set of prime numbers S is defined as:

$$\lim_{X\to\infty} \frac{\#\{x < X \mid x \in S\}}{\#\{x < X \mid x \text{ is prime}\}}.$$

Let $f \in \mathbb{Z}[X]$ be an irreducible integer polynomial of degree n. Denote $\mathrm{Gal}(f)$ its Galois group, seen as a subgroup of the symmetric group $\mathfrak{S}_n$. The Frobenius density Theorem states that for $(n_1, \ldots, n_k)$ a partition of n, the set of prime numbers above which f has decomposition type $(n_1, \ldots, n_k)$ has density

$$\frac{\#\{\sigma \in \mathrm{Gal}(f) \mid \sigma \text{ has type } (n_1, \ldots, n_k)\}}{\#\,\mathrm{Gal}(f)}.$$

Application to Factory algorithms. Let $n = \eta\kappa$, h, and f_0 be as before. We assume that the decompositions of h and f_0 modulo prime numbers are independent. For each polynomial selection method, we give the density of prime numbers for which Diagram 5 can be completed with $\mathcal{K}_i$ and $\mathbb{F}_{p_i^n}$. We denote $\mathrm{Gal}(f_0)_k$ the subset of permutations of $\mathrm{Gal}(f_0)$ that have k in their pattern cycle. We also define $\sigma_{f_0,k} = \frac{\#\,\mathrm{Gal}(f_0)_k}{\#\,\mathrm{Gal}(f_0)}$. We use similar notations with respect to h. The notation $\sigma_{\mathrm{conj},f_0,\kappa}$ is defined below in Paragraph *Conjugation Factory.*

All polynomial selection methods listed in 3.2 require that for a prime p_i to work in the per-field step, h must be irreducible modulo p_i, and f_0 must have an irreducible factor of degree κ. Unless some obstruction calls for more precise investigation, we expect that the density of primes p_i for which these conditions are met is

$$\frac{\#\,\mathrm{Gal}(h)_\eta \cdot \#\,\mathrm{Gal}(f_0)_\kappa}{\#\,\mathrm{Gal}(h) \cdot \#\,\mathrm{Gal}(f_0)} = \sigma_{h,\eta} \cdot \sigma_{f_0,\kappa}.$$

In particular, this is clearly the case with the Generalized-Joux-Lercier Factory approach, and the Joux-Pierrot Factory, first approach. We review the remaining cases for which this density result is not as obvious. Note of course that whenever the algorithm used is not combined with the Tower variant, then $\eta = 1$ and h is a linear polynomial, thus $\#\,\mathrm{Gal}(h) = \#\,\mathrm{Gal}(h)_\eta = 1$.

Conjugation Factory. In the setup given in Section 3.2, the condition on p_i is that μ factors modulo p_i and that f_0 has an irreducible factor φ of degree κ. It turns out that if κ is odd, the latter implies the former: φ remains irreducible in $\mathbb{F}_{p_i^2}$, and since $f_0 = (g_0 + \beta g_1)\,(g_0 + \gamma g_1)$ where β and γ are the two distinct roots of μ in $\mathbb{F}_{p_i^2}$, we can say that φ has to be one of these two factors, hence β and γ are in $\mathbb{F}_p$. Therefore, if κ is odd, the density of primes p_i for which the per-field step works is the same as above. If κ is even, unfortunately we do not have the exact same statement, although experimental evidence suggests that a similar result holds. In both cases, we denote $\sigma_{\mathrm{conj},f_0,\kappa}$ this density.

Joux-Pierrot Factory, second approach. In order to estimate the density of prime numbers for which Factory can proceed, based on the Galois group of h and f_0, we would like to ensure that $X^\kappa + R(X) - u_i$ is irreducible modulo $p_i := P(u_i)$ whenever p_i is prime and f_0 admits an irreducible factor of degree κ modulo p_i. Unfortunately, this is not true. Instead, we estimate the density of prime numbers $p_i = P(u_i)$ with $X^\kappa + R(X) - u_i$ irreducible modulo p_i among prime numbers of the form $P(u_i)$. Short of a more satisfactory result, we rely on the heuristic that when $p = P(u)$ is prime, the probability that $X^\kappa + R(X) - u$ be irreducible is the same as the probability that a random polynomial over $\mathbb{F}_p$ be irreducible, which is $1/\kappa$. This leads us to the following.

Assumption 1 *In a large interval (a, b), the number of integers u satisfying the conditions that $p = P(u)$ is prime and $X^\kappa + R(X) - u$ is irreducible modulo p is about $1/\kappa$ times the number of integers $a < u < b$ for which $P(u)$ is prime.*

Numerical test of Assumption 1. We performed a numerical test with $\kappa = 5$, $\lambda = 2$, $P(X) = X^2 + 2X + 2$, and $R(X) = X^4 - X^3 + 1$. Using SageMath, we picked one million random integer u between 2^{30} and 2^{40}. All one million polynomials $f_u = X^5 + R(X) - u$ were found to be irreducible over $\mathbb{Z}$. Out of the one million integer, 25,823 resulted in a prime value for $P(u)$, and f_u was irreducible modulo $P(u)$ for 5,215 of them. According to Assumption 1, the number of integers u for which $P(u)$ is prime and f_u is irreducible modulo $P(u)$ should be approximately 1/5 times 25,823, yielding an approximate value of 5,165. Similar results were obtained for different polynomials with varying values of κ and λ.

3.4 Two constructions for 500 and 600-bit target finite fields

As an illustration, we exhibit two different constructions for NFS Factory and TNFS Factory, together with an evaluation of the proportion of primes (characteristics) reached by this method.

NFS Factory with Conjugation. In the 593-bit discrete logarithm computation on $\mathbb{F}_{p^3}$ reported in [GMT16], the authors performed the computation with NFS. The polynomials generated with the *Conjugation* method are:

$$\begin{aligned} f_0 &= 28X^6 + 16X^5 - 261X^4 - 322X^3 + 79X^2 + 152X + 28 \\ f_1 &= 24757815186639197370442122X^3 + 40806897040253680471775183X^2 \\ &\quad -33466548519663911639551183X - 24757815186639197370442122 \end{aligned}$$

If the precomputation of Factory is performed using the polynomial f_0, then, since 3 is odd, the expected density of prime numbers modulo which the Diagram 5 can be completed is $\#\operatorname{Gal}(f_0)_3/\#\operatorname{Gal}(f_0)$. The Galois group of f_0 comprises eighteen permutations, out of which eight have 3 in their cycle pattern. The expected density of such primes is 4/9. This is observed experimentally.

For instance, let $p_2 = 92534543354086556401570712749117100539035615701111 3$ modulo which f_0 factors into an irreducible polynomial of degree three and three linear polynomials. If we apply the method given in Section 3.2, we find another polynomial f_2, written below, that allows to complete Diagram 5. Furthermore, the largest coefficient in absolute value of f_2 is smaller than $1.45 \times \sqrt{p_2}$.

$$\begin{aligned} f_2 = {} & 17678995119854355812622458X^3 + 43866070922692969501665811X^2 \\ & -91709144368700979362015 63X - 17678995119854355812622458 \end{aligned}$$

TNFS Factory with Conjugation. In [MGP21], a 521-bit discrete logarithm computation was carried out on $\mathbb{F}_{p_1^6}$ with $p_1 = 1350664108659952233496 03927$ using TNFS where polynomials were chosen with the *Conjugation* method as:

$$\begin{aligned} h &= X^3 - X + 1, \\ f_0 &= X^4 + 1 = \mathrm{Res}_Y(X^2 + 1 + XY, Y^2 - 2), \\ f_1 &= 11672244015875X^2 + 1532885840586X + 11672244015875 \end{aligned}$$

If one desires to perform a TNFS Factory, since 2 is even, we are unable to estimate the density of prime numbers for which Factory can proceed. We evaluate this density experimentally. We randomly pick one hundred thousand primes p such that p^6 has approximately 521 bits, satisfying the conditions that h is irreducible, μ has a root, and f_0 has an irreducible factor of degree 2, with a success rate of approximately 8.30%.

For example, let us consider $p_2 = 131115867028015243141537139$ modulo which the polynomial h is irreducible, and f_0 factors into two irreducible polynomials of degree 2. By applying the method described in Section 3.2, we obtain the polynomial $f_2 := 7293863374885X^2 + 4971416414367X - 7293863374885$, which completes Diagram 5. The largest coefficient in absolute value in f_2 is smaller than $0.64 \times \sqrt{p_2}$.

It's worth noting that in this particular case, μ does not have a root modulo p_2, but we can still construct f_2 with coefficients of size $\sqrt{p_2}$. In fact, for this setup, the construction is possible whenever f_0 has an irreducible factor modulo p, regardless of the presence of a root of μ modulo p. This observation suggests that the density of prime numbers suitable for Factory given the precomputed table in this specific setup, is actually $\#\,\mathrm{Gal}(h)_3 \#\,\mathrm{Gal}(f_0)_2 / (\#\,\mathrm{Gal}(h) \#\,\mathrm{Gal}(f_0))$, that is equal to $1/4$. However, this does not hold true for the general case when κ is even, as we found counter-examples with $\kappa = 4$

4 Asymptotic analysis

This section provides the complexities of the *one-off* step and the *computation per field* step in each of the NFS variants that we combine with Factory. We compare and we refer to [BGGM15], [KB16], [Pie15], [SS16a], and [JP14] for the complexities of NFS and its variants without Factory.

Notations. For $Q = p^n$ a finite field size, c_A, c_0, and c_* are constants such that $A = L_Q(1/3, c_A)$ denotes the relation search space, i.e., the number of elements ϕ tested for smoothness in $\mathcal{K}_0$. The smoothness bounds are denoted $B_0 = L_Q(1/3, c_0)$ for $\mathcal{K}_0$ and $B_* = L_Q(1/3, c_*)$ for all the $\mathcal{K}_i$ with $i < 0$. Moreover, $\mathcal{N}_i$ denotes the sieve elements norms once mapped to $\mathcal{K}_i$ for all i. In all variants, and for all i we take the parameters such that $\mathcal{N}_i = L_Q(2/3, c_{\mathcal{N}_i})$, where $c_{\mathcal{N}_i}$ depends on c_A and other parameters. The probability of an element in $\mathcal{K}_0$ of norm $\mathcal{N}_0$ to be B_0-smooth is denoted $\mathbf{P}_0$ and is equal to $L_Q(1/3, c_{\mathcal{N}_0}/(3c_0))^{-1}$. Similarly, the probability of an element in $\mathcal{K}_i$ of norm $\mathcal{N}_i$ to be B_*-smooth is denoted $\mathbf{P}_*$ and is equal to $L_Q(1/3, c_{\mathcal{N}_i}/(3c_*))^{-1}$. This comes from Corollary 1.

Methodology. The *one-off* step is performed by a sieve algorithm detecting elements that are B_0-smooth once mapped to $\mathcal{K}_0$. The asymptotic complexity of this step is A. The number of elements to be stored for later use in *computation per field* step is the number of sieve elements that are B_0-smooth once mapped to $\mathcal{K}_0$, that is $A\mathbf{P}_0 = L_Q(1/3, c_A - c_{\mathcal{N}_0}/(3c_0))$. The *computation per field* step starts by detecting which of the stored elements are B_*-smooth once mapped to $\mathcal{K}_i$. This detection can be performed using a batch technique, or by smoothness tests on each element using the ECM algorithm. The batch technique has quasi-linear complexity in the stored table size, and the complexity of the ECM algorithm to test an element for B-smoothness with $B = L_Q(1/3)$ is $L_Q(1/6)$. Regardless of the technique used, the complexity of detecting which of the stored elements are B_*-smoothness is $A\mathbf{P}_0$, which is also the complexity in memory of the algorithm. The *computation in each field* proceeds with a sparse linear algebra phase that costs $(B_0 + B_*)^2$, and an individual logarithm computation of negligible complexity compared to the two previous steps. The complexity of the *computation per field* step is $A\mathbf{P}_0 + (B_0 + B_*)^2$. As in many asymptotic analysis of NFS, we impose that smoothness detection and linear algebra have equal costs: $2\max(c_0, c_*) = c_A - c_{\mathcal{N}_0}/(3c_0)$. In short, the cost of the *computation per field* step is $L_Q(1/3, 2\max(c_0, c_*))$. To have enough equations for the linear algebra step, the number of expected relations that is $A\mathbf{P}_0\mathbf{P}_1 = L_Q\left(1/3, 2\max(c_0, c_*) - c_{\mathcal{N}_i}/(3c_*)\right)$ must be greater than the factor basis size $B_0 + B_* = L_Q(1/3, \max(c_0, c_*))$.

We choose parameters that minimize the complexity of the *computation per field* step under the conditions of having enough relations and equalizing the costs of smoothness detection and linear algebra, thus:

$$\text{minimize: } \max(c_0, c_*) \tag{2}$$

under conditions:

$$\max(c_0, c_*) - \frac{c_{\mathcal{N}_i}}{3c_*} \geq 0 \tag{3}$$

$$\text{and} \qquad 2\max(c_0, c_*) = c_A - c_{\mathcal{N}_0}/(3c_0) \tag{4}$$

where $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_i}$ are polynomials of degree at most one in c_A, and are independent with respect to c_0 and c_*. If the system above has a solution, then it has a solution with $c_0 = c_*$. Indeed, if $c_0 > c_*$, then replacing c_* by $\tilde{c_*} = c_0$ satisfies

Conditions (3) and (4), and provides the same minimum value given by (2). On the other hand, if $c_0 < c_*$, then replace c_0 by $\tilde{c_0} = c_*$ and replace c_A by $\tilde{c_A} < c_A$ to satisfy Equation (4). Indeed the number of stored elements $c_A - c_{\mathcal{N}_0}/(3c_0)$ increases as a function of c_A. Then Inequality (3) is still valid and the minimum value in (2) is unchanged. Hence, we take $B_0 = B_* = L_Q(1/3, c)$ and rewrite the system of equations:

$$\text{minimize: } c \tag{5}$$

under conditions:

$$3c^2 \geq c_{N_i} \tag{6}$$

$$\text{and} \qquad 6c^2 - 3c_A c + c_{N_0} = 0 \tag{7}$$

Such parameters provide the asymptotic complexity to solve the discrete logarithm in $\mathbb{F}_Q$ using Factory. In each variant, we provide a range of prime numbers p_i, that depends on Q, h, and f_0, for which the stored table can be used to solve the discrete logarithm in $\mathbb{F}_{p_i^n}$ with complexity $L_Q(1/3, 2c)$. To announce these complexities we need the following assumption:

Assumption 2 *Frobenius's Theorem 3.3 is still valid when considering large intervals of prime numbers instead of the set of all prime numbers.*

For instance, if the Galois group of f_0 is made of k elements out of which k_0 have κ in their cycle pattern, then under Assumption 2, f_0 admits an irreducible factor of degree κ modulo k_0/k of the prime numbers in a given large interval. This assumption is further supported by two experiments presented in Section 3.4.

4.1 NFS Factory

For the NFS setup with Tower, let $\mathcal{R} = \mathbb{Z}$, and consider f_0, f_1 two polynomials, defining $\mathcal{K}_i$ as in Diagram 4, for $i = 0, 1$. The norm of a sieve element $\phi \in \mathbb{Z}[X]$ once mapped to $\mathcal{K}_i$ is $\mathcal{N}_i := |\operatorname{Res}(\phi, f_i)| = \widetilde{O}\left(\|\phi\|_\infty^{\deg(f_i)} \|f_i\|_\infty^{\deg(\phi)}\right)$, for $i = 0, 1$.

Large characteristic finite fields Factory with GJL. We study the case where $p = L_Q(\alpha)$ with $2/3 < \alpha < 1$. The case of finite fields with $\alpha = 1$, i.e., prime finite fields, is detailed in [Bar13]. The *Generalized-Joux-Lercier* polynomial selection outputs irreducible and co-prime polynomials f_0 and $f_1 \in \mathbb{Z}[X]$ that share an irreducible factor of degree n modulo p. They have respective degrees $d + 1 > n$ and d, and respective coefficient sizes in $\widetilde{O}(1)$ and $\widetilde{O}\left(p^{n/(d+1)}\right)$. The sieve for the *one-off* step is performed in dimension 2, because the sieved elements $\phi \in \mathbb{Z}[X]$ are of the form $aX - b$, where, $a, b \in \mathbb{Z}$. This turns out to be the best choice for large characteristic finite fields. Hence, $\sqrt{A}$ bounds the coefficients of ϕ. Furthermore, we set a constant γ such that $d = 1/\gamma \left(\log(Q)/\log(\log(Q))\right)^{1/3}$. The norms of the sieve elements can be expressed as: $\mathcal{N}_0 = \widetilde{O}\left(A^{(d+1)/2}\right) = L_Q\left(2/3, c_A/(2\gamma)\right)$, and $\mathcal{N}_1 = \widetilde{O}\left(A^{d/2} Q^{1/(d+1)}\right) = L_Q\left(2/3, c_A/(2\gamma) + \gamma\right)$.

We detail the resolution of the system that minimizes Constraint (5), while verifying Conditions(6), and (7) in this variant. Thanks to Equation (7), we get $c_A = (12c^2\gamma)/(6c\gamma-1)$. Substituting c_A in Condition (6) we get $(-6c\gamma^2+(18c^3+1)\gamma-9c^2)/(6c\gamma-1) \geq 0$. The discriminant of the numerator is $324c^6-180c^3+1$, which has one negative real root and one positive real root, namely $c_0 = ((5+2\sqrt{6})/18)^{1/3}$. If $0 < c < c_0$, then the numerator of Condition (6) is negative for all γ, which implies that the denominator must be negative, contradicting the fact that $c_A > 0$. Therefore, c must be greater than or equal to c_0. In fact, $c = c_0$ is a valid solution. The solution to the system is given by:

$$c = \left(\frac{5+2\sqrt{6}}{18}\right)^{\frac{1}{3}} \approx 0.8193, \gamma = \frac{3+\sqrt{6}}{6c} \approx 1.1086, c_A = \frac{2\left(\sqrt{6}+3\right)}{\sqrt{6}+2} \times c \approx 2.0068$$

The complexity of the *one-off* step is $L_Q(1/3, c_A) \approx L_Q(1/3, 2.01)$, and the complexity of the *computation in each field* step is $L_Q(1/3, 2c) \approx L_Q(1/3, 1.64)$.

Suppose a *one-off* step was performed using the polynomial f_0 on a target size $Q = p^n$. Let p_i be a prime number that satisfies two conditions: first, f_0 has an irreducible factor of degree n modulo p_i, and second, $Q \leq p_i^n \leq QQ^{1/n}$. Using the *GJL* polynomial selection method, as explained in Section 3.2, we complete Diagram 4 with $\mathbb{F}_{p_i^n}$ at the bottom and $\mathcal{K}_i$ defined by a polynomial f_i of degree d with coefficient sizes in $\widetilde{O}\left(p_i^{n/(d+1)}\right)$. Considering that $p_i^n \leq QQ^{1/n}$, the norm of the stored elements, once mapped to $\mathcal{K}_i$, is expressed as $\mathcal{N}_i = \widetilde{O}\left(A^{d/2}Q^{1/(d+1)}Q^{1/(n(d+1))}\right)$. Moreover, $Q^{1/(n(d+1))} = p^{1/(d+1)} = L_Q(\alpha - 1/3)$, where $\alpha - 1/3 < 2/3$. Hence, $\mathcal{N}_i = L_Q\left(2/3, c_A/(2\gamma)+\gamma\right)$, which is the same asymptotic expression for $\mathcal{N}_1$. Based on the complexity analysis above and thanks to the stored table, we solve the discrete logarithm in $\mathbb{F}_{p_i^n}$ with a complexity of $L_Q(1/3, 2c) \approx L_Q(1/3, 1.64)$. The following theorem provides a summary of the complexity of NFS Factory in large characteristic finite fields.

Theorem 2. *Under Assumption 2 and classical NFS heuristics, for $Q = p^n$ where $p = L_Q(\alpha)$ with $2/3 < \alpha < 1$, for $\sigma_{f_0,n}$ of the prime numbers p_i such that $Q \leq p_i^n \leq QQ^{1/n}$. With a one time computation of complexity $L_Q\left(1/3, c_A\right)$ and a memory complexity of $L_Q\left(1/3, 2c\right)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q\left(1/3, 2c\right)$, where $2c = 2\left((5+2\sqrt{6})/18\right)^{1/3} \approx 1.64$, and $c_A = 2c\left(\sqrt{6}+3\right)/(\sqrt{6}+2) \approx 2.01$.*

For the sake of comparison, the complexity of NFS in large characteristic finite fields is $L_Q\left(1/3, (64/9)^{1/3}\right) \approx L_Q(1/3, 1.92)$, and the multiple variant MNFS, which is the state-of-the-art algorithm for general large characteristic finite fields, has a complexity of $L_Q(1/3, (2(46+13\sqrt{13})/27)^{1/3}) \approx L_Q(1/3, 1.90)$.

Boundary case: $\boldsymbol{\alpha = 2/3}$. We assume here that the characteristic p is at the boundary case between medium and large characteristic areas, meaning that $p = L_Q(2/3, c_p)$ for some positive constant c_p.

The boundary case with GJL. The asymptotic analysis in the large characteristic case applies as soon as $d = 1/\gamma\left(\log(Q)/\log(\log(Q))\right)^{1/3}$ is larger or equal than $n = 1/c_p\left(\log(Q)/\log(\log(Q))\right)^{1-\alpha}$, which is equivalent to $c_p \geq \gamma$ since $1-\alpha = 1/3$. For this range of finite fields, namely when $p = L_Q(2/3, c_p)$ with $c_p \geq \gamma$ and $\gamma = (3+\sqrt{6})/(6c) \approx 1.1086$, we get exactly the same asymptotic complexities as in Theorem 2. This should be compared with the asymptotic complexities of NFS and MNFS with GJL at the boundary case. These complexities are respectively equal to, $L_Q\left(1/3, (64/9)^{1/3}\right) \approx L_Q(1/3, 1.92)$ with the condition $c_p \geq (8/3)^{1/3} \approx 1.39$, and $L_Q(1/3, (2(46+13\sqrt{13})/27)^{1/3}) \approx L_Q(1/3, 1.90)$ with condition $c_p \geq ((7+2\sqrt{13})/6)^{1/3} \approx 1.33$.

The boundary case with Conjugation. The *Conjugation* polynomial selection method outputs irreducible and co-prime polynomials f_0 and $f_1 \in \mathbb{Z}[X]$ that share an irreducible factor of degree n modulo p. They have respective degrees $2n$ and n, and respective coefficient sizes in $\widetilde{O}(1)$ and $\widetilde{O}\left(\sqrt{p}\right)$. Let t a fixed integer that denotes the sieve dimension. The norms of the sieve elements can be expressed as: $\mathcal{N}_0 = \widetilde{O}\left(A^{(2n)/t}\right) = L_Q\left(2/3, 2c_A/(c_p t)\right)$, and $\mathcal{N}_1 = \widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}\right) = L_Q\left(2/3, c_A/(c_p t) + (t-1)c_p/2\right)$. The solution of the system that minimizes Constraint (5), while verifying Conditions (6), and (7) as function of c_p and t is c the largest real solution of equation:

$$18c_p t X^3 - 24X^2 - 3c_p^2 t(t-1)X + 2c_p(t-1) = 0 \tag{8}$$

and $c_A = 6c_p tc^2/(3c_p tc - 2)$. The asymptotic complexity of the *one-off* step is $L_Q(1/3, c_A)$, and the complexity of the *computation in each field* step is $L_Q(1/3, 2c)$.

Suppose a *one-off* step was performed with the polynomial f_0 on a target size $Q = p^n$. Let p_i be a prime number that satisfies two conditions: first, f_0 has an irreducible factor of degree n modulo p_i, and second, $p \leq p_i \leq pp^{o(1)}$. Diagram 4 is constructed thanks to Conjugation and $\mathcal{K}_i$ is defined by a polynomial f_i of degree n and with coefficient sizes in $\widetilde{O}\left(\sqrt{p_i}\right)$. The norms of the stored elements are $\mathcal{N}_i = \widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}p^{o(1)}\right)$. Since, $p^{o(1)}$ is negligible compared to any function in $L_Q(2/3)$, then $\mathcal{N}_i = L_Q\left(2/3, c_A/(c_p t) + (t-1)c_p/2\right)$. We get the following theorem:

Theorem 3. *Under Assumption 2 and classical NFS heuristics, for $Q = p^n$ where $p = L_Q(2/3, c_p)$ with c_p a positive constant, for $t \geq 2$ an integer, for $\sigma_{conj,f_0,n}$ of the prime numbers p_i such that $Q \leq p_i^n \leq QQ^{o(1)}$. With a one time computation of complexity $L_Q\left(1/3, c_A\right)$ and a memory complexity of $L_Q\left(1/3, 2c\right)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q\left(1/3, 2c\right)$, where c is the largest real solution of Equation (8), and $c_A = 6c_p tc^2/(3c_p tc - 2)$.*

This should be compared with the asymptotic complexities of NFS and MNFS with Conjugation at the boundary case. NFS has complexity $L_Q\left(1/3, 2c\right)$ with $c = 1/(c_p t) + \sqrt{1/(c_p t)^2 + c_p(t-1)/6}$, and MNFS has complexity $L_Q(1/3, 2\tilde{c})$

with $\tilde{c} = 1/(c_p t) + \sqrt{5/(9(c_p t)^2) + c_p(t-1)/6}$. The various asymptotic complexities of NFS, MNFS, and NFS Factory at boundary case, with both polynomial selection methods, GJL and Conjugation, are represented in Figure 8.

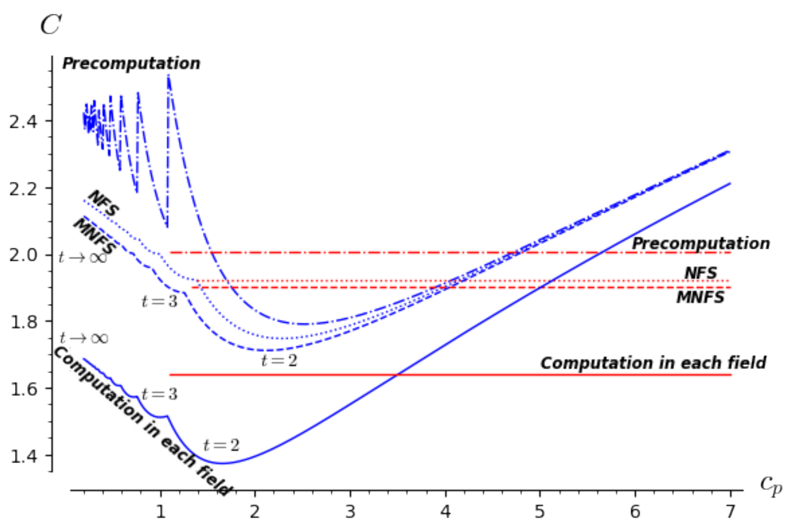

Fig. 8. Asymptotic complexities of NFS, MNFS, and NFS Factory for finite fields $\mathbb{F}_{p^n}$ with $p = L_{p^n}(2/3, c_p)$. The complexities are $L_{p^n}(1/3, c)$ and c is represented as a function of c_p in each case. Red lines correspond to algorithms that use GJL whereas blue curves are for Conjugation method. .

Medium characteristic finite fields Factory with Conjugation. Now we assume that p the characteristics are such that $p = L_Q(\alpha)$ with $1/3 < \alpha < 2/3$. The Conjugation polynomial selection outputs irreducible and coprime polynomials f_0 and $f_1 \in \mathbb{Z}[X]$ that share an irreducible factor of degree n modulo p. They have respective degrees $2n$ and n, and respective coefficient sizes in $\widetilde{O}(1)$ and $\widetilde{O}\left(\sqrt{p}\right)$. Denote t the sieving dimension that we take equal to $t = \delta n \left(\log(Q)/\log(\log(Q))\right)^{-1/3}$ for a positive constant δ. Hence, $A^{1/t}$ bounds the coefficients of the sieve elements. The norms of the sieve elements can be expressed as: $\mathcal{N}_0 = \widetilde{O}\left(A^{(2n)/t}\right) = L_Q\left(2/3, 2c_A/\delta\right)$, and $\mathcal{N}_1 = \widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}\right) = L_Q\left(2/3, c_A/\delta + \delta/2\right)$. The solution of the system that minimizes Constraint (5), while verifying Conditions (6), and (7) is given by: $c = \left(\frac{3+2\sqrt{2}}{9}\right)^{1/3} \approx 0.8652$, $\delta = \frac{2(\sqrt{2}+2)}{(\sqrt{2}+1)c} \approx 2.6308$, $c_A = \frac{2(\sqrt{2}+2)}{\sqrt{2}+1} \times c \approx 2.4471$

In other words, the asymptotic complexity of the *one-off* step is $L_Q(1/3, c_A) \approx L_Q(1/3, 2.45)$, and the asymptotic complexity of the *computation per field* step is $L_Q(1/3, 2c) \approx L_Q(1/3, 1.73)$.

Suppose a *one-off* step was performed with the polynomial f_0 on a target size Q. Let p_i be a prime number that satisfies two conditions: first, f_0 has an irreducible factor of degree n modulo p_i, and second, $Q \leq p_i^n \leq QQ^{1/t}$. Again we complete Diagram 4 thanks to the *Conjugation* polynomial selection method, as explained in Section 3.2 and 3.3. $\mathcal{K}_i$ defined by a polynomial f_i of degree n and with coefficient sizes in $\widetilde{O}\left(\sqrt{p_i}\right)$.

Since $p_i^n \leq QQ^{1/t}$, the coefficient sizes of f_i are in $\widetilde{O}\left(Q^{1/(2n)}Q^{1/(2tn)}\right)$, which implies that $\mathcal{N}_i = O\left(A^{n/t}Q^{(t-1)/(2n)}Q^{1/(2n)}\right)$. Furthermore, $Q^{1/(2n)} = L_Q(\alpha)$ and $\alpha < 2/3$. We get the norms $\mathcal{N}_i = L_Q\left(2/3, c_A/\delta + \delta/2\right)$. By the complexity analysis above, we deduce the following theorem that sums up the complexity of NFS Factory in medium characteristic finite fields.

Theorem 4. *Under Assumption 2 and classical NFS heuristics, for $Q = p^n$ where $p = L_Q(\alpha)$ with $1/3 < \alpha < 2/3$, for $\sigma_{conj,f_0,n}$ of prime numbers p_i such that $Q \leq p_i^n \leq QQ^{1/t}$. With a one time computation of complexity $L_Q\left(1/3, c_A\right)$ and a memory complexity of $L_Q\left(1/3, 2c\right)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q\left(1/3, 2c\right)$, where $2c = 2(3+2\sqrt{2})/9)^{1/3} \approx 1.73$, and $c_A = 2c(\sqrt{2}+2)/(\sqrt{2}+1) \approx 2.45$.*

Let us recall that the asymptotic complexites of NFS and MNFS, without Factory, in medium characteristic finite fields. NFS has asymptotic complexity $L_Q\left(1/3, (96/9)^{1/3}\right)$ which is approximmately $L_Q(1/3, 2.20)$, and MNFS has asymptotic complexity $L_Q(1/3, (8(9+4\sqrt{6})/15)^{1/3})$ that is approximately equal to $L_Q(1/3, 2.16)$.

4.2 TNFS Factory for medium characteristic finite fields

The Tower Number Field Sieve (TNFS) targets finite fields of medium characteristic with composite extension degree. Let $n = \eta\kappa$ a non trivial decomposition of the extension degree n. For the ease of presentation, we only consider the case where η and κ are co-prime. The general case is fundamentally the same. As in Diagram 4, h of degree η and coefficient sizes in $\widetilde{O}(1)$ defines $\mathcal{R}$, the polynomials f_0 and f_1 in $\mathbb{Z}[X]$ output by the *Conjugation* method define $\mathcal{K}_0$ and $\mathcal{K}_1$, and $\mathbb{F}_Q$ at the bottom with $Q = p^n$. The polynomials f_0 and f_1 have respective degrees 2κ and κ, and respective coefficient sizes in $\widetilde{O}(1)$ and $\widetilde{O}(\sqrt{p})$. Let $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q)))^{1/3}$ with c_κ a constant. The sieve is done over elements of the form $a(\iota)X - b(\iota) \in \mathcal{R}[X]$ with $a(\iota)$ and $b(\iota)$ in $\mathbb{Z}[\iota]$ of degree at most $\eta - 1$. As explained in Section 2.3, the norms are $\mathcal{N}_i(\phi) = |\operatorname{Res}_Y(\operatorname{Res}_X(a(Y)X - b(Y), f_i(X)), h(Y))| = \widetilde{O}(\|\phi\|_\infty^{\eta \deg(f_i)} \|f_i\|_\infty^{\eta} \|h\|_\infty^{(\eta-1)\deg(f_i)})$, for $i = 0, 1$. More precisely, since $\|\phi\|_\infty$ is bounded by $A^{1/(2\eta)}$, we get $\mathcal{N}_0 = \widetilde{O}(A^\kappa) = L_Q(2/3, c_A/c_\kappa)$ and $\mathcal{N}_1 = \widetilde{O}(A^{\kappa/2}Q^{1/(2\kappa)}) = L_Q(2/3, c_A/(2c_\kappa) + c_\kappa/2)$. The solution of the system that minimizes Constraint (5), while verifying Conditions (6), and (7) is:

$$c = \left(\frac{3+2\sqrt{2}}{18}\right)^{\frac{1}{3}} \approx 0.6867,\ \ c_\kappa = \frac{\sqrt{2}+2}{3c} \approx 1.6573,\ \ c_A = \frac{2(\sqrt{2}+2)}{\sqrt{2}+1} \times c \approx 1.9422$$

Suppose a *one-off* step was performed with the polynomials h and f_0 on a target size Q. Let p_i be a prime number that satisfies three conditions: first, h is irreducible modulo p_i, second, f_0 has an irreducible factor of degree κ modulo p_i, and third, $Q \leq p_i^n \leq QQ^{1/\eta}$. Thanks to *Conjugation* we can draw a diagram as in Figure 4 where $\mathcal{K}_i$ is defined by a polynomial f_i of degree κ and with coefficient sizes in $\widetilde{O}\left(\sqrt{p_i}\right)$. Since $p_i^n \leq QQ^{1/\eta}$, the coefficient sizes of f_i are in $\widetilde{O}\left(Q^{1/(2n)}Q^{1/(2\eta n)}\right)$, which implies that, $\mathcal{N}_i = \widetilde{O}(A^{\kappa/2}Q^{1/(2\kappa)}Q^{1/(2n)})$. Furthermore, $Q^{1/(2n)} = L_Q(\alpha)$ and $\alpha < 2/3$. Hence, the norms $\mathcal{N}_i = \widetilde{O}\left(A^{n/t}Q^{1/(2\kappa)}\right) = L_Q\left(2/3, c_A/(2c_\kappa) + c_\kappa/2\right)$. By the complexity analysis given in the above paragraph, we use the stored table and are able to solve the discrete logarithm in $\mathbb{F}_{p_i^n}$ with asymptotic complexity $L_Q(1/3, 2c) \approx L_Q(1/3, 1.37)$. To state the theorem that recapitulates the complexity of the TNFS Factory, we first need the following assumption:

Assumption 3 *The two events, h is irreducible modulo a prime number p and f_0 admits an irreducible factor of degree κ modulo the same prime p, are independent.*

Assumption 3, jointly with Assumption 2, are supported by the experiment given in Section 3.4.

Theorem 5. *Under Assumptions 2 and 3, and classical NFS heuristics, for $Q = p^n$ with $p = L_Q(\alpha)$ with $1/3 < \alpha < 2/3$, for $\sigma_{h,\eta}\sigma_{conj,f_0,\kappa}$ of the prime numbers p_i such that $Q \leq p_i^n \leq QQ^{1/\eta}$. With a one time computation of complexity $L_Q\left(1/3, c_A\right)$ and a memory complexity of $L_Q\left(1/3, 2c\right)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q\left(1/3, 2c\right)$, where $2c = 2(3+2\sqrt{2})/18)^{1/3} \approx 1.37$, and $c_A = 2c(\sqrt{2}+2)/(\sqrt{2}+1) \approx 1.94$.*

Note for the sake of comparison that the asymptotic complexities, of TNFS without Factory is $L_Q\left(1/3, (48/9)^{1/3}\right) \approx L_Q(1/3, 1.75)$, and of the Multiple Tower variant MTNFS is $L_Q(1/3, 2c)$, with $2c = 2(3/10 + 2\sqrt{2/3}/5)^{1/3}) \approx 1.71$.

4.3 SNFS Factory

The Special Number Field Sieve (SNFS) algorithm is designed for finite fields where the characteristic p is a sparse prime, meaning we can write $p = P(u)$, where P is a polynomial of small degree and coefficients, and u is an integer. Let p be such a prime, and λ be the degree of P. In the SNFS setup, f_0 and f_1 are given thanks to the *Joux-Pierrot* method. The polynomials have degrees n and λn, with coefficient sizes in $\widetilde{O}(p^{1/\lambda})$ and $\widetilde{O}(\log(\kappa)^\lambda)$. The norms of the sieve elements are in $\widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)})$ in one of the number fields and $\widetilde{O}(A^{\lambda n/t}\log(n)^{\lambda(t-1)})$ in the other, where t denotes the sieve dimension.

Large sparse characteristic finite fields. Here $p = L_Q(\alpha)$ with $2/3 < \alpha \le 1$. We consider the polynomials given by the first approach of Joux-Pierrot, (see Section 3.2). Hence, $\mathcal{N}_0 = \widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)})$ and $\mathcal{N}_1 = \widetilde{O}(A^{\lambda n/t}\log(n)^{\lambda(t-1)})$. The sieve dimension t is set to 2 and λ to $1/(c_\lambda n)(\log(Q)/\log\log(Q))^{1/3}$ with c_λ a constant. The norm of the sieve elements are $\mathcal{N}_0 = L_Q(2/3, c_\lambda)$ and $N_1 = L_Q(2/3, c_A/(2c_\lambda))$, since $\log(n)^\lambda$ is negligible in comparison with $L_Q(\alpha_p - 2/3)$, and $\alpha_p - 2/3 \le 1/3 < 2/3$. From Condition (7) we get $c_A = 2c + c_\lambda/(3c)$. Substituting c_A in Condition (6), we get $c_\lambda \ge 6c^2/(18c^3 - 1)$. For a given value c, it is best to choose the smallest possible value of c_λ in order to minimize c_A, hence c_λ is set to $c_\lambda = 6c^2/(18c^3 - 1)$. Moreover, c can be chosen close to zero. In return, c_A grows to infinity as c tends to zero. We choose c to minimize c_A:

$$c = \left(\frac{1}{3}\right)^{1/3} \approx 0.6934, \quad c_\lambda = 2\left(\frac{1}{3}\right)^{2/3} \approx 0.9615, \quad c_A = \frac{8}{3}\left(\frac{1}{3}\right)^{1/3} \approx 1.8490,$$

Suppose a *one-off* step that is performed with the polynomial f_0 on a target size Q. Let P_i be a polynomial of degree $d_i = \lambda + o(\lambda)$, with small coefficients, such that $P_i(u) = p_i$ is a prime number and f_0 is irreducible modulo p_i. Diagram 4 is completed with $\mathcal{K}_i$ defined by a polynomial f_i of degree $d_i n$ and with coefficient sizes in $\widetilde{O}(\log(n)^{d_i})$.

The norm of the stored elements are $\mathcal{N}_i = \widetilde{O}(A^{\lambda n/2}A^{o(\lambda n)}\log(n)^d)$, and $A^{o(\lambda n)}\log(n)^d$ is negligible compared to any function in the class $L_Q(2/3)$ as Q tends to infinity. Thus $\mathcal{N}_i = L_Q(2/3, c_A/(2c_\lambda))$. With this table of smooth elements in hand, we can solve the discrete logarithm in $\mathbb{F}_{p_i^n}$ with complexity $L_Q(1/3, 2c) \approx L_Q(1/3, 1.39)$. Finally, the asymptotic complexities of SNFS Factory in large characteristic finite fields are given by:

Theorem 6. *Let $\lambda = 1/(c_\lambda n)\cdot(\log(Q)/\log\log(Q))^{1/3}$ with $c_\lambda = 2\cdot(1/3)^{2/3}$, and let u an integer close to $Q^{1/(n\lambda)}$ as in the definition of f_0. Under Assumption 2 and classical NFS heuristics, for $\sigma_{f_0,n}$ of prime numbers p_i of the form $p_i = P_i(u)$ with P_i of a polynomial of degree $\lambda + o(\lambda)$ and coefficients in $\widetilde{O}(1)$. With a one time computation of complexity $L_Q(1/3, c_A)$ and a memory complexity of $L_Q(1/3, 2c)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q(1/3, 2c)$, where $2c = (8/3)^{1/3} \approx 1.39$, and $c_A = 8/3(1/3)^{1/3} \approx 1.85$.*

We recall the complexity of SNFS without Factory in large characteristic finite fields, that is $L_Q\left(1/3, (32/9)^{1/3}\right) \approx L_Q(1/3, 1.53)$.

Boundary case with sparse characteristic finite fields: $\boldsymbol{\alpha = 2/3}$. Our study indicates that coupling Factory with SNFS where $p = L_{p^n}(2/3, c_p)$ and c_p is a positive constant, does not always yield improved complexities. While reducing the complexity of the *main phase* (sieving and linear algebra in each fields), it leads to an increase in the complexity of the *individual logarithm* step. Consequently, for certain ranges of c_p, the resulting complexity becomes significantly large. We omit the analysis for this particular case.

Medium sparse characteristic finite fields. Here $p = L_Q(\alpha)$ with $1/3 < \alpha < 2/3$ and the polynomials f_0 and f_1 are chosen thanks to the second approach for the Joux-Pierrot method, see Section 3.2. Hence, $\mathcal{N}_0 = \widetilde{O}(A^{\lambda n/t}\log(n)^{\lambda(t-1)})$, and $\mathcal{N}_1 = \widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)})$. An integer $\lambda > 1$ is fixed and we set the sieve dimension t to $\delta n(\log(Q)/\log\log(Q))^{-1/3}$. The norm of the sieve elements are $\mathcal{N}_0 = L_Q(2/3, \lambda c_A/\delta)$, since $\log(n)^{\lambda(t-1)}$ is negligible in comparison with $L_Q(2/3)$, and $\mathcal{N}_1 = L_Q(2/3, c_A/\delta + \delta/\lambda)$. A solution of the usual system is:

$$\begin{aligned} c \geq \tilde{c} &= \left(\tfrac{\lambda+4+2\sqrt{2\lambda+4}}{9\lambda}\right)^{1/3},\ c_A = \tfrac{6c^2\delta}{3c\delta-\lambda} \\ \delta &= \tfrac{\lambda(9c^3+1)+\sqrt{-27\lambda c^3+\lambda^2(81c^6-18c^3+1)}}{6c} \end{aligned}$$

When $\lambda \in \{2, 3\}$, c is taken equal to $\tilde{c}$. However, when $\lambda \in \{4, 5\}$, c is taken larger than $\tilde{c}$ in order to keep the individual logarithm step negligible, as shown in Appendix A. Table 9 shows the values taken for c for various values of λ. The complexity of the *one-off* step is $L_Q(1/3, c_A)$, and the complexity of the *computation in each field* step is $L_Q(1/3, 2c)$.

Suppose a *one-off* step was performed with the polynomial $f_0 = P(X^n + R(X))$ on a target size Q. Let $Q^{1/(\lambda n)} \leq u_i \leq Q^{1/(\lambda n)}Q^{1/(\lambda nt)}$ an integer such that $p_i := P(u_i)$ is prime and $f_2 := X^n + R(X) - u_i$ is irreducible modulo p_i. This completes Diagram 4 with $\mathcal{K}_i$ defined with f_i. Moreover, f_i has coefficient sizes in $\widetilde{O}(Q^{1/(\lambda n)}Q^{1/(\lambda nt)})$, hence, the norms of the stored elements in $\mathcal{K}_i$ are $\mathcal{N}_i = \widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)}Q^{(t-1)/(\lambda nt)})$ with $Q^{(t-1)/(\lambda nt)}$ negligible with respect to $L_Q(2/3)$. In short $\mathcal{N}_i = L_Q(2/3, c_A/\delta + \delta/\lambda)$. The following theorem sums up the complexity of SNFS Factory in medium characteristic finite fields.

Theorem 7. *Let $\lambda > 1$ an integer and P a polynomial of degree λ and with coefficient sizes in $\widetilde{O}(1)$, and $t = \delta n(\log(Q)/\log\log(Q))^{-1/3}$ with the constant δ determined below. Under Assumption 1 and classical NFS heuristics, for $1/n$ of the integers $Q^{1/(\lambda n)} \leq u_i \leq Q^{1/(\lambda n)}Q^{1/(\lambda nt)}$ such that $p_i := P(u_i)$ is prime. With a one time computation of complexity $L_Q(1/3, c_A)$ and a memory complexity of $L_Q(1/3, 2c)$, the asymptotic complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q(1/3, 2c)$, where c and c_A are described above.*

We recall the complexity of SNFS (without Factory) in medium characteristic finite fields: $L_Q(1/3, (64/9 \times (\lambda+1)/\lambda)^{1/3})$. Table 9 shows the complexities of SNFS (with and without Factory) for medium characteristic finite fields for different explicit values of λ.

4.4 STNFS Factory

The Special Tower Number Field Sieve targets medium characteristic finite fields with sparse characteristic p, and composite extension degree $n = \kappa\eta$. Consider $p = P(u)$, where P is a polynomial of degree λ with small coefficients, and $u \approx p^{1/\lambda}$ is an integer. Again we assume that κ and η are co-prime. For the STNFS setup, let $h \in \mathbb{Z}[X]$ irreducible of degree η, and f_0 and f_1 output by the

λ	SNFS (without Factory)	one-off	computation per field
$\lambda = 2$	2.20	**2.45**	**1.73**
$\lambda = 3$	2.12	**2.50**	**1.58**
$\lambda = 4$	2.07	**2.16**	$2(1.1 \times \tilde{c}) \approx$ **1.64**
$\lambda = 5$	2.04	**2.15**	$2(1.1 \times \tilde{c}) \approx$ **1.57**

Table 9. Asymptotic complexities of the two steps of SNFS Factory and of SNFS in medium characteristic finite field, expressed as $L_Q(1/3, c)$. Only an approximation of c is given in this table. When $\lambda = 4$ or $\lambda = 5$, we adjust the parameters to keep the individual logarithm step negligible. $\tilde{c}$ is given above in Section 4.3.

second approach of Joux-Pierrot as in Section 3.2. They have respective degrees $\lambda\kappa$ and κ, and respective coefficient sizes in $\widetilde{O}(\log(\kappa)^\lambda)$ and $\widetilde{O}(p^{1/\lambda})$. We set $t = 2$ (this turns out to be the best choice), and $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q))^{1/3}$ for some constant c_κ. Thanks to the bound in Section 2.3, the sieve elements norms are $N_0 = \widetilde{O}(A^{\lambda\kappa/2}\log(\kappa)^\lambda) = L_Q(2/3, \lambda c_A/(2c_\kappa))$ and $\mathcal{N}_1 = \widetilde{O}(A^{\kappa/2}Q^{1/(\lambda\kappa)}) = L_Q(2/3, c_A/(2c_\kappa)+c_\kappa/\lambda)$. Eventually, the solution of the system related to Equations (5), (6), (7) is:

$$c \geq \tilde{c} = \left(\frac{\lambda+4+2\sqrt{2\lambda+4}}{18\lambda}\right)^{1/3}, c_A = \frac{12c^2c_\kappa}{6cc_\kappa-\lambda}$$
$$c_\kappa = \frac{\lambda(18c^3+1)+\sqrt{-144\lambda c^3+\lambda^2(324c^6-36c^3+1)}}{12c}.$$

When λ is equal to two, c is then equal to $\tilde{c}$. However, when λ is three, four, or five, c is taken larger than $\tilde{c}$ in order to keep the individual logarithm step negligible, as shown in Appendix A. Table 10 shows the values taken for c for various values of λ. The complexity of the *one-off* step is $L_Q(1/3, c_A)$, and the complexity of the *computation in each field* step is $L_Q(1/3, 2c)$.

Suppose a one-off step is performed with the polynomial $f_0 = P(X^\kappa + R(X))$ on a target size Q. Let $Q^{1/(\lambda n)} \leq u_i \leq Q^{1/(\lambda n)}Q^{1/(\lambda n\eta)}$ an integer such that $p_i := P(u_i)$ is prime and h and $f_i := X^n + R(X) - u_i$ are irreducible modulo p_i. This completes Diagram (4). Moreover, f_i has coefficient sizes in $\widetilde{O}(Q^{1/(\lambda n)}Q^{1/(\lambda n\eta)})$. The norms of the stored elements, when mapped to $\mathcal{K}_i$, are in $\mathcal{N}_i = \widetilde{O}(A^{\kappa/2}Q^{1/(\lambda\kappa)}Q^{1/(\lambda n)})$ where $Q^{1/(\lambda n)} = L_Q(\alpha)$ with $\alpha < 2/3$. In short $\mathcal{N}_i = L_Q(2/3, c_A/(2c_\kappa) + c_\kappa/\lambda)$. Thanks to the complexity analysis above, we know we can use the stored table to solve the discrete logarithm problem in $\mathbb{F}_{p_i^n}$ with complexity $L_Q(1/3, 2c)$. This gives:

Theorem 8 (STNFS Factory). *Let $\lambda > 1$ and integer and P a polynomial of degree λ and with small coefficients, and $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q))^{1/3}$ with the constant c_κ explicit below. Under Assumptions 1 and 2 and classical NFS heuristics, for $\sigma_h \times 1/\kappa$ of the integers $Q^{1/(\lambda n)} \leq u_i \leq Q^{1/(\lambda n)}Q^{1/(\lambda n\eta)}$ such that $p_i := P(u_i)$ is prime. With a one time computation of complexity $L_Q(1/3, c_A)$ and a memory complexity of $L_Q(1/3, 2c)$, the complexity of computing a discrete logarithm in $\mathbb{F}_{p_i^n}$ is $L_Q(1/3, 2c)$, where c, and c_A are described above.*

For the sake of comparison, we recall the asymptotic complexity of STNFS in medium characteristic finite fields, that is $L_Q(1/3, (32/9 \times (\lambda + 1)/\lambda)^{1/3})$. Table 10 recaps the asymptotic complexities of STNFS with and without Factory for different values of λ.

λ	STNFS without Factory	one-off	computation per field
$\lambda = 2$	1.75	**1.94**	**1.37**
$\lambda = 3$	1.68	**1.73**	$2(1.1 \times \tilde{c}) \approx$ **1.38**
$\lambda = 4$	1.64	**1.71**	$2(1.1 \times \tilde{c}) \approx$ **1.30**
$\lambda = 5$	1.62	**1.70**	$2(1.15 \times \tilde{c}) \approx$ **1.31**

Table 10. Asymptotic complexities approximations for the two steps of STNFS Factory and for STNFS, expressed as $L_Q(1/3, c)$. Only the approximation of c is given in this table. When λ is equal to 3, 4, or 5, we adjust the parameters to keep the individual logarithm step negligible. $\tilde{c}$ is given above in Section 4.4.

Conclusion of the asymptotic analysis

In Appendix A, we prove that the individual logarithm step is of negligible complexity compared to the complexity of the *computation per field step* in all the variants discussed in this section. Therefore, the complexities presented here represent the overall asymptotic complexities for Factory in each case.

The Factory variant offers the ability to reduce the complexity of NFS and its variants for a wide range of finite fields, requiring a one-time one-off. A trade-off between the *one-off* step and the *computation per field* step is possible. In our analysis, we choose to minimize the complexity of the computation in each field step at the expense of a larger one-time *one-off* step. Table 3 provides a summary of the complexities for NFS, all relevant variants included.

5 Estimation of practical cost

The purpose of this section is to compare computational cost estimates of TNFS and TNFS Factory on 1024-bit finite fields with extension degree equal to 6.

Setup. The factors of $n = 6$ are taken equal to $\eta = 2$ and $\kappa = 3$ as our analysis indicates that both TNFS and TNFS Factory perform the best with this choice. Denote A the sieve space, and Q the finite field size. Intuitively, relaying on the bound on the norms in Section 2.3, if one sets κ to 2, then the product of the norms of a sieve element in both number fields has order of magnitude $N_2 := A^3 Q^{1/2}$. On the other hand, the order of magnitude is equal to $N_3 := A^{9/2} Q^{1/6}$ if κ is set to 3. Hence, the ratio of the two order of magnitudes is $N_2/N_3 = A^{-3/2} Q^{1/3}$. For our example, Q is approximately 2^{1024}, and the sieve space A is certainly smaller than 2^{100}, hence, N_2/N_3 is greater than 2^{190}. In short, the choice $\eta = 2$ and $\kappa = 3$ gives smaller norms of the sieve elements. Let p be a prime number such that p^6 has roughly 1024 bits.

Polynomial selection. Previous records as in [MGP21] and [Rob22], suggest that Conjugation is the best method to select the polynomials in practice, see Section 3.2. Our analysis supports this suggestion and indicates that it should be the best method in practice for TNFS Factory as well. Let $h \in \mathbb{Z}[X]$ a degree 2 irreducible polynomial with small coefficients, and f_0 and f_1 in $\mathbb{Z}[X]$ of respective degrees 6 and 3, output by Conjugation. Based on the properties of the polynomials output by Conjugation, and on polynomials used in previous records, we make the assumption that $\|h\|_\infty = 1$, $\|f_0\|_\infty = 1$, and $\|f_1\|_\infty = \sqrt{p}$. For instance, in the previous record [MGP21] on $\mathbb{F}_{p_0^6}$, the polynomials had infinite norms respectively equal to $1, 1$, and approximately $1.0043 \times \sqrt{p_0}$. The number fields $\mathbb{Q}(\iota)$, $\mathcal{K}_0 := \mathbb{Q}(\iota, \alpha_0)$ and $\mathcal{K}_1 := \mathbb{Q}(\iota, \alpha_1)$, as in Diagram 4 are defined with h, f_0 and f_1, where ι, α_0 and α_1 are their respective roots.

One-off for TNFS Factory and relation collection for TNFS: The Special-q technique [Pol93]. The aim of the *one-off* step in TNFS Factory is to find B-smooth elements $\phi(x, \iota) = a(\iota) - b(\iota)\alpha_0$, where B is a smoothness bound, and a and b are polynomials of degree at most $\eta - 1$. The aim of the relation collection in TNFS is to find $a(\iota)$ and $b(\iota)$ of degrees at most $\eta - 1$ such that both $a(\iota) - b(\iota)\alpha_0$ and $a(\iota) - b(\iota)\alpha_1$ are B-smooth. In both cases, a *special-q* technique should be used to divide the search space into groups of elements that share a common factor in $\mathcal{K}_0$ (or $\mathcal{K}_i$ depending on where we put the *special-q*), that is an ideal $\mathfrak{q}$.

For TNFS Factory, given an ideal $\mathfrak{q}$ in say $\mathcal{O}_0$ the ring of integers of $\mathcal{K}_0$, a sieve algorithm is applied to detect which of the elements $(a(\iota) - b(\iota)\alpha_0)/\mathfrak{q}$ are B-smooth. Furthermore, the sieve algorithm only considers vectors (a, b) for which the 2η dimensional vector constitute of the coefficients of a and b has an Euclidean norm smaller or equal than a given radius R. If the Euclidean norm of (a, b) is written r, relaying on Section 2.3, we estimate the norm of $(a(\iota) - b(\iota)\alpha_i)$ by $N_i(r) := r^{\eta \deg(f_i)}\|f_i\|_\infty^\eta$, for $i = 0, 1$. Moreover, we denote $V_{2\eta}(r)$ the volume of the 2η-sphere of radius r, and ρ is the Dickman function. In short, we estimate the number of B-smooth elements among all the elements that are divisible by a special-q $\mathfrak{q}$ of norm q by:

$$\sum_{r=0}^{R-1} \left(V_{2\eta}(r+1) - V_{2\eta}(r)\right) \rho\left(\frac{\log(N_0(r)) - \log(q)}{\log(B)}\right).$$

Thus, we estimate the total number of B-smooth elements i.e., the elements to store, by the number of special-q considered times the number of smooth elements per special-q . Furthermore, we estimate the cost of the *one-off* step by the number of special-q considered times the cost of the sieve algorithm per special-q, that we approximate to $V_{2\eta}(R) \log\log(B)$.

For TNFS, a sieve is performed in both number fields to detect elements that are B-smooth in both number fields, i.e., elements that produce relations. Another alternative is to perform a sieve algorithm on the side of the special-q, and a batch algorithm (a product tree algorithm) to detect elements that are smooth on the other side. The number of expected relations for a special-q $\mathfrak{q}$ of size q, put on, say $\mathcal{K}_1$, is:

$$\sum_{r=0}^{R-1} \left(V_{2\eta}(r+1) - V_{2\eta}(r)\right) \rho\left(\frac{\log(N_0(r))}{\log(B)}\right) \rho\left(\frac{\log(N_1(r)) - \log(q)}{\log(B)}\right).$$

The total number of expected relations is the number of special-q times the number of expected relations per special-q . The cost of the relation collection step is the number of special-q times $2V_{2\eta}(R)\log\log(B)$ if a sieve is performed on both sides. If a sieve is performed on one side and a batch on the other, then we estimate the cost of the relation collection by the number of special-q times $V_{2\eta}(R)\log\log(B)$ plus a quasi-linear cost in the number of smooth elements output by the sieve.

computation per field for TNFS Factory and linear algebra for TNFS. The *computation per field* for TNFS Factory starts by detecting which of the stored elements from the *one-off* are B-smooth. This can be done using a product tree (batch technique) with cost quasi-linear in the number of the stored elements. The total number of relations produced is estimated by:

$$\sum_{r=0}^{R-1} \left(V_{2\eta}(r+1) - V_{2\eta}(r)\right) \rho\left(\frac{\log(N_0(r)) - \log(q)}{\log(B)}\right) \rho\left(\frac{\log(N_1(r))}{\log(B)}\right).$$

Then a sparse linear algebra phase computes the discrete logarithms of the factor basis for a cost that we estimate being equal to $(2\mathrm{Li}(B))^2$, where Li is the logarithmic integral function. Indeed, the factor basis size is taken to be approximately equal to $2\mathrm{Li}(B)$, since the logarithmic integral function evaluated on x estimates the number of prime numbers smaller than x. Moreover, the sparse linear algebra is quadratic in the factor basis size.

After the collection of relations, TNFS enters a sparse linear algebra phase with a cost that we estimate to $(2\mathrm{Li}(B))^2$ again.

We do not estimate the cost of the individual logarithm step for the *computation per field* step of Factory, nor for TNFS. This cost should be negligible, and roughly the same for both algorithms. Indeed, previous records and the asymptotic analysis in Appendix A support this statement.

Best parameters. For TNFS Factory, we search for the parameters that minimize the cost of the *computation per field*, under the condition of having enough relations, i.e., more relations than the factor basis size that is estimated to $2\mathrm{Li}(B)$. We denote $[sp - q_{min},\ sp - q_{max}]$ the range of the special-q space. The best parameters we found are:

$$R = 196, \quad sp - q_{min} \approx 2^{35.76}, \quad sp - q_{max} \approx 2^{38.32}, \quad B = 2^{33}.$$

As a consequence, the estimated cost of the *one-off* is $2^{67.77}$, and the estimated cost of the *computation per field* is $2^{60.84}$.

For TNFS without Factory, we search for the parameters that minimize the sum of the costs of the relation collection and the linear algebra steps, under the condition of having enough relations. Sieving on both sides gave the better estimated cost. In short, the best parameters we found are:

$$R = 138, \quad sp - q_{min} \approx 2^{33.74}, \quad sp - q_{max} \approx 2^{36.30}, \;\; B = 2^{35}.$$

The estimated cost of TNFS is therefor $2^{64.44}$.

What is the value of these estimations? Estimating the practical cost of NFS and its variants is a very difficult problem and we do not claim to get precise results in this section, far from it. A better approach would be computing "good" polynomials, sampling on special-q ideals and sieve elements, and trying to estimate the different costs by extrapolating the costs on these samples. This approach demands large efforts and is left for an independent future work.

On the one hand, experiments on recent records showed that our estimation of the norms sizes are rather accurate. On the other hand, we equalize many unknown constants to 1 when estimating the cost of the sieve and the sparse linear algebra algorithms. Nevertheless, by estimating the cost of TNFS and TNFS Factory in the same manner, we get costs that are comparable. In that sens, since $2^{67.77}/(2^{64.44} - 2^{60.84}) \approx 11$, our estimation suggests that when considering some tens of finite fields $\mathbb{F}_{p^6}$ of size 1024 bits, the TNFS Factory algorithm is more advantageous than applying the TNFS algorithm on each of the target finite fields.

6 Conclusion

The Factory variant for NFS brings a shift in the attacker's approach by targeting a specific size, such as 1024 bits, rather than a particular finite field. Through a costly one-time computation, the attacker gains the ability to efficiently target finite fields of the same size. Furthermore, the flexibility provided by the potential trade-off between the costs of the *one-time computation* and the *computation per field* enables accommodation of the available computation power and memory. This allows for better optimization based on the specific resources at hand. This technique can be leveraged to accelerate discrete logarithm computations for desired finite field sizes in software like Sage or Magma.

A drawback of Factory in practical usage is its subexponential memory complexity. The required table for storage grows subexponentially in size. However, if the attacker has prior knowledge of the specific finite fields being targeted (not just their size), it is possible to employ a batch technique for directly testing sieve elements for smoothness, as extensively explored in [BL14] for the *Factoring Factory* algorithm. In such cases, the memory requirements for Factory are equivalent to those for NFS and its variants.

References

ABD$^+$15. David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In *22nd ACM Conference on Computer and Communications Security*, October 2015.

AP22. Haetham Al Aswad and Cécile Pierrot. Individual discrete logarithm with sublattice reduction. *IACR Cryptol. ePrint Arch.*, 2022:912, 2022.

Bar13. Razvan Barbulescu. *Algorithmes de logarithmes discrets dans les corps finis*. PhD thesis, Université de Lorraine, 2013.

BB06. Anca Iuliana Bonciocat and Nicolae Ciprian Bonciocat. Some classes of irreducible polynomials. *Acta Arithmetica*, 123:349–360, 2006.

BGG$^+$20. Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé, and Paul Zimmermann. Comparing the difficulty of factorization and discrete logarithm: A 240-digit experiment. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2020, Part II*, LNCS, pages 62–91. Springer, Heidelberg, August 2020.

BGGM15. Razvan Barbulescu, Pierrick Gaudry, Aurore Guillevic, and François Morain. Improving NFS for the discrete logarithm problem in non-prime finite fields. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 129–155. Springer, Heidelberg, April 2015.

BGJT14. Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 1–16. Springer, Heidelberg, May 2014.

BL14. Daniel J. Bernstein and Tanja Lange. Batch NFS. In Antoine Joux and Amr M. Youssef, editors, *SAC 2014*, volume 8781 of *LNCS*, pages 38–58. Springer, Heidelberg, August 2014.

BLP93. J. P. Buhler, A. K. Lenstra, and C. Pomerance. Factoring integers with the number field sieve. In A. K. Lenstra and H. W. Lenstra, Jr., editors, *The development of the number field sieve*, volume 1554 of *Lecture Notes in Math.*, pages 50–94. Springer–Verlag, 1993.

BP14. Razvan Barbulescu and Cécile Pierrot. The multiple number field sieve for medium and high characteristic > finite fields. Cryptology ePrint Archive, Report 2014/147, 2014. `https://eprint.iacr.org/2014/147`.

CEP83. E. Rodney Canfield, Paul Erdös, and Carl Pomerance. On a problem of Oppenheim concerning “factorisatio numerorum”. *Journal of Number Theory*, 17(1):1–28, 1983.

CHM$^+$20. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward. Marlin: Preprocessing zksnarks with universal and updatable SRS. In *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 738–768. Springer, 2020.

Cop93. Don Coppersmith. Modifications to the number field sieve. *Journal of Cryptology*, 6:169–180, 1993.

Cop94. Don Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp.*, 62(205):333–350, 1994.

DM21. Gabrielle De Micheli. *Discrete Logarithm Cryptanalyses : Number Field Sieve and Lattice Tools for Side-Channel Attacks.* Theses, Université de Lorraine, May 2021.

FGHT17. Joshua Fried, Pierrick Gaudry, Nadia Heninger, and Emmanuel Thomé. A kilobit hidden SNFS discrete logarithm computation. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 202–231. Springer, Heidelberg, April / May 2017.

GKZ14. Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. Breaking '128-bit secure' supersingular binary curves - (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$). In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 126–145. Springer, Heidelberg, August 2014.

GMT16. Aurore Guillevic, François Morain, and Emmanuel Thomé. Solving discrete logarithms on a 170-bit MNT curve by pairing reduction. In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016*, volume 10532 of *LNCS*, pages 559–578. Springer, Heidelberg, August 2016.

Gor93. Daniel M. Gordon. Discrete logarithms in *GF(P)* using the number field sieve. *SIAM J. Discret. Math.*, 6(1):124–138, 1993.

Gré17. Laurent Grémy. Computations of discrete logarithms sorted by date, 2017. `https://dldb.loria.fr/`.

Gro16. Jens Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2016.

Gui18. Aurore Guillevic. Faster individual discrete logarithms in finite fields of composite extension degree. *CoRR*, abs/1809.06135, 2018.

Gui19. Aurore Guillevic. Faster individual discrete logarithms in finite fields of composite extension degree. *Mathematics of Computation*, 88(317):1273–1301, January 2019.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019. `https://eprint.iacr.org/2019/953`.

JLSV06. Antoine Joux, Reynald Lercier, Nigel Smart, and Frederik Vercauteren. The number field sieve in the medium prime case. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 326–344. Springer, Heidelberg, August 2006.

JP14. Antoine Joux and Cécile Pierrot. The special number field sieve in $\mathbb{F}_{p^n}$ - application to pairing-friendly constructions. In Zhenfu Cao and Fangguo Zhang, editors, *PAIRING 2013*, volume 8365 of *LNCS*, pages 45–61. Springer, Heidelberg, November 2014.

KB16. Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 543–571. Springer, Heidelberg, August 2016.

KBL14. Thorsten Kleinjung, Joppe W. Bos, and Arjen K. Lenstra. Mersenne factorization factory. 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*, volume 8873 of *Lecture Notes in Computer Science*, pages 358–377. Springer, 2014.

KJ17. Taechan Kim and Jinhyuck Jeong. Extended tower number field sieve with application to finite fields of arbitrary composite extension degree. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 388–408. Springer, Heidelberg, March 2017.

KR20. Biswajit Koley and A Satyanarayana Reddy. An irreducible class of polynomials over integers. *arXiv preprint arXiv:2004.00233*, 2020.

KW22. Thorsten Kleinjung and Benjamin Wesolowski. Discrete logarithms in quasi-polynomial time in finite fields of fixed characteristic. *Journal of the American Mathematical Society*, 35:581–624, 2022.

LLL82. A. K. Lenstra, H. W. Lenstra, and L. Lovasz. Factoring Polynomials with Rational Coefficients. *Mathematische Annalen*, 261:515–534, 1982.

LLMP90. Arjen K. Lenstra, Hendrik W. Lenstra Jr., Mark S. Manasse, and John M. Pollard. The number field sieve. In *22nd ACM STOC*, pages 564–572. ACM Press, May 1990.

Mat03. Dimitry Matyukhin. On asymptotic complexity of computing discrete logarithms over GF(p). *Discrete Mathematics and Applications*, 13:27–50, 2003.

MGP21. Gabrielle De Micheli, Pierrick Gaudry, and Cécile Pierrot. Lattice enumeration for tower NFS: A 521-bit discrete logarithm computation. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part I*, volume 13090 of *Lecture Notes in Computer Science*, pages 67–96. Springer, 2021.

Pie15. Cécile Pierrot. The multiple number field sieve with conjugation and generalized Joux-Lercier methods. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 156–170. Springer, Heidelberg, April 2015.

Pol93. John M. Pollard. The Lattice Sieve. In Arjen K. Lenstra and Hendrik W. Lenstra, editors, *The development of the number field sieve*, pages 43–49, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

Rob22. Oisín Robinson. An implementation of the extended tower number field sieve using 4d sieving in a box and a record computation in fp4. *CoRR*, abs/2212.04999, 2022.

SL96. Peter Stevenhagen and Hendrik Willem Lenstra. Chebotarëv and his density theorem. *The Mathematical Intelligencer*, 18:26–37, 1996.

SS16a. Palash Sarkar and Shashank Singh. A general polynomial selection method and new asymptotic complexities for the tower number field sieve algorithm. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 37–62. Springer, Heidelberg, December 2016.

SS16b. Palash Sarkar and Shashank Singh. New complexity trade-offs for the (multiple) number field sieve algorithm in non-prime fields. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 429–458. Springer, Heidelberg, May 2016.

SS19. Palash Sarkar and Shashank Singh. A unified polynomial selection method for the (tower) number field sieve algorithm. *Advances in Mathematics of Communications*, 13(3):435–455, 2019.

TCG19. The Trusted Computing Group. *Trusted Platform Module*, 2019. Latest version Nov. 2019. https://trustedcomputinggroup.org/resource/tpm-library-specification/.

A Complexity of the Individual Logarithm step in Factory

The individual logarithm step is the last one in NFS and its variants, and also the last one inside the *computation per field* phase in Factory, coupled or not with other variants. We prove in this Appendix that the complexity of *the individual logarithm* step is negligible compared to the rest of the *computation per field* step for all the variants we studied. Hence, the complexities announced and recapitulated in Table 3 are indeed the whole asymptotic complexities of the *computation per field* step. The individual logarithm step consists of two main steps: the smoothing and the descent step.

A.1 Smoothing step

The smoothing step consists in reducing the computation of the discrete logarithm of the target to the discrete logarithm of another element that is $\widetilde{B}$-smooth once lifted to one of the number fields, where $\widetilde{B} = L_Q(2/3, c_{\widetilde{B}}) > B$. The smoothing step was improved for finite fields of composite extension degree in [Gui18, AP22]. The following lemma recapitulates the complexity of the smoothing step for all the Factory variants:

Lemma 9. *In all NFS Factory variants, the running time of the smoothing step in $\mathbb{F}_{p^n}$ to output an element $\widetilde{B}$-smooth is $L_{p^n}(1/3, C = 3^{1/3}(23/27)^{2/3})$, where $\widetilde{B} = L_{p^n}(2/3, c_{\widetilde{B}})$ with $c_{\widetilde{B}} = (1/3)^{1/3}(27/23)^{2/3}$. The approximated values are: $C \approx 1.30$, and $c_{\widetilde{B}} \approx 0.77$.*

Proof. The lemma is a direct consequence of Corollary 6.4 and Corollary 6.5 in [Gui18], where substituting e and d by 1 is valid for all our Factory variants.

The complexity of the smoothing step is thus negligible compared to the complexity of the *computation per field* step in all Factory variants.

A.2 Descent step

This paragraph is inspired from [Bar13], where the descent step is presented for NFS Factory in prime finite fields. We adapt the idea to other characteristic sizes and to the different variants coupled with Factory.

After the smoothing step, the target is $\widetilde{B}$-smooth with $\widetilde{B} = L_Q(2/3, c_{\widetilde{B}}) > B$, where $c_{\widetilde{B}}$ is as in Lemma 9. Thanks to the previous steps, we know the virtual logarithms of the prime ideals in $\mathcal{K}_0$ that are both factors of the target and of norm smaller than B. It remains to compute the virtual logarithms of the prime ideals in $\mathcal{K}_0$ that are factors of the target but of norm between B and $\widetilde{B}$. Let $\mathfrak{q}$ be such a prime ideal, that is of degree one, and denote q its norm. Define the special-q lattice $\mathcal{L}_\mathfrak{q}$ of dimension 2η over $\mathbb{Z}$, and of determinant q, that corresponds to the elements $(a(\iota), b(\iota))$ such that $(a(\iota) - b(\iota)\alpha_0)$ is divisible by $\mathfrak{q}$ in $\mathcal{K}_0$. Using the LLL algorithm, compute $(u_0, \ldots, u_{2\eta-1})$ a basis of $\mathcal{L}_\mathfrak{q}$ where $\|u_i\|_\infty = \widetilde{O}(p^{1/(2\eta)})$ for $i = 0, \ldots, 2\eta - 1$. Let $\xi \in (0, 1)$ a positive real number smaller than one, to be determined later. The first step of the descent step consists in finding $(a(\iota), b(\iota)) \in \mathcal{L}_\mathfrak{q}$ such that :

- $\frac{\mathcal{N}_0(b(\iota)-a(\iota)\alpha_0)}{q}$ is q^ξ-smooth.
- and $\mathcal{N}_i\left(b(\iota) - a(\iota)\alpha_1\right)$ is q^ξ-smooth.

This permits to express the virtual logarithm of $\mathfrak{q}$ as a linear combination of virtual logarithms of prime ideals of norms smaller than q^ξ. To recover the virtual logarithm of $\mathfrak{q}$, it is sufficient to repeat the process on each of the ideals in the linear combination until they are all in the factor basis.

We start by proving that the first step of the descent, i.e., finding $(a(\iota), b(\iota))$ as above, is the dominant step of the descent in terms of complexity. To descend the ideal $\mathfrak{q}$ to the factor basis, we construct a tree where the root is $\mathfrak{q}$ and the leaves are ideals in the factor basis. Each ideal that descends due to a pair $(a(\iota), b(\iota))$ introduces at most $\log_2(\mathcal{N}_0(b(\iota)-a(\iota)\alpha_0)+\log_2(\mathcal{N}_i(b(\iota)-a(\iota)\alpha_1)$ new nodes. By Corollary 6.4 in [Gui18], both norms are smaller than Q. Hence, the arity of the tree is less than $2\log_2 Q$, and its depth is smaller than the smallest integer k such that $\xi^k \log \widetilde{B} \leq \log B$. Hence, $k = O((\log\log Q))$. The number of nodes in the tree is less than $(2\log_2(Q))^k = \exp(O(\log\log(Q)^2))$. Denote $\mathcal{C}$ the complexity of the first descent of $\mathfrak{q}$. We prove in the following paragraph that $\mathcal{C} = L_Q(1/3)$. Hence, the complexity of descending $\mathfrak{q}$ to the factor basis is dominated by $\exp(O(\log\log(Q)^2))\cdot\mathcal{C} = \mathcal{C}$. This process is applied on all the prime factors of the target that are not in the factor basis, their number is in $O(\log Q)$. In short, the complexity of the descent step is the complexity of descending $\mathfrak{q}$, that is the complexity of finding $(a(\iota), b(\iota))$ as described above.

Complexity of the descent step for NFS Factory and its variants. For $\mu = (\mu_0, \ldots, \mu_{2\eta-1})$ of infinite norm S, we look for "good" $(a(\iota), b(\iota))$ of the form $\mu_0 u_0 + \ldots \mu_{2\eta-1}u_{2\eta-1}$, either by sieving or ECM tests. Hence, $\|(a(\iota), b(\iota))\|_\infty = \widetilde{O}(Sq^{1/(2\eta)})$. We take $S^{2\eta} := L_Q(1/3, s)$ for a positive s to be chosen. From the bound in Section 2.3, we get $\mathcal{N}_i(a(\iota)-b(\iota)\alpha_i) = \widetilde{O}((S^{2\eta})^{\deg(f_i)/2}\|f_i\|_\infty^\eta q^{\deg(f_i)/2})$, for $i = 0, 1$. We assume the two following usual heuristics. The probability of each of the norms being q^ξ-smooth is the same as for a random integer of the same size, and the q^ξ-smoothness probability of both norms are independent. Under these assumptions, the probability that both norms are q^ξ-smooth is greater than the probability of a random integer of size the product of the norms being

q^ξ-smooth. Besides, the product of the norms divided by q is of size

$$N = \widetilde{O}\left((S^{2\eta})^{(\deg(f)+\deg(f_1))/2}\|f\|_\infty^\eta \|f_1\|_\infty^\eta q^{(\deg(f)+\deg(f_1))/2-1}\right).$$

Denote $q = L_Q(\alpha_q, c_q)$, where $1/3 \leq \alpha_q \leq 2/3$, with $c_q > c$ if $\alpha_q = 1/3$, and $c_q < c_{\widetilde{B}}$ if $\alpha_q = 2/3$, since $B < q < \widetilde{B}$. Hence, $q^\xi = L_Q(\alpha_q, \xi c_q)$. The complexity of a q^ξ-smoothness test by ECM is $L_Q(\alpha_q/2, (2\alpha_q \xi c_q)^{1/2})$. It is negligible compared to $L_Q(1/3)$ whenever $\alpha_q < 2/3$, and is equal to $L_Q(1/3, (4\xi c_q/3)^{1/2})$ if $\alpha_q = 2/3$.

Large characteristic descent step for Factory. Plugging the properties of the polynomials output by the *GJL* polynomial selection, with $\eta = 1$, we get $N = \widetilde{O}((S^2)^{(2d+1)/2} Q^{1/(d+1)} q^{(2d+1)/2-1}$. Hence, $N = L_Q(2/3, s/\gamma + \gamma + c_q/\gamma)$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, c_q/\gamma)$ if $\alpha_q > 1/3$. The asymptotic complexity of the descent step is the inverse of the probability of N being q^ξ-smooth (see Section 2.3) times the cost of ECM. Thus this complexity is:

- $L_Q\left(\frac{1}{3}, \frac{s}{3\gamma\xi c_q} + \frac{\gamma}{3\xi c_q} + \frac{1}{3\xi\gamma}\right)$, if $\alpha_q = \frac{1}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{3\xi\gamma}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{3\xi\gamma} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

When $\mathfrak{q}$ is small, i.e., $\alpha_q = 1/3$, the complexity of the descent grows as q^ξ decreases, it is maximal when $\xi c_q = c$. Furthermore, the space of search of (a, b) has to be equal to the inverse of the probability of N being q^ξ-smooth, which translates into $s = s/(3\gamma\xi c) + \gamma/(3\xi c) + 1/(3\xi\gamma)$ after equalizing ξc_q and c. Thus, $s = (\gamma^2\xi + c)/((3c\gamma - 1)\xi)$. Taking for instance $\xi = 0.999$, we get the complexity of the descent in approximately $L_Q(1/3, 1.19)$, which is negligible compared to the smoothing step. The complexity of the descent when α_q is between $1/3$ and $2/3$ is upper bounded by the complexity when $\mathfrak{q}$ is of large size, i.e., $\alpha_q = 2/3$. In this last case, the complexity grows as q grows, it is maximal when $c_q = c_{\widetilde{B}}$. Hence, the complexity is upper bounded by $L_Q(1/3, 1/(3\xi\gamma) + (4c_{\widetilde{B}}\xi/3)^{1/2})$. By minimizing the last quantity in ξ, we get $\xi = 1/(3c_{\widetilde{B}}\gamma^2)^{1/3}$. In short, the complexity of descending $\mathfrak{q}$ is approximately $L_Q(1.3, 1.28)$, which is also negligible compared to the smoothing step. In conclusion, the complexity of the descent step in NFS Factory for large characteristic finite fields is negligible compared to the complexity of the smoothing step.

The analysis giving the best parameter choices for the other variants follow the same idea. We omit the optimization details. Table 11 recapitulates the asymptotic complexities for the individual logarithm step in all the variants.

Boundary case descent step for Factory. We target finite fields $\mathbb{F}_{p_i^n}$ where $p = L_Q(2/3, c_p)$, with c_p a positive constant. When the polynomial selection method used is GJL, the complexity analysis of the descent step is the same as for NFS Factory in large characteristic. It is negligible compared to the smoothing step.

Algorithm	Characteristic	Smoothing	Descent	computation per field
NFS Factory	Large	1.30	1.28	**1.64**
	Boundary case	Figure 12		
	Medium	1.30	1.43	**1.73**
TNFS Factory	Medium	1.30	1.28	**1.37**
SNFS	Large	1.30	1.06	**1.39**
	Medium	Table 13		
STNFS Factory	Medium	Table 14		

Table 11. Asymptotic complexities of the individual logarithm step and the *computation per field step* in NFS Factory and its variants. This table recap an approximation of c when the complexities are expressed as $L_Q(1/3, c)$. For NFS at the boundary case, the complexity depends on the finite field size. We refer to a figure plotting these complexities. For SNFS in medium characteristic (with or without Tower), the complexities depend on an integer λ. We refer to tables giving the complexities for various values of λ.

When using Conjugation, instead of looking for a "good" (a,b) in $\mathcal{L}_\mathfrak{q}$, we look for a "good" vector of dimension $\tilde{t}$, where $\tilde{t}$ is a positive integer greater than or equal to two. Hence, $\eta = 1$, the dimension of $\mathcal{L}_\mathfrak{q}$ is $\tilde{t}$ and its determinant is q. We need to adapt the formula given for N at the beginning of this Appendix and use instead the formula at the beginning of Section 4.1 with the properties of the polynomials output by Conjugation. In short, taking $S^{\tilde{t}} = L_Q(1/3, s)$, we get $N = \widetilde{O}((S^{\tilde{t}})^{3n/\tilde{t}} Q^{(\tilde{t}-1)/(2n)} q^{3n/\tilde{t}-1})$. Hence $N = L_Q(2/3, 3s/(\tilde{t}c_p) + (\tilde{t}-1)c_p/2 + 3c_q/(\tilde{t}c_p))$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, 3c_q/(\tilde{t}c_p))$ if $\alpha_q > 1/3$. The complexity of the descent step is then:

- $L_Q\left(\frac{1}{3}, \frac{s}{\xi c_q c_p \tilde{t}} + \frac{(\tilde{t}-1)c_p}{6\xi c_q} + \frac{1}{\xi \tilde{t} c_p}\right)$, if $\alpha_q = \frac{1}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{\xi \tilde{t} c_p}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{\xi \tilde{t} c_p} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

Figure 12 plots the asymptotic complexities of different parts of Factory: the smoothing step, the descent step for both small and large $\mathfrak{q}$, and the computation in each step. We see that both the descent step and the smoothing step are negligible with regard to the computation per field.

Medium characteristic descent step for Factory. Here the analysis is quite close from the one at the boundary case for Conjugation. Again, we look for a "good" vector of dimension $\tilde{t}$, where $\tilde{t}$ is taken equal to $\tilde{\delta} n(\log(Q)/\log\log(Q))^{-1/3}$. Hence, $\eta = 1$, the dimension of $\mathcal{L}_\mathfrak{q}$ is $\tilde{t}$ and its determinant is q. Taking $S^{\tilde{t}} = L_Q(1/3, s)$, we get $N = \widetilde{O}((S^{\tilde{t}})^{3n/\tilde{t}} Q^{(\tilde{t}-1)/(2n)} q^{3n/\tilde{t}-1})$. Hence we can write the norm $N = L_Q(2/3, 3s/\tilde{\delta} + \tilde{\delta}/2 + 3c_q/\tilde{\delta})$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, 3c_q/\tilde{\delta})$ if $\alpha_q > 1/3$. The asymptotic complexity of the descent step depends on the size of $\mathfrak{q}$, it is:

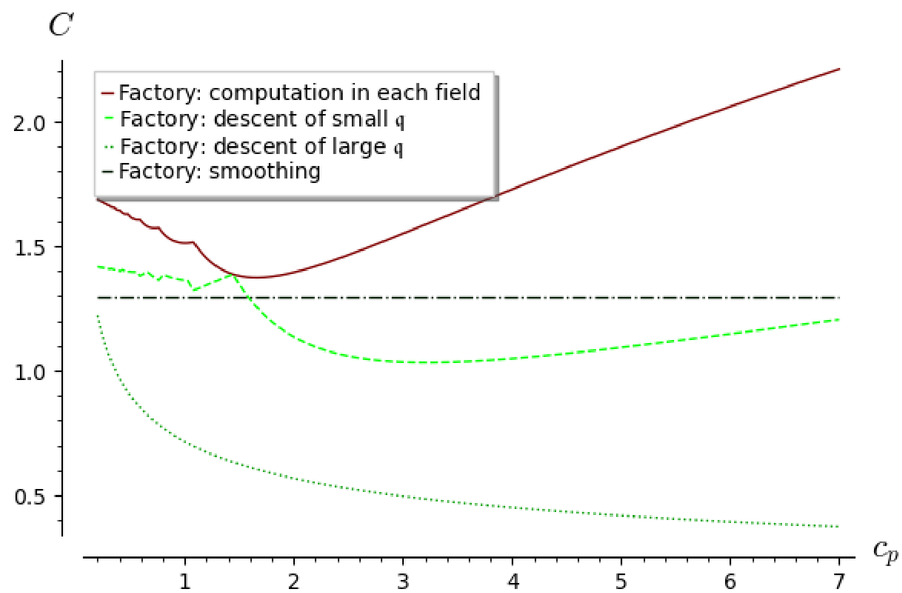

Fig. 12. Asymptotic complexities of some steps inside NFS Factory at the boundary case. Target finite fields have characteristic p such that $p = L_{p^n}(2/3, c_p)$. This graph shows how c varies as a function of c_p when the complexities are expressed as $L_{p^n}(1/3, c)$.

- $L_Q\left(\frac{1}{3}, \frac{s}{\xi c_q \tilde{\delta}} + \frac{\tilde{\delta}}{6\xi c_q} + \frac{1}{\xi\tilde{\delta}}\right)$, if $\alpha_q = \frac{1}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{\xi\tilde{\delta}}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{\xi\tilde{\delta}} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

The hardest $\mathfrak{q}$ to descend is the one of small size with a complexity in approximately $L_Q(1/3, 1.43)$. The descent step has a complexity that is dominant compared to the smoothness step, but negligible compared to the *computation per field step*.

Medium characteristic descent step for TNFS Factory. We consider Conjugation for the polynomial selection. We get $N = \widetilde{O}((S^{2\eta})^{3\kappa/2} Q^{1/(2\kappa)} q^{3\kappa/2-1})$. Hence, $N = L_Q(2/3, 3s/(2c_\kappa)+c_\kappa/2+3c_q/(2c_\kappa))$ if $\alpha_q = 1/3$ and $N = L_Q(2/3, 3c_q/(2c_\kappa))$ if $\alpha_q > 1/3$. The complexity of the descent step is:

- $L_Q\left(\frac{1}{3}, \frac{s}{2\xi c_q c_\kappa} + \frac{c_\kappa}{6\xi c_q} + \frac{1}{2\xi c_\kappa}\right)$, if $\alpha_q = \frac{1}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{1}{2\xi c_\kappa}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.

– $L_Q\left(\frac{1}{3}, \frac{1}{2\xi c_\kappa} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

The hardest $\mathfrak{q}$ to descend is the one of large size with a complexity in approximately $L_Q(1/3, 1.28)$, which is negligible compared to the complexity of the smoothness step.

Large characteristic descent step for SNFS Factory. Plugging the properties of the polynomials given by the Joux-Pierrot polynomial selection, we get the norm $N = \widetilde{O}((S^2)^{n(\lambda+1)/2}Q^{1/(\lambda n)}q^{n(\lambda+1)/2-1})$. Hence, $N = L_Q(2/3, s/(2c_\lambda) + c_\lambda + c_q/(2\lambda))$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, c_q/(2\lambda))$ if $\alpha_q > 1/3$. The complexity of the descent step is:

– $L_Q\left(\frac{1}{3}, \frac{s}{6\xi c_q c_\lambda} + \frac{c_\lambda}{3\xi c_q} + \frac{1}{6\xi c_\lambda}\right)$, if $\alpha_q = \frac{1}{3}$.
– $L_Q\left(\frac{1}{3}, \frac{1}{6\xi c_\lambda}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
– $L_Q\left(\frac{1}{3}, \frac{1}{6\xi c_\lambda} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

The hardest $\mathfrak{q}$ to descend is the one of large size with a complexity in approximately $L_Q(1/3, 1.06)$, which is negligible compared to the complexity of the smoothness step.

Medium characteristic descent step for SNFS Factory. We look for a "good" vector of dimension $\tilde{t}$, where $\tilde{t}$ is taken equal to $\tilde{\delta}n(\log(Q)/\log\log(Q))^{-1/3}$. Therefor, $\eta = 1$, the dimension of $\mathcal{L}_\mathfrak{q}$ is $\tilde{t}$ and its determinant is q. We use the formula for N of Section 4.1, with the properties of the polynomials output by the Joux-Pierrot method. Writting $S^{\tilde{t}} = L_Q(1/3, s)$, we obtain $N = \widetilde{O}((S^{\tilde{t}})^{n(\lambda+1)/\tilde{t}}Q^{(\tilde{t}-1)/(\lambda n)}q^{n(\lambda+1)/\tilde{t}-1})$. Hence, $N = L_Q(2/3, s(\lambda+1)/\tilde{\delta} + \tilde{\delta}/\lambda + (\lambda+1)c_q/\tilde{\delta})$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, (\lambda+1)c_q/\tilde{\delta})$ if $\alpha_q > 1/3$. The complexity of the descent step is:

– $L_Q\left(\frac{1}{3}, \frac{s(\lambda+1)}{3\xi c_q\tilde{\delta}} + \frac{\tilde{\delta}}{3\xi c_q\lambda} + \frac{\lambda+1}{3\xi\tilde{\delta}}\right)$, if $\alpha_q = \frac{1}{3}$.
– $L_Q\left(\frac{1}{3}, \frac{\lambda+1}{3\xi\tilde{\delta}}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
– $L_Q\left(\frac{1}{3}, \frac{\lambda+1}{3\xi\tilde{\delta}} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

As previously, the hardest $\mathfrak{q}$ to descend is the one of large size. Table 13 presents approximate values of the complexity for various values of λ. The complexity of the descent step is always dominant compared to the smoothing step, but still negligible compared to the *computation per field* step.

Medium characteristic descent step for STNFS Factory. With Joux-Pierrot selection and the usual notations, we get, $N = \widetilde{O}((S^{2\eta})^{\kappa(\lambda+1)/2}Q^{1/(\lambda\kappa)}q^{\kappa(\lambda+1)/2-1})$. Hence, $N = L_Q(2/3, (\lambda+1)s/(2c_\kappa) + c_\kappa/\lambda + (\lambda+1)c_q/(2c_\kappa))$ if $\alpha_q = 1/3$, and $N = L_Q(\alpha_q + 1/3, (\lambda+1)c_q/(2c_\kappa))$ if $\alpha_q > 1/3$. The complexity of the descent step depends on λ, it is:

λ	Smoothing	Descent	computation per field
$\lambda = 2$	1.43	1.30	**1.73**
$\lambda = 3$	1.46	1.30	**1.58**
$\lambda = 4$	1.33	1.30	**1.64**
$\lambda = 5$	1.36	1.30	**1.57**

Table 13. Asymptotic complexities for different part of medium characteristic SNFS Factory. These complexities are expressed as $L_Q(1/3, c)$ and only an approximation of c is given. The individual logarithm phase, that consists of the smoothing step and the descent step, is always negligible with regard to the other steps in the computation per field.

- $L_Q\left(\frac{1}{3}, \frac{s(\lambda+1)}{6\xi c_q c_\kappa} + \frac{c_\kappa}{3\xi c_q \lambda} + \frac{\lambda+1}{6\xi c_\kappa}\right)$, if $\alpha_q = \frac{1}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{\lambda+1}{6\xi c_\kappa}\right)$, if $\frac{1}{3} < \alpha_q < \frac{2}{3}$.
- $L_Q\left(\frac{1}{3}, \frac{\lambda+1}{6\xi c_\kappa} + \sqrt{\frac{4\xi c_q}{3}}\right)$, if $\alpha_q = \frac{2}{3}$.

The hardest $\mathfrak{q}$ to descend is the one of large size. Table 14 presents approximate values of the complexity for different small values of λ. We see that the asymptotic complexity of both the descent step is negligible compared to the complexity of the smoothing step. Note that both the smoothing and the descent are negligible with regard to the computation in each field when λ is lower of equal to 4, but when $\lambda = 5$ the smoothing step starts to be dominant.

λ	Descent	Smoothing	computation per field
$\lambda = 2$	1.26	1.30	**1.37**
$\lambda = 3$	1.04	1.30	**1.38**
$\lambda = 4$	1.06	1.30	**1.30**
$\lambda = 5$	1.08	1.30	**1.31**

Table 14. Asymptotic complexities for different part of medium characteristic STNFS Factory. These complexities are expressed as $L_Q(1/3, c)$ and only an approximation of c is given. The dominant step is indicated in bold.