IACR Communications in Cryptology
Vol. 0, No. 0, 38 pages.

Discrete Logarithm Factory

Haetham Al Aswad[a], Emmanuel Thomé and Cécile Pierrot

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

Abstract. The Number Field Sieve and its variants are the best algorithms to solve the discrete logarithm problem in finite fields (except for the weak small characteristic case). 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. A precomputation, solely dependent on an approximate finite field size and an extension degree, allows to efficiently compute discrete logarithms in a constant proportion of the finite fields of the given approximate size and extension degree. We combine this idea with two other variants of NFS, namely the tower and special variant. This combination improves the asymptotic complexity. We also notice that combining our approach with the MNFS variant would be an unnecessary complication as all the potential gain of MNFS is subsumed by our Factory variant anyway. Furthermore, we demonstrate how Chebotarev's density theorem allows to compute the density of finite fields that can be solved with a given precomputation. Finally, we provide experimental data in order to assess the practical reach of our approach.

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, GKZ18, 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]. Currently, the most efficient algorithms to compute discrete logarithm in medium or large characteristic finite fields is still (a variant of) NFS. Numerous variants exist, depending on the sub-case, but they all compute discrete logarithms in finite fields

E-mail: haetham.al-aswad@inria.fr (Haetham Al Aswad), emmanuel.thome@inria.fr (Emmanuel Thomé), cecile.pierrot@inria.fr (Cécile Pierrot)

[a]The author acknowledges support of the grant funded by *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.

Table 1: Variants of NFS and their asymptotic complexities. All 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 finite fields that satisfy the constraint expressed in bold above it. Some complexities are given as lower bounds, which are reached when the input satisfies some additional constraints. The complexities of SNFS and STNFS for medium characteristic are functions of another parameter λ that is omitted here.

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

in time $L_{p^n}(1/3, c)$ for some constant $0 < c < 2.3$ that depends on the precise sub-case. The special 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 SNFS resulting in the STNFS algorithm [KB16, KJ17]. See Table 1 for a summary. In the boundary case between medium and large characteristic, complexities are functions of p and harder to express than with a simple $L(1/3, c)$ expression with constant c. See later Figure 7 and expressions given in §A.1 for this particular parameter range.

The general framework is common to all variants of NFS. First one 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 difference 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: the individual logarithm step.

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 [DGP21] 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

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

in [DGP21]. 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].

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

Finite field	Bitsize of p^n	Year	Team and work
$\mathbb{F}_p$	795	2019	Boudot, Gaudry, Guillevic, Heninger, Thomé, Zimmermann [BGG+20]
$\mathbb{F}_{p^2}$	595	2015	Barbulescu, Gaudry, Guillevic, Morain [BGGM15]
$\mathbb{F}_{p^3}$	593	2016	Gaudry, Guillevic, Morain [GGM16]
$\mathbb{F}_{p^4}$	512	2022	Robinson [Rob22]
$\mathbb{F}_{p^5}$	324	2017	Grémy, Guillevic, Morain [GGM17]
$\mathbb{F}_{p^6}$	521	2021	De Micheli, Gaudry, Pierrot [DGP21]
$\mathbb{F}_{p^{12}}$	203	2013	Hayasaka, Aoki, Kobayashi,Takagi [HAKT14]

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 field, not on the specific target element whose logarithm is desired. This is how the Logjam attack [ABD+15] was carried out, by precomputing 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 finite fields, which have the same extension degree and the same given bitsize, rather than using NFS (or its variants) on each field separately? In particular, is there a configuration where some kind of precomputation would be beneficial? Whether or not the precise set of fields is known in advance, such an attack scenario is referred to as a Factory-like computation, owing to the state-of-the-art algorithms described below. Most of this article assumes that the target finite fields are *not* known in advance.

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, §7.2].

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. In this context, most often extension degrees are composite (e.g. $n = 12$). To give but one example, we find non

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 8 and 9 give the complexities for various values of λ. Moreover, the Multiple variant does not couple with the Special variants SNFS and STNFS.

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 7**			
	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 8**			
STNFS	Medium p	**Table 9**			

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 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. Interestingly, our complexity analysis shows that the combination of the Multiple NFS variant with our Factory approach does not make it better (MNFS Factory brings no improvement over NFS Factory). 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 precisely based on the Chebotarev density theorem.

For each variant combined with Factory we provide, based on usual NFS heuristics, an improved asymptotic complexity for the computation of discrete logarithms, 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 one-off precomputation 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 we obtain in this work.

Besides, we employ an analytic approach in order to assess the crossover point above which our Factory approach for NFS and TNFS is likely to be profitable. When applied to TNFS with 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 have in mind 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 and explains how we can predict how many fields can be addressed with a given Factory setup. 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 to target several 795-bit prime fields or several 1024-bit finite fields of extension degree 6.

2 Background

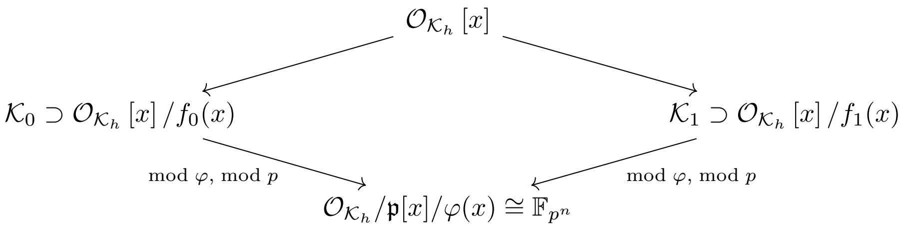
Notations. From now on, p always denotes a prime number. When the extension degree n of the finite field $\mathbb{F}_{p^n}$ is composite, η and κ denote non trivial factors of n such that $n = \eta\kappa$. Asymptotic estimates use the classical O and o notations, as well as the soft-O notation $f = \widetilde{O}(g)$ 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 x-*smooth* if we can write it as product of integers that are all smaller than x. Likewise, an ideal is x-smooth if it factors into a product of prime ideals whose (absolute) norm is less 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. For a number field K, we let $\mathcal{O}_K$ be its ring of integers. For simplicity, we assume throughout this article that $\mathcal{O}_{\mathcal{K}_h} = \mathbb{Z}[\iota]/h$. This implies in particular that h is monic. (For the usual NFS, we rather let $\eta = 1$, $\mathcal{K}_h = \mathbb{Q}$, and $\mathcal{O}_{\mathcal{K}_h} = \mathbb{Z}$; 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 $\mathcal{O}_{\mathcal{K}_h}$ 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. Let α_i be root of f_i in $\mathcal{K}_i$ for $i = 0, 1$. Because of the conditions on the polynomials h,



Diag. 4: Commutative diagram of Tower NFS.

f_0 and f_1, there exist two ring homomorphisms from $\mathcal{O}_{\mathcal{K}_h}[x]$ 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 in Figure 4. For simplicity, we assume that f_0 and f_1 are defined over $\mathbb{Z}$, although this is only possible when κ and η are coprime.

The *polynomial selection* refers to 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 virtual logarithms 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.

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 them) 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 $\phi(x,\iota) \in \mathcal{O}_{\mathcal{K}_h}[x]$ at the top of the diagram, the candidates that yield a relation. A relation is found if the polynomial $\phi(x,\iota)$ mapped to principal ideals in $\mathcal{O}_{\mathcal{K}_0}$ and $\mathcal{O}_{\mathcal{K}_1}$ are *smooth* (respectively B_0- and B_1-smooth). Most often the search space for relation collection consists of linear polynomials $\phi(x,\iota) = a(\iota) - b(\iota)x \in \mathcal{O}_{\mathcal{K}_h}[x]$, and for usual NFS this simplifies to searching for polynomials $a - bx$ with integers coefficients a, b, since $\mathcal{O}_{\mathcal{K}_h} = \mathbb{Z}$ in that case. The ideals that occur in the factorizations in $\mathcal{O}_{\mathcal{K}_0}$ and $\mathcal{O}_{\mathcal{K}_1}$ 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 $i = 0, 1$:

$$\mathcal{F}_i(B_i) = \{\text{prime ideals of } \mathcal{O}_{\mathcal{K}_i} \text{ of norm} \leq B_i \text{ and inertia degree 1 over } \mathcal{K}_h\}.$$

To test 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:

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

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 test the B_i-smoothness over integer values. The relation collection 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 one or several target elements. 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 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, AAP23].

2.2 Other variants of NFS

Special NFS. When the characteristic is sparse (the meaning of which will be made precise later on), 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. It makes sense to ask whether an MNFS Factory variant is worthwhile. It turns out that the answer is no. Such a combination would not lower the complexity of the per-field step, as we briefly discuss in Section 4.

2.3 Smoothness probability

As is classical with analysis of NFS-based algorithms, we assume throughout the paper that the probability of a norm being smooth is the same as that of a random integer of the same size. To assess this latter probability, we use the following restatement of results from [CEP83]:

Proposition 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 given by 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 $\mathcal{O}_{\mathcal{K}_h} = \mathbb{Z}[\iota]$, we can assume that all coefficients of $\phi(x, y)$ are integers, all below a bound $\|\phi\|_\infty$. We obtain

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

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 $\deg_y \phi < \deg h$ and (unless specified otherwise) $\deg_x \phi = 1$.

3 Discrete logarithm Factory

3.1 Common Setting

Whether it is deployed for integer factorization or for discrete logarithm in medium or large characteristic finite fields, the *Factory* algorithm revolves around the same idea. The primary objective is to share a portion of the *relation collection* step in NFS (or a variant). Our common setting is as follows.

Common Setting. *A family of instances (Q_i, n_i), indexed by i, is defined. Both Q_i and n_i increase along with i. We conduct an asymptotic study as i tends to infinity. How Q and n evolve relative to eachother defines several regimes, which are studied separately.*

For each instance (hereafter denoted without the subscript i for brevity), Q denotes an approximate finite field cardinality, and n an extension degree. A given instance focuses on the set of finite fields $\mathbb{F}_{p^n}$ of approximate cardinal Q—therefore we have $p^n \approx Q$. Our goal is to define a precomputation algorithm that allows efficient computation of discrete logarithms in a significant proportion—dependent on the instance parameters Q and n—of them.

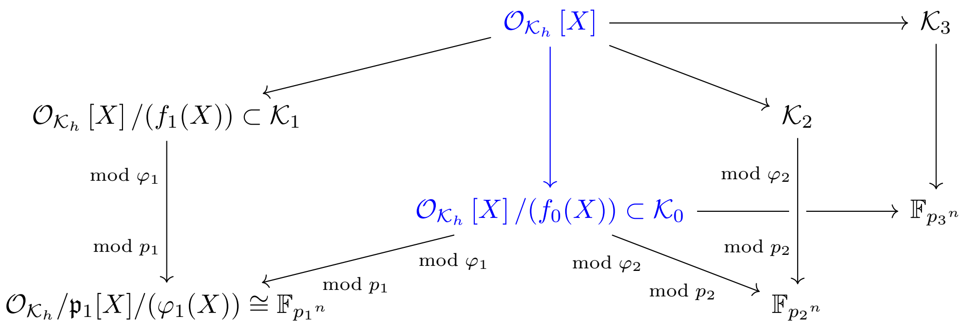
In the following, we explain how one instance of the Factory algorithm works: we consider finite fields $\mathbb{F}_{p_1^n}, \mathbb{F}_{p_2^n}, \ldots$ of extension degree n and approximate cardinal $Q \approx p_1^n \approx p_2^n \approx \cdots$.

To achieve this, our Factory approach consists of two steps. Figure 5 illustrates this.

The "one-off" step. Inputs are Q and n. We construct *half* of the diagram of Figure 4, namely $\mathcal{K}_h$ and $\mathcal{K}_0$. Then, a first search aims to find (and store for later use) 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 step —including B_0 and the number of elements ϕ to test— depend on Q and n.

The "per-field" step. Consider one of the p_i's of the common setting. Complete the diagram of Figure 4 (define a number field $\mathcal{K}_i$) so that the target finite field is $\mathbb{F}_{p_i^n}$. The *relation collection* step proceeds by determining which of the stored ϕ are B_*-smooth when mapped to $\mathcal{K}_i$, where B_* is another smoothness bound. Because this *per-field* step works in a similar way for primes of similar size, parameters such as B_* are identical for all the fields. The remaining steps of NFS (or the variant) are unchanged.

The complexities we formulate are functions of Q and n. Just like finite field discrete logarithm distinguishes between small, medium, and large characteristic, we will make distinctions based on how Q and n evolve asymptotically. Likewise, we will define several variants that are adapted to n factoring in a certain way, or the primes p_i being of a special form.



Diag. 5: Example of a commutative diagram for Factory for three target finite fields. The blue central branch is where the *one-off* precomputation takes place. This extends Diagram 4 to multiple right sides (here, $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_3$).

3.2 A baseline: Factory algorithm for prime fields

The factorization Factory algorithm was introduced by Coppersmith [Cop93] and its adaptation to the discrete logarithm problem in prime finite fields was proposed by Barbulescu [Bar13].

We follow the common setting of §3.1 but restrict ourselves to $n = 1$. The *one-off* step sets $\mathcal{K}_h = \mathbb{Q}$ (hence $\eta = 1$), and starts with the well known base-m method. Choose a degree d and an integer m close to $Q^{1/d}$. Define $\mathcal{K}_0$ by $f_0(X) = X - m$. For the *per-field* step, write the base-m expansion of p_i as $p_i = \sum_{k=0}^{d} a_k m^k$ and set $f_i(X) = \sum_{k=0}^{d} a_k X^k$. Then, f_0 and f_i share a common root modulo p_i, which is m. Define $\mathcal{K}_i$ as $\mathbb{Q}[X]/f_i$ (the polynomial f_i is generally irreducible). This completes Diagram 4.

3.3 Factory for non prime finite fields: polynomial selection

The novelty of this article is the generalization of the Factory approach to finite fields of arbitrary extension degree. 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 cannot be used.

We follow the notations of §2.1. In particular, $\eta = \deg h$ is non trivial only in the tower cases (TNFS, STNFS). In order to simplify the exposition, we assume that η and κ are coprime, which allows us to search for f_0 and f_i in $\mathbb{Z}[X]$ rather than in $\mathcal{O}_{\mathcal{K}_h}[X]$. Both f_0 and f_i must be coprime and irreducible, and share an irreducible factor φ_i of degree κ modulo p_i. Then $\mathbb{F}_{p_i^n}$ is represented as $(\mathcal{O}_{\mathcal{K}_h}/p_i\mathcal{O}_{\mathcal{K}_h})[X]/(\varphi_i)$. In the different polynomial selection methods that we now review, we assume that the polynomial h has been fixed beforehand, and we only detail how the polynomials f_0, and f_i are chosen (in conjunction with p_i).

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.

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 $\varphi_i(X) = X^\kappa + \sum_{j=0}^{\kappa-1} \varphi_{i,j} X^j$ with $-p_i/2 < \varphi_{i,j} \le p_i/2$ for $0 \le j \le \kappa - 1$. Build the lattice of dimension $(d+1)\times(d+1)$ whose basis matrix is:

$$M_{p_i} = \begin{pmatrix} p_i & & & & & \\ & \ddots & & & & \\ & & p_i & & & \\ \varphi_{i,0} & \varphi_{i,1} & \dots & 1 & & \\ & \ddots & \ddots & & \ddots & \\ & & \varphi_{i,0} & \varphi_{i,1} & \dots & 1 \end{pmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} \kappa \text{ rows} \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} d+1-\kappa \text{ rows} \end{matrix}$$

The shortest vector output by the LLL algorithm when applied to M_{p_i} gives the coefficients of a polynomial f_i that is a multiple of φ_i modulo p_i. We safely assume that f_i is irreducible over $\mathbb{Z}$; in the unlikely event that it is not, we replace it with the appropriate irreducible factor that reduces modulo p_i to a multiple of φ_i. Remark that as the dimension of M_{p_i} is $d+1$, and its determinant is p_i^{κ}, lattice reduction 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 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 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 $u/v \equiv \lambda_i \mod p$ 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 $O(\sqrt{p_i})$.

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 characteristic in both medium and large characteristic 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 $\mathbb{Q}$.

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 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 $\widetilde{O}(\log(\kappa)^{d_i})$.

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 $\mathbb{Q}$. 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 both 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 $\mathbb{Q}$.

Table 6 summarizes the degrees and sizes of the coefficients of the polynomials output by the methods that we just mentioned. To fix terminology, in the remainder of the paper

Table 6: Degrees and infinity norms of the polynomials given by the different polynomial selections used for our Factory variants. This table assumes that $p_0^n \approx p_i^n \approx Q$.

Polynomial selection \ Properties of f_0 and f_i	$\deg(f_0)$	$\deg(f_i)$	$\|f_0\|_\infty$	$\|f_i\|_\infty$
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, 1st approach	κ	$\lambda d,\ d \approx \lambda$	$\widetilde{O}\left(Q^{1/(\lambda n)}\right)$	$\widetilde{O}\left(\log(\kappa)^d\right)$
Joux–Pierrot, 2nd approach	$\lambda\kappa$	κ	$\widetilde{O}\left(\log(\kappa)^\lambda\right)$	$\widetilde{O}\left(p^{1/\lambda}\right)$

we will sometimes refer to *NFS Factory* when $\eta = 1$ and *TNFS Factory* if both η and κ are greater than one. As is the case with non-Factory variants, the Generalized-Joux–Lercier method is well suited to the large characteristic case, while the Conjugation method is intended for the medium characteristic case. The boundary case is not a clear cut. As regards "special" primes, whenever either of the Joux–Pierrot constructions can be used we use the terms *SNFS Factory* (when $\eta = 1$) or *STNFS Factory* (in the tower case).

3.4 Fantastic primes and how many are there?

Each of the polynomial selection methods in §3.3 lays out requirements on the primes p_i. How many of the primes p_i work with a given setup of the *one-off* step depends on properties of the number field tower that is used to define $\mathcal{K}_0$. This is actually controlled by the Chebotarev density theorem.

3.4.1 Chebotarev density Theorem in towers of number fields.

Consider the tower $\mathbb{Q} \subset \mathcal{K}_h \subset \mathcal{K}_0$. The field $\mathcal{K}_0$ need not be a normal field, so let us also define its normal closure L and let $G = \mathrm{Gal}(L/\mathbb{Q})$. By Galois correspondence, this tower is connected to the chain of subgroups $\{1\} < G^{\mathcal{K}_0} < G^{\mathcal{K}_h} < G$, where G^X denotes the subgroup of G that fixes the subfield $X \subset L$. The group G acts on the cosets $G/G^{\mathcal{K}_h}$, which are partitioned in a set of smaller cosets $G/G^{\mathcal{K}_0}$. The Frobenius symbol $\left[\frac{L/\mathbb{Q}}{p}\right]$ (defined up to conjugation) and the Chebotarev density Theorem [Mil20, Chapter 8] tell us two things. Here, we consider only primes that are coprime to $\mathrm{disc}(L/\mathbb{Q})$ and to all leading coefficients of the defining polynomials.

- The decomposition of a prime number $p \in \mathbb{Q}$ as a product of prime ideals in $\mathcal{K}_h$ and $\mathcal{K}_0$ (and, eventually, in L) can be read off directly from the orbits of the action of the cyclic subgroup generated by $\left[\frac{L/\mathbb{Q}}{p}\right]$ on the cosets $G/G^{\mathcal{K}_h}$, $G/G^{\mathcal{K}_0}$, and so on. For example its orbits on $G/G^{\mathcal{K}_h}$ have sizes $n_1, \dots, n_k$ if and only if p factors into prime ideals of degrees $n_1, \dots, n_k$ in $\mathcal{K}_h$. If we take a closer look at how $\left[\frac{L/\mathbb{Q}}{p}\right]$ acts on the smaller cosets $G/G^{\mathcal{K}_0}$, then these orbits split into orbits of sizes $(n_{i,j})_{1\le i\le k, 1\le j\le k_i}$ (with $\sum_j n_{i,j} = n_i$) if and only if the i-th prime ideal above p in $\mathcal{K}_h$ splits into factors of degrees $n_{i,1}, \dots, n_{i,k_i}$ in $\mathcal{K}_0$. This extends to towers of arbitrary height.
- For S a subset of the set of prime numbers, define the density of S as

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

> Chebotarev's theorem says that the density of primes whose decomposition patterns along the tower matches the orbit sizes of the action of a conjugacy class $C \subset G$ is exactly the ratio $|C|/|G|$.

Computationally accessible data. In theory, the above results are strong enough to predict the density of primes that work with the setup of any given *one-off* step. Alas, the computation of the Galois group of (the normal closure of) $\mathcal{K}_0$ may be out of reach. In some specific cases, it is possible to compute the densities based on data related to smaller fields. We will discuss a few such cases below. Supplementary material of this work includes a short Magma program that computes these densities, given a tower of number fields.

Intervals and explicit bounds. It will be of some use in this paper to discuss the density of primes that we can use in intervals rather than over all primes. This is a well studied problem, which happens to be easy in the instances we will be looking at (and very challenging otherwise). Namely, we will be interested in intervals of the form $[x, x^A]$ for $A > 1$, and in such cases the error bounds given by [LO77] suffice to prove that we have the expected density. We will not discuss this point further.

3.4.2 Some specific cases.

Here we allow some simplifying assumptions. A baseline is given in the case where $\mathcal{K}_0$ and $\mathcal{K}_h$ are defined over $\mathbb{Q}$ (we already made this assumption in §2.1), and that their normal closures have no isomorphic subfields. Then the decompositions of h and f_0 modulo prime numbers are independent. In this case, the probability that h is irreducible modulo p_i, and f_0 has an irreducible factor of degree κ modulo p_i is given by

$$\frac{\#\operatorname{Gal}(h)_\eta \cdot \#\operatorname{Gal}(f_0)_\kappa}{\#\operatorname{Gal}(h) \cdot \#\operatorname{Gal}(f_0)}.$$

In this expression, $\operatorname{Gal}(f)_k$ is the set of elements of $\operatorname{Gal}(f)$ which have a cycle of length k in their action on the roots of f. The formula above applies to both the Generalized-Joux–Lercier Factory approach, and the Joux–Pierrot Factory, first approach. Note, of course, that in the non-tower cases, we have $\eta = 1$ and thus $\#\operatorname{Gal}(h) = \#\operatorname{Gal}(h)_\eta = 1$.

Conjugation Factory. In the Conjugation setup given in §3.3, the condition is more specific. Let α be a root of f_0 in $\mathcal{K}_0$. Then $\theta = -g_0(\alpha)/g_1(\alpha)$ is a root of μ, and $M = \mathcal{K}_h(\theta)$ is a subfield of $\mathcal{K}_0$, of degree 2 above $\mathcal{K}_h$. The number field tower that is of interest to us is $\mathbb{Q} \subset \mathcal{K}_h \subset M \subset \mathcal{K}_0$. The primes p_i that work in the Conjugation setup are those for which there exists a prime ideal $\mathfrak{p} \subset \mathcal{O}_{\mathcal{K}_0}$ such that $[\mathfrak{p} \cap \mathcal{O}_{\mathcal{K}_h} : p\mathbb{Z}] = \eta$, $[\mathfrak{p} \cap \mathcal{O}_M : \mathfrak{p} \cap \mathcal{O}_{\mathcal{K}_h}] = 1$, and $[\mathfrak{p} \cap \mathcal{O}_{\mathcal{K}_0} : \mathfrak{p} \cap \mathcal{O}_M] = \kappa$. If the Galois group of $\mathcal{K}_0$ and its subfields can be computed, we can determine how many Frobenius symbols reveal that at least one such prime ideal exists above p. By Chebotarev's theorem, this also gives the density of such primes.

Joux–Pierrot Factory, second approach. This case seems to be outside the scope of investigation by the methods that we just mentioned. As described in §3.3, an integer u_i varies, and the cases of interest are when $p_i = P(u_i)$ is prime and the polynomial $X^\kappa + R(X) - u_i$ is irreducible mod p_i. Contrary to the cases above, this polynomial varies together with p_i. Short of a better solution, we hypothesize the following.

Assumption 1. *In the context of the Joux–Pierrot construction (second approach), 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.*

In addition to the above, the Joux–Pierrot setup, when instantiated in the tower case, also requires that h is irreducible modulo p_i. We will assume that this latter condition is independent of the irreducibility of $X^\kappa + R(X) - u$.

It is straightforward to test Assumption 1 over arbitrary examples. We did so for various choices of η, κ, and $\lambda = \deg P$, and found good accordance of experimental data with Assumption 1.

3.4.3 Limitations of the Galois point of view.

There are two main caveats to estimates given by the Galois theory approach. First, explicitly computing Galois groups is not always easy, and while these computations are extremely easy in the examples we considered, we cannot rule out that it becomes out or reach in certain cases. Second, even if we can mathematically write what the proportion is, it can actually be that this formula predicts a density of zero, which is not very useful. We can, for example, fabricate examples in the Conjugation method with $\kappa = \ell^2$ for a prime ℓ and for which $\mathrm{Gal}(f_0) = \mathbb{Z}/(2\ell)\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$. In such a case, no prime ideal of degree ℓ^2 exists, and obviously such setups would be of no use for computing discrete logarithms!

This being said, the high-level view tells us that the factorization patterns modulo primes definitely follow predictable patterns. Empirical observations are a quick and easy way to get an idea of the correct ratios (in fact, these same empirical observations can be leveraged to get insights about what the Galois groups are). For a Factory approach to apply to as many primes as possible, it certainly makes sense to assess what happens modulo a moderate collection of primes.

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

As an illustration, we show two different constructions, together with an evaluation of the proportion of primes (i.e. characteristics) that can be reached. The ratios of primes that we mention can be computed with the Magma script that is provided as supplementary material with this work.

NFS Factory with Conjugation. The authors of [GMT16] report a discrete logarithm computation on $\mathbb{F}_{p^3}$ with NFS (that is, no tower is at play: we have $\eta = 1$) for the 593-bit prime $p = 908761003790427908077548955758380356675829026531247$. The Conjugation method was used, and it produced:

$$\begin{aligned} f_0 &= \mathrm{Res}_Y(X^3 - 3X - 1 - Y(X^2 + X), 28Y^2 + 16Y - 109) \\ &= 28X^6 + 16X^5 - 261X^4 - 322X^3 + 79X^2 + 152X + 28 \\ f_1 &= 24757815186639197370442122X^3 + 40806897040253680471775183X^2 \\ &\quad -33466548519663911639551183X - 24757815186639197370442122 \end{aligned}$$

The absolute Galois group $\mathrm{Gal}(f_0)$ comprises eighteen permutations. Eight of them act on the cosets of the Galois tower in a way that is consistent with p splitting in M and being inert in $\mathcal{K}_0$. This predicts that a fraction of 4/9 of the primes work, which we observe experimentally. For instance, let

$$p_2 = 925345433540865564015707127491171005390356157011113,$$

modulo which f_0 factors into an irreducible polynomial of degree 3 and three linear polynomials. If we apply the method given in §3.3, 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 less than $1.45 \times \sqrt{p_2}$.

$$f_2 = 176789951198543558126224 58X^3 + 438660709226929695016658 11X^2$$
$$-917091443687009793620156 3X - 176789951198543558126224 58$$

TNFS Factory with Conjugation. In [DGP21], a 521-bit discrete logarithm computation was carried out on $\mathbb{F}_{p_1^6}$ with $p_1 = 1350664108659952233496039 27$ using TNFS where polynomials were chosen with the Conjugation method as:

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

In this case, the tower $\mathcal{K}_0 \supset M \supset \mathcal{K}_h \supset \mathbb{Q}$ corresponds to the chain of Galois groups $\mathbb{Z}/2\mathbb{Z} < (\mathbb{Z}/2\mathbb{Z})^2 < (\mathbb{Z}/2\mathbb{Z})^3 < (\mathbb{Z}/2\mathbb{Z})^3 \ltimes (\mathbb{Z}/3\mathbb{Z})$. We can also write this chain as $\langle\alpha\rangle < \langle\alpha,\beta\rangle < \langle\alpha,\beta,\gamma\rangle < \langle\alpha,\beta,\gamma,\sigma\rangle$, with $\alpha^2 = \beta^2 = \gamma^2 = \sigma^3 = 1$ and the only non-abelian relation being $\alpha\sigma = \sigma^2\alpha$. The $G^{\mathcal{K}_0}$-cosets can be written as $G^{\mathcal{K}_0}\beta^{\{0,1\}}\gamma^{\{0,1\}}\sigma^{\{0,1,2\}}$, the G^M-cosets can be written as $G^M\gamma^{\{0,1\}}\sigma^{\{0,1,2\}}$, the $G^{\mathcal{K}_h}$-cosets can be written as $G^{\mathcal{K}_h}\sigma^{\{0,1,2\}}$. The multiplication by $\tau = \beta\sigma$ on the right has a single orbit of size 3 on the $G^{\mathcal{K}_h}$-cosets, which splits into two orbits, still of size 3, on the G^M-cosets. These become two orbits of size 6 on the $G^{\mathcal{K}_0}$-cosets. The only elements of G with this pattern are τ and τ^{-1}, which makes for $\frac{1}{12}$ of the possible Frobenius elements. This correctly predicts the fraction of primes p_i for which this number field tower works in a Factory setting.

For example, we can consider $p_2 = p_1 + 456$, modulo which the polynomial h is irreducible, $Y^2 - 2$ has a root, and f_0 factors into two irreducible polynomials of degree 2. The Conjugation method yields $f_2 := 11622094549025X^2 - 115506194478X + 11622094549025$, which completes Diagram 5. Its largest coefficient in absolute value is less than $1.01 \times \sqrt{p_2}$.

4 Asymptotic analysis

This section provides the complexities of the *one-off* step and the *per-field* step in each of the NFS variants that we combine with Factory. In Table 3 we compare our results to the analyses found in the literature for the non-Factory NFS variants [BGGM15, KB16, Pie15, SS16a, JP14]. Recall that our common setting is as in §3.1, and that as far as analysis goes, we will assume the classical NFS heuristics of §2.3.

Notations. For Q a finite field size, we let c_A, c_0, c_* be 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 $i > 0$. We let $\mathcal{N}_0$ (resp. $\mathcal{N}_*$) denote bounds on the norms of the sieve elements norms once mapped to K_0 (resp. to $\mathcal{K}_i$ for $i > 0$). In all variants, parameters are such that $\mathcal{N}_0 = L_Q(2/3, c_{\mathcal{N}_0})$ (likewise for $\mathcal{N}_*$) where $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_*}$ depend on c_A and other parameters. By Proposition 1, an element in $\mathcal{K}_0$ of norm $\mathcal{N}_0$ is B_0-smooth with probability $\mathbf{P}_0 = L_Q(1/3, c_{\mathcal{N}_0}/(3c_0))^{-1}$. Likewise, for other fields $\mathcal{K}_i$ we define $\mathbf{P}_*$ and we have $\mathbf{P}_* = L_Q(1/3, c_{\mathcal{N}_*}/(3c_*))^{-1}$.

Methodology. The *one-off* step is performed by a sieve algorithm that detects elements that are B_0-smooth once mapped to $\mathcal{K}_0$. The asymptotic complexity of this step is $A^{1+o(1)}$. The number of elements stored for later use 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 *per-field* step starts by detecting which of the stored elements are B_*-smooth once mapped to $\mathcal{K}_i$. We can perform this detection with either 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_*-smooth is $(A\mathbf{P}_0)^{1+o(1)}$, which is similar to the complexity in memory of the algorithm.

The *per-field* step proceeds with a sparse linear algebra phase that costs $(B_0+B_*)^{2+o(1)}$, and an individual logarithm computation of negligible complexity compared to the two previous steps. The complexity of the *per-field* step is $\left(A\mathbf{P}_0 + (B_0 + B_*)^2\right)^{1+o(1)}$.

We want to minimize the complexity of the *per-field* step (Equation (2) below). Some necessary conditions apply: we need enough equations for the linear algebra step (Equation (3) below), and we want to balance the costs of smoothness detection and linear algebra, as is done in many asymptotic analyses of NFS (Equation (4) below). This rewrites as:

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

under conditions

$$c_A - c_{\mathcal{N}_0}/(3c_0) - c_{\mathcal{N}_*}/(3c_*) \geq \max(c_0, c_*) \tag{3}$$

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

where $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_*}$ are polynomials of degree at most one in c_A, and do not depend on 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 $\tilde{c_A} < c_A$ so that the right-hand side of Equation (4) does not change. This can be done because $c_A - c_{\mathcal{N}_0}/(3c_0)$ increases as a function of c_A. Then Condition (3) still holds and the minimum value in (2) is unchanged.

Therefore we may take $B_0 = B_* = L_Q(1/3, c)$ and slightly rearrange the system into the following equivalent form.

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

under conditions

$$3c^2 \geq c_{\mathcal{N}_*} \tag{6}$$

$$\text{and} \qquad 6c^2 - 3c_A c + c_{\mathcal{N}_0} = 0 \tag{7}$$

MNFS Factory. We briefly discuss the possibility of an MNFS Factory variant. If $V := L_Q(1/3, c_v)$ polynomials are selected at the per-field step to construct a MNFS diagram, then the system of equations to solve becomes

$$\begin{cases} \text{Minimize } c \\ 6c^2 - 3(c_A + c_V)c + c_{\mathcal{N}_0} & = 0 \\ 3c^2 - 3c_V c - c_{N_*} & = 0 \\ c - c_V & > 0 \end{cases}, \tag{8}$$

where $B = L_Q(1/3, c)$ is the smoothness bound on the shared side and $B_* := L_Q(1/3, c - c_v)$ is the smoothness bound on the other sides. By combining the two first equations we get that for any (c_a, c_V) we have $c = (c_{\mathcal{N}_0} + 2c_{N_*})/(3c_A - 3c_V)$, where $c_{\mathcal{N}_0}$ and c_{N_*} are both constants with respect to c_V, regardless of the polynomial selection method employed. Therefore, the optimal values for c is always achieved at $c_V = 0$. In other words, the optimal solution is reached without the Multiple variant: and MNFS Factory variant cannot lower the complexity of the per-field step.

4.1 NFS Factory and TNFS Factory

Theorem 1 presents the complexities of NFS Factory and TNFS Factory in the large characteristic, boundary, and medium characteristic cases.

Theorem 1 (Complexities of NFS Factory and TNFS Factory)**.** *Let $\alpha \in (1/3, 1)$ and $c_p > 0$ be two constants. In the common setting of §3.1, we study the regime where inputs Q and n are such that $Q^{1/n} = L_Q(\alpha, c_p)$. Let f_0 (and h for the tower variants) be polynomials constructed for the* one-off *step by one of the methods in 3.3. For a proportion σ of the prime numbers p_i such that $Q \le p_i^n \le Q \cdot Q^{o(1)}$, the Factory algorithm succeeds. The proportion σ can be computed along the lines of §3.4 (either with Galois theory or empirically). The* one-off *step costs $L_Q(1/3, c_A)$, the storage cost is $L_Q(1/3, 2c)$, and the* per-field *cost is $L_Q(1/3, 2c)$. The values of c_A and c depend on the characteristic size and the algorithm employed:*

1. *Large characteristic: $2/3 < \alpha < 1$.*
   - *(a)* ***NFS Factory.*** *f_0 is constructed by the GJL method. The optimal values are $2c = 2((2+\sqrt{6})/6)^{2/3} \approx 1.64$, and $c_A = c\sqrt{6} \approx 2.01$.*
2. *Boundary: $\alpha = 2/3$ (hence $Q^{1/n} = L_Q(2/3, c_p)$).*
   - *(a)* ***NFS Factory with GJL.*** *Under the condition $c_p \ge \gamma$, the situation is identical to the case above, the threshold value γ being $\frac{\sqrt{6c}}{2} \approx 1.11$.*
   - *(b)* ***NFS Factory with Conjugation.*** *Let t be a fixed integer that denotes the sieve dimension (i.e., $\deg_x \phi = t-1$). f_0 is constructed by the Conjugation method. The optimal value for c is the smallest real solution of Equation (9), resulting in $c_A = 6c_p tc^2/(3c_p tc - 2)$.*

$$18c_p tX^3 - 24X^2 - 3c_p^2 t(t-1)X + 2c_p(t-1) = 0 \tag{9}$$

3. *Medium characteristic: $1/3 < \alpha < 2/3$.*
   - *(a)* ***NFS Factory.*** *f_0 is constructed by the Conjugation method. The optimal values are $2c = 2((1+\sqrt{2})/3)^{2/3} \approx 1.73$ and $c_A = 2c\sqrt{2} \approx 2.45$.*
   - *(b)* ***TNFS Factory.*** *h and f_0 are constructed by the Conjugation method. The degree of h is denoted η and is a non trivial factor of n. Denote $\kappa = n/\eta$. In the optimal case where $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q)))^{1/3+o(1)}$ with $c_\kappa = \sqrt{2}((2+2\sqrt{2})/3)^{1/3} \approx 1.66$, the optimal values are $2c = ((2+2\sqrt{2})/3)^{2/3} \approx 1.37$, and $c_A = 2c\sqrt{2} \approx 1.94$.*

Table 3 recapitulates the complexities announced in Theorem 1 together with the previous state-of-the-art complexities of NFS and its variants.

Remark 1. It is worth noting that if the large characteristic regime of Theorem 1 is pushed towards $\alpha = 1$, the asymptotic complexities of the *one-off* step and the *per-field* step for large characteristic finite fields are the same as in NFS Factory for prime fields. However, the parameter values that allow to reach the minimal complexity for the *per-field* step are not. Specifically, our parameter γ in the proof of Theorem 1 and the corresponding parameter $1/\delta$ in [Bar13, page 98] are different.

Proof. We prove the complexity announced for NFS Factory for large characteristic finite fields in Theorem 1. The rest of the proof is in in Appendix A, since it follows the same patterns.

We study the case where $Q^{1/n} = L_Q(\alpha)$ with $2/3 < \alpha < 1$. The case of finite fields with $\alpha = 1$, i.e., prime finite fields, is detailed in [Bar13, §7.2]. The *Generalized-Joux–Lercier* method is detailed in §3.3 and the degrees and coefficient sizes of the polynomials it outputs are given by Table 6. The sieve for the *one-off* step is performed in dimension 2, because $\deg \phi = 1$ turns out to be the best choice for large characteristic finite fields. It follows that $\|\phi\|_\infty \leq \sqrt{A}$. Furthermore, we set a constant γ such that $d = 1/\gamma\left(\log(Q)/\log(\log(Q))\right)^{1/3}$. Following the bound given in §2.3, the upper bounds on the norms 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 $\mathcal{N}_* = \widetilde{O}\left(A^{d/2}Q^{1/(d+1)}\right) = L_Q\left(2/3, c_A/(2\gamma)+\gamma\right)$, from which we obtain the expressions of $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_*}$.

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 $\rho = ((2+\sqrt{6})/6)^{2/3}$. If $0 < c < \rho$, 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, $c \geq \rho$. In fact, $c = \rho$ is a valid solution. The solution to the system is given by

$$c = \left(\frac{2+\sqrt{6}}{6}\right)^{\frac{2}{3}} \approx 0.82, \quad \gamma = \frac{\sqrt{6c}}{2} \approx 1.11, \quad c_A = c\sqrt{6} \approx 2.01.$$

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 *per-field* step is $L_Q(1/3, 2c) \approx L_Q(1/3, 1.64)$.

Still in the context of the common setting given in §3.1, we want to know how much leeway we have in the choice of p_i. The size of p_i only affects $\mathcal{N}_*$. As long as $p_i^n \leq Q^{1+o(1)}$, it is easy to see that the asymptotic results above are unchanged. □

Remark 2 (Comparisons at the boundary). Multiple algorithms compete in the boundary case. In addition to the complexities given by Theorem 1, other state of the art results are usual (non-Factory) NFS, as well as the MNFS variant. Both can use either the GJL or Conjugation constructions [BGGM15, Pie15]. Their costs are:

- NFS with GJL: $L_Q\left(1/3, (64/9)^{1/3}\right) \approx L_Q(1/3, 1.92)$ if $c_p \geq (8/3)^{1/3} \approx 1.39$
- MNFS with GJL: $L_Q(1/3, (2(46+13\sqrt{13})/27)^{1/3}) \approx L_Q(1/3, 1.90)$ if $c_p \geq ((7+2\sqrt{13})/6)^{1/3} \approx 1.33$.
- NFS with Conjugation: $L_Q\left(1/3, 2/(c_pt) + 2\sqrt{1/(c_pt)^2 + c_p(t-1)/6}\right)$.
- MNFS with Conjugation: $L_Q\left(1/3, 2/(c_pt) + 2\sqrt{5/(9(c_pt)^2) + c_p(t-1)/6}\right)$.

Figure 7 depicts the interplay of these different results, together with the complexities of Theorem 1.

4.2 SNFS Factory and STNFS Factory

SNFS is designed for finite fields where the characteristic p is a sparse prime, where the adjective "sparse" is taken here with the ad hoc meaning that we can write $p = P(u)$, where P is a polynomial of small degree and coefficients (subject to specific size constraints), and u is an integer. Theorem 2 presents the complexities of SNFS in large and medium characteristic finite fields and of STNFS Factory.

Theorem 2 (Complexities of SNFS Factory and STNFS Factory)**.** *Let $\alpha \in (1/3, 1)$ be a constant. In the common setting of §3.1, we study the regime where inputs Q and n are such that $Q^{1/n} = L_Q(\alpha)$. Let f_0 (and h for the tower variant) be polynomials constructed for the* one-off *step by one of the methods of §3.3. For a proportion σ of a set $\mathcal{P}$ of* sparse

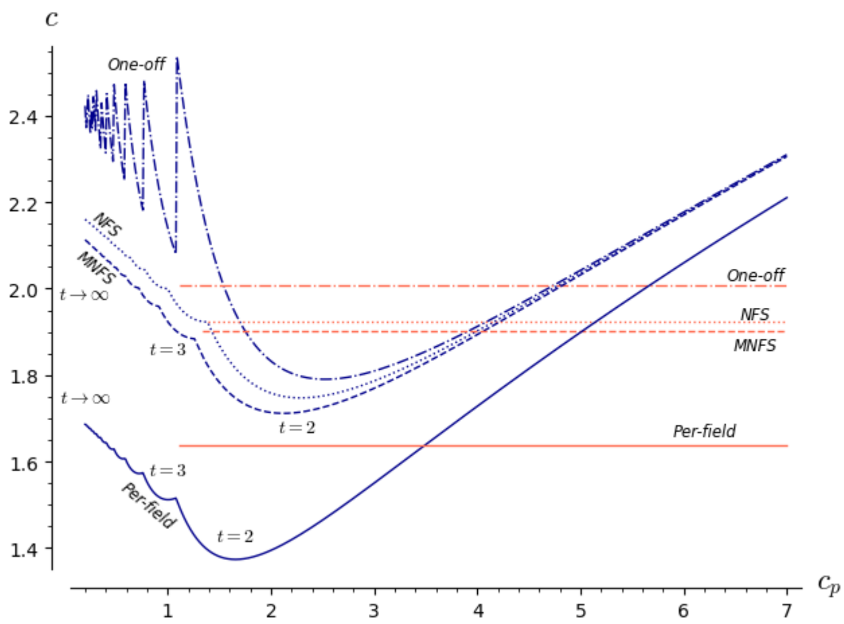


Fig. 7: Asymptotic complexities of NFS, MNFS, and NFS Factory when $p = L_{p^n}(2/3, c_p)$. The complexities are $L_{p^n}(1/3, c)$ and c is a function of c_p in each case. Tomato lines (resp. darkblue curves) are for algorithms that use GJL (resp. Conjugation) method.

prime numbers p_i, the Factory algorithm succeeds. The one-off *step costs $L_Q(1/3, c_A)$, the storage cost is $L_Q(1/3, 2c)$, and the* per-field *cost is $L_Q(1/3, 2c)$. The values of c_A and c depend on the characteristic size and the algorithm employed:*

1. *Large characteristic: $2/3 < \alpha < 1$:*

   (a) ***SNFS Factory.*** *Let $\lambda = 1/(c_\lambda n) \cdot (\log(Q)/\log\log(Q))^{1/3}$ with $c_\lambda = (8/9)^{1/3} \approx 0.96$, and u an integer close to $Q^{1/(n\lambda)}$. The polynomial f_0 is constructed with the Joux–Pierrot first approach method. The prime p_i is chosen from the set $\mathcal{P} = \{P(u) \mid P \in \mathbb{Z}[x],\ P(u) \text{ is prime},\ \deg(P) = \lambda + o(1), \text{ and } \|P\|_\infty = O(1)\}$. A proportion $\sigma = \frac{\#\operatorname{Gal}(f_0)_n}{\#\operatorname{Gal}(f_0)}$ of these primes work. The optimal values are $2c = (8/3)^{1/3} \approx 1.39$ and $c_A = 2(8/9)^{2/3} \approx 1.85$.*

2. *Medium characteristic: $1/3 < \alpha < 2/3$. Let $\lambda > 1$ an integer and P a polynomial of degree λ and with coefficients in $O(1)$. In both cases below, the prime p_i is chosen from the set $\mathcal{P} = \{P(u) \mid P(u) \text{ is prime},\ Q \leq P(u)^n \leq Q \cdot Q^{o(1)}\}$.*

   (a) ***SNFS Factory.*** *The polynomial f_0 of degree λn is constructed with the Joux–Pierrot second approach method. Based on Assumption 1, a proportion $\sigma = \frac{1}{n}$ of the primes in $\mathcal{P}$ work. The optimal values are $c \geq \tilde{c} = ((\lambda + 4 + 2\sqrt{2\lambda + 4})/(9\lambda))^{1/3}$ and $c_A = 2c(1 + 2\lambda/(X - 2\lambda))$, with $X = (9c^3 + 1)\lambda - 2\lambda + (-72c^3\lambda + (81c^6 - 18c^3 + 1)\lambda^2)^{1/2}$. When $\lambda \geq 4$, we must have $c > \tilde{c}$. See Appendix A for more details.*

   (b) ***STNFS Factory.*** *The polynomials h and f_0 are constructed with the the Joux–Pierrot second approach method. The degree of h is denoted η and is a non trivial factor of n. Denote $\kappa = n/\eta$. Based on Assumption 1, a proportion $\sigma = \frac{1}{\kappa} \cdot \frac{\#\operatorname{Gal}(h)_\eta}{\#\operatorname{Gal}(h)}$ of the primes in $\mathcal{P}$ work. The optimal values are obtained* *when* $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q))^{1/3+o(1)}$ *(c_κ is given in Appendix A), and are as follows.* $c \geq \tilde{c} = ((\lambda + 4 + 2\sqrt{2\lambda+4})/(18\lambda))^{1/3}, c_A = 2c(1 + 2\lambda/(X - 2\lambda))$, *with* $X = (18c^3 + 1)l + (-144c^3 l + (324c^6 - 36c^3 + 1)l^2)^{1/2}$. *When* $\lambda \geq 3$, *we must have* $c > \tilde{c}$. *See Appendix A for more details.*

Proof. We give the proof of Theorem 2 in Appendix A. □

Table 3 recapitulates the complexities announced in Theorem 2 together with the previous state-of-the-art complexities of SNFS and its variants. Moreover, Tables 8 and 9 present the complexities of SNFS and STNFS in the medium characteristic case for various values of λ.

Remark 3 (Comparisons at the boundary). Our study indicates that coupling Factory with SNFS in the boundary case $Q^{1/n} = L_Q(2/3, c_p)$ does not always yield better 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 case.

Table 8: Asymptotic complexities (in $L_Q(1/3,\cdot)$) of SNFS (without Factory) and the two steps of SNFS Factory (case 2a of Theorem 2) in medium characteristic finite fields. The parameter λ is an integer that is an input to the Joux–Pierrot polynomial selection, second approach §3.3. When $\lambda \geq 4$, we adjust the parameters to keep the individual logarithm step negligible. $\tilde{c}$ is given by Equation (11) in §A.4.1.

λ	SNFS (without Factory)	SNFS Factory	
		one-off	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**

4.3 Conclusion of the asymptotic analysis

In Appendix B, we prove that the *individual logarithm* step is negligible compared to the *per field step* in all the variants discussed in this section. Therefore, the complexities

Table 9: Asymptotic complexities (in $L_Q(1/3,\cdot)$) of STNFS (without Factory) and the two steps of STNFS Factory (case 2b of Theorem 2) in medium characteristic finite fields of composite extension degree and appropriately sized factors. The parameter λ is an integer that is an input to the Joux–Pierrot polynomial selection, second approach §3.3 When $\lambda \geq 3$, we adjust the parameters to keep the individual logarithm step negligible. $\tilde{c}$ is given in Equation (A.4.2) in §A.4.2.

λ	STNFS without Factory	STNFS Factory	
		one-off	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**

presented here represent the overall asymptotic complexities for Factory in each case. Table 3 provides a summary of the complexities for NFS, all relevant variants included. We see that the Factory approach reduces the complexity of computing discrete logarithms for a wide range of finite fields, at the expense of a *one-off* computation. In our analysis, we choose to minimize the complexity of the *per-field* step at the expense of a larger *one-off* step, but other trade-offs are possible.

Covering an arbitrary large density of prime numbers with several one-off steps. In each NFS Factory variant, a given *one-off* step allows to target a constant proportion of finite fields of a given extension degree and size (see §3.4). We note however that this proportion typically depends on the extension degree n, which questions whether we live up to the promise of the common setting of §3.1, which is to succeed with significant probability as we consider instance sizes going to infinity. We can do better. If we combine several *one-off* steps, the number of "missed" primes drops exponentially, and nearly all prime numbers can be covered without affecting the asymptotic complexity. For example, consider finite fields of extension degree n and approximate cardinal Q. Assume that the ratio of successful primes is consistently at least $\frac{1}{2n}$. We believe that such a lower bound holds generically, except for pathological polynomial choices that should be easy to avoid. Prepare $2\log(Q)$ different *one-off* steps. Doing so forces us to mildly increase the upper bounds on the polynomial sizes, but calculation shows that the complexities come out unaffected by this change. The proportion of missed primes is then upper bounded by $(1 - 1/(2n))^{2\log(Q)} \leq Q^{-1/n} \approx 1/p$. Therefore we expect that only $O(1)$ primes around $p \approx Q^{1/n}$ are missed.

Possible optimizations if the target fields are known in advance. We assume here that we slightly depart from the common setting of §3.1 in that the target finite fields are known *before* the *one-off* computation begins. Following the reasoning above, it is possible to adaptively choose polynomials so that only a few *one-off* steps are needed to cover all primes. Additionally, the *one-off* and *per-field* steps can conceivably be merged in a single computation, which removes the need to store the output of the first sieve. The techniques of [BL14] would apply in this situation.

Logjam-Factory attack: multiple targets in each finite field. It is possible to combine a Logjam attack as in [ABD+15] with Factory in order to target not one, but several targets in several finite fields. After performing the *one-off* and *per-field* steps on a finite field, we learn the logarithms of the factor base elements related to some target field. Subsequently, an individual logarithm step recovers the logarithm of any target in this field with a negligible cost compared to the *per-field* step. Specifically, we can recover the logarithms of $L_Q(1/3, c_1 - c_2)$ targets without increasing the *per-field* step's asymptotic complexity, where $L_Q(1/3, c_1)$ and $L_Q(1/3, c_2)$ are the respective complexities of the *per-field* step and the *individual logarithm* steps.

5 Estimation of practical cost

Our asymptotic complexity results are promising, but hardly solve the question of the concrete parameter range where the Factory approach may be worthwhile. The assessment of the practicality of this approach, or of any NFS-related cryptanalysis proposal, can typically be done in a variety of ways, from the most to the least accurate.

1. Actual computations can be carried out. This is the case for large computational records such as the discrete logarithm record on a 795 bit prime finite [BGG+20].

The computational cost for this approach is high. For example, computations for a 1000-bit finite field are unattainable at this point (with academic resources).

2. Try to simulate each step and obtain a projection of the running time, as for example in [GS19].

3. Ignore all concerns related to actual data, and rely on the $L(1/3, c)$ formula, taking $o(1) = 0$ in its expression. This approach is common, but it has obvious shortcomings. (see [LV01] §2.4.4, p 263, or [LGST21]).

Our results remain asymptotic in nature, but this section tries nevertheless to provide insight that goes beyond the $L(1/3)$ estimates.

5.1 NFS Factory

In the NFS-DL context, there is some material that can be used as a base to provide experimental data. For a prime bit size around 800 bits, the record in [BGG+20] is a significant undertaking, and experimental data is provided publicly in reproducible form[3]. We used this as a base to understand whether a Factory approach could make sense at this size.

Comparison with [BGG+20], for the same degree. Our rationale for choosing parameters is as follows. The "ideal" setting of a Factory approach has little reason to use the same polynomial degrees as a non-factory computation. Indeed, Theorem 1 shows that their asymptotics differ. However, as a first step for a comparison with this record computation it makes sense to rely on the same data: this way, the amount of effort that is put in the selection of an exceptionally good polynomial for a given p can be put on an equal footing across different primes.

In [BGG+20], relations involving 35-bit primes were obtained by exploring an area of 2^{31} (a, b) pairs for each special-q in the range $[150e9, 300e9)$ which contains about $2^{32.4}$ of them. The yield is about 0.75 relation per special-q (counting only unique relations). Smoothness estimates based on the Dickman rho function, given the size of the norms that are considered in the computation, are slightly more optimistic and suggest a yield around 2.5 relations for each special-q. Good explanations for this are that we need to take duplicate relations into account, first, and also that actually finding the smooth numbers in this range is quite a bit of work, and despite all effort that went into the implementation, some part of them are inevitably missed. As a reference, the rough cost estimate of the whole relation collection in this computation, again based on the Dickman rho estimates, is about $2^{63.4}$. The meaning of this number does not matter much beyond the comparison to our Factory variant that follows.

For the Factory approach to make sense, we need to reduce the smoothness bounds. We explored the option of reducing them by two bits (hence 33-bit primes only), which should readily yield a factor 16 improvement of the linear algebra cost. We chose an area of 2^{33} (a, b) pairs for each special-q in the range $[600e9, 1200e9)$. Smoothness estimates based on the Dickman rho function suggest that we should have a yield around 0.13 relation per special-q. We ran the cado-nfs lattice sieving program for these parameters, and we obtained a yield closer to 0.07 relation per special-q (again, unique relations only), which is only slightly off. Nevertheless, this is enough to form a complete relation set. We estimate that the cost of the precomputation in this case would be about $2^{67.4}$ (so about 16 times more than in the record computation), and in terms of core-years, the timings that we obtained give a projected time which increases by a factor that is closer to 25. Based on [BGG+20], this should be close to 50,000 core-years. The data that would need to be

[3] https://gitlab.inria.fr/cado-nfs/records.

stored after the precomputation would be about 2^{60} (a, b) pairs. Counting 10 bytes for each, and given the current price of hard disk storage, such a facility would cost about a hundred million USD. With this precomputed data, our estimates are that we would reach a per-field cost in the whereabouts of 2^{60} for relation collection. Recall that linear algebra cost was reduced even further, so the expected speedup is about 10-fold.

These concrete settings are a good sign that for a cost that is about 25 times the cost of the 795-bit record computation in [BGG+20], we can compute discrete logarithms in any 795-bit prime field in a fraction of one-tenth of the cost of [BGG+20]. An attacker can thus use the Factory approach profitably if more than a few dozen fields are targeted.

Comparison with [BGG+20], for a different degree. As discussed above, asymptotics suggest that the degree of a Factory computation should be larger than in the single-field case. Compared to our previous experiment, an alternative parameter setting that seems to work would be to work with degree 5 instead of degree 4. Here, we do not have the same level of confidence than above in the quality of the polynomials, sieving parameters, and actual timings that we obtained. Based on Dickman rho estimates, we obtained parameter sets that suggest a factor of two improvement compared to the previous paragraph, which is certainly well within the error margin.

Comparison for 1024-bit fields. A similar approach can be followed for 1024-bit fields. Code is not exactly ready to tackle this problem size, and we are left with an approach that is based on Dickman rho estimates. It does seem that in this case as well, improving the per-field cost by a factor of 16 is possible with adequate parameter choices. (The strategy remains similar: decrease smoothness bounds, use larger special-q's, and increase the sieve table size for each of them.)

5.2 TNFS Factory

The purpose of this section is to compare computational cost estimates of TNFS and TNFS Factory on finite fields with extension degree equal to 6 with sizes $Q = p^6$ ranging between 300 and 2000 bits. For each finite field size two setups are possible: $\eta = 2$ and $\kappa = 3$ or $\eta = 3$ and $\kappa = 2$.

Polynomial selection. Previous records such as [DGP21, Rob22] suggest that the Conjugation method (§3.3) performs best in practice. Our analysis supports this and indicates that it should be the best method in practice for TNFS Factory as well. Let $h \in \mathbb{Z}[X]$ a degree η irreducible polynomial with small coefficients, and f_0 and f_1 in $\mathbb{Z}[X]$ of respective degrees 2κ and κ, output by the Conjugation method. Following Table 6, we assume that $\|h\|_\infty = 1$, $\|f_0\|_\infty = 1$, and $\|f_1\|_\infty = \sqrt{p}$. (This is supported by reported experiments: in [DGP21], these values were respectively equal to 1, 1, and approximately $1.0043 \times \sqrt{p}$.) The number fields of Diagram 4 are defined as $\mathcal{K}_h := \mathbb{Q}(\iota)$, $\mathcal{K}_0 := \mathbb{Q}(\iota, \alpha_0)$ and $\mathcal{K}_1 := \mathbb{Q}(\iota, \alpha_1)$, where ι, α_0 and α_1 are the respective roots of h, f_0 and f_1,

One-off step 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 elements $\phi(x, \iota) = a(\iota) - b(\iota)x$ such that $\phi(\alpha_0, \iota)$ is B-smooth, 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 similar ϕ such that both $\phi(\alpha_0, \iota)$ and $\phi(\alpha_1, \iota)$ 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 prime ideal $\mathfrak{q}$ in their factorization in one of the number fields.

For TNFS Factory, given an ideal $\mathfrak{q} \subset \mathcal{O}_{\mathcal{K}_0}$, a sieve algorithm is applied to detect which of the elements $\phi(\alpha_0, \iota) \in \mathfrak{q}$ are B-smooth (not counting the ideal $\mathfrak{q}$ in the factorization).

Furthermore, the sieve algorithm only considers 2η-dimensional vectors $(a, b) := \mathcal{M}_{\mathfrak{q}}(i, j)^T$ where $\mathcal{M}_{\mathfrak{q}}$ is a reduced basis of the lattice that is of determinant q and $(i, j)^T$ is within an Euclidean ball of some radius R. If the Euclidean norm of (i, j) corresponding to (a, b) is written r, relaying on §2.3 we estimate the norm of $\phi(\alpha_i, \iota)$ by $N_i(r, q) := S^{\eta \deg(f_i)} \|f_i\|_\infty^\eta$, for $i = 0, 1$, where $S = r \times q^{1/(2\eta)}$ is the bound on the coefficients of (a, b). (Here we assume $\|h\|_\infty = 1$, and ignore the extra combinatorial factor). Moreover, let $V_{2\eta}(r)$ be the volume of the 2η-dimensional ball of radius r, and ρ be the Dickman-de Bruijn function. We estimate the number of B-smooth elements among all elements that are divisible by $\mathfrak{q}$ as:

$$\int_{r=0}^{R} \rho\left(\frac{\log(N_0(r, q)) - \log(q)}{\log(B)}\right) \mathrm{d}V_{2\eta}(r).$$

In turn, the total number of B-smooth elements (i.e., the output size of the *one-off* step) is the product of the above estimate by the number of special-q considered. Furthermore, we estimate the computational cost of the *one-off* step by the number of special-q considered times the cost of the sieve algorithm per special-q, which we approximate as $V_{2\eta}(R) \log\log(B)$.

For TNFS, a sieve is performed in both number fields to detect elements that are B-smooth in both number fields. Alternatively, it is also possible to combine a sieve algorithm on the special-q side with a batch smoothness detection algorithm on the other side. The number of expected relations for $\mathfrak{q}$ (assuming it is on the $\mathcal{K}_1$ side) is:

$$\int_{r=0}^{R} \rho\left(\frac{\log(N_0(r, q))}{\log(B)}\right) \rho\left(\frac{\log(N_1(r, q)) - \log(q)}{\log(B)}\right) \mathrm{d}V_{2\eta}(r).$$

Again, this must be multiplied by the number of special-q considered to get the total number of expected relations, and 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 batch smoothness detection on the other side, then this estimate drops to 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 *per-field* step of TNFS Factory starts by detecting which of elements stored after the *one-off* step are B-smooth in $\mathcal{K}_i$. This can be done with batch smoothness detection with a quasi-linear cost in the number of the stored elements. The total number of relations produced is estimated as:

$$\int_{r=0}^{R} \rho\left(\frac{\log(N_0(r, q)) - \log(q)}{\log(B)}\right) \rho\left(\frac{\log(N_1(r, q))}{\log(B)}\right) \mathrm{d}V_{2\eta}(r).$$

Then a sparse linear algebra phase computes the discrete logarithms of the factor basis for a cost that we estimate as $(2\,\mathrm{Li}(B))^2$, where Li is the logarithmic integral function. Similarly, the linear algebra cost for TNFS is $(2\,\mathrm{Li}(B))^2$.

We neglect the cost of the individual logarithm step in both cases. Anyway this cost is not only small, but also of roughly identical cost with both algorithms. Appendix B supports this statement.

Estimation. Figure 10 shows the cost estimates of TNFS and TNFS-factory. For TNFS Factory, we searched for parameters that minimize the cost of the *per-field* step, under the condition of having enough relations. For TNFS without Factory, we searched for parameters that minimize the sum of the costs of the relation collection and the linear algebra steps, under the condition of having enough relations.

When the factors of n are set to $\eta = 3$, $\kappa = 2$, the norm sizes are estimated to $\mathcal{N}_0(r, q) = S^{12}$ and $\mathcal{N}_1(r, q) = S^6 Q^{1/4}$. The norm $\mathcal{N}_0$ is too small compared to the norm on the other side and the factory algorithm considers the special-q on the side 0. This results in the *one-off* step doing the "easy" work. This is confirmed by Figure 10 in which we observe that the cost of TNFS is lower than the cost of the *one-off* step, which is itself lower than the cost of the *per-field* step. TNFS-Factory provides no benefit in this scenario.

When the factors of n are set to $\eta = 2$, $\kappa = 3$, the norm sizes are estimated to $\mathcal{N}_0(r, q) = S^{12}$ and $\mathcal{N}_1(r, q) = S^6 Q^{1/6}$. In this setup the norms are more balanced on both sides. This provides an interesting setup for the factory algorithm as shown in Figure 10.

Figure 10 shows that TNFS with the setup $\eta = 2$, $\kappa = 3$ is better that TNFS with the other setup for finite fields of sizes larger than 800 bits. Hence, TNFS-Factory is interesting when considering finite fields of size larger than 800 bits.

Best parameters for 1024-bit finite field $\mathbb{F}_{p^6}$. We denote $[q_{\min},\ q_{\max}]$ the special-q range. The best parameters we found for TNFS-Factory are $R = 196$, $q_{\min} \approx 2^{35.8}$, $q_{\max} \approx 2^{38.3}$, $B = 2^{33}$. As a consequence, our calculations show that the estimated cost of the *one-off* step is $2^{67.8}$, and the estimated cost of the *per-field* step is $2^{60.8}$.

For TNFS without Factory, the best parameters we found are $R = 138$, $q_{\min} \approx 2^{33.7}$, $q_{\max} \approx 2^{36.3}$, $B = 2^{35}$. We computed that this implies an estimated cost of TNFS around $2^{64.4}$.

What is the value of these estimates? Estimating the practical cost of NFS and its variants is a difficult problem and we do not claim to get precise results in this section. It would be possible to be more accurate. Actual computations in the 1000-bit range are out of reach at this point, but a middle ground could be to make the simulation more accurate by basing it on sample runs that closely follow the expected form of the input of the different stages of the algorithm. Unfortunately, the implementation available of TNFS does not yet allow such estimates. This is left for a future work.

Nevertheless, our approach (which follows, for example, what is done in [GS21]) tells more than if we content ourselves with the $L(1/3, c)$ estimates only, as is too often encountered. Furthermore, we believe that the qualitative comparison of TNFS versus TNFS Factory is likely to be modeled correctly by our approach. In that sense, since $2^{67.8}/(2^{64.4} - 2^{60.8}) \approx 11$, our estimation suggests that when considering some tens of finite fields $\mathbb{F}_{p^6}$ of size 1024 bits, TNFS Factory is more advantageous than applying TNFS on each of the target finite fields. To be precise, we also need to take into account the constraints on the decomposition of the primes in the number fields $\mathcal{K}_h$ and $\mathcal{K}_0$. If the Galois considerations of §3.4 predict that these constraints are met for, say, a fraction of $\frac{1}{12}$ of the primes, this means that the number of primes to consider for the Factory approach to be profitable is around 100.

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. Our practical estimates suggest that in the kilobit range, this Factory approach is more efficient than the non-Factory approach if several tens of finite fields are considered.

A given *one-off* step is only able to target a constant proportion of the finite field characteristics, but we show how this proportion can be computed. By combining a few *one-off* steps, it is possible to reach almost all primes without affecting the asymptotic complexity significantly.

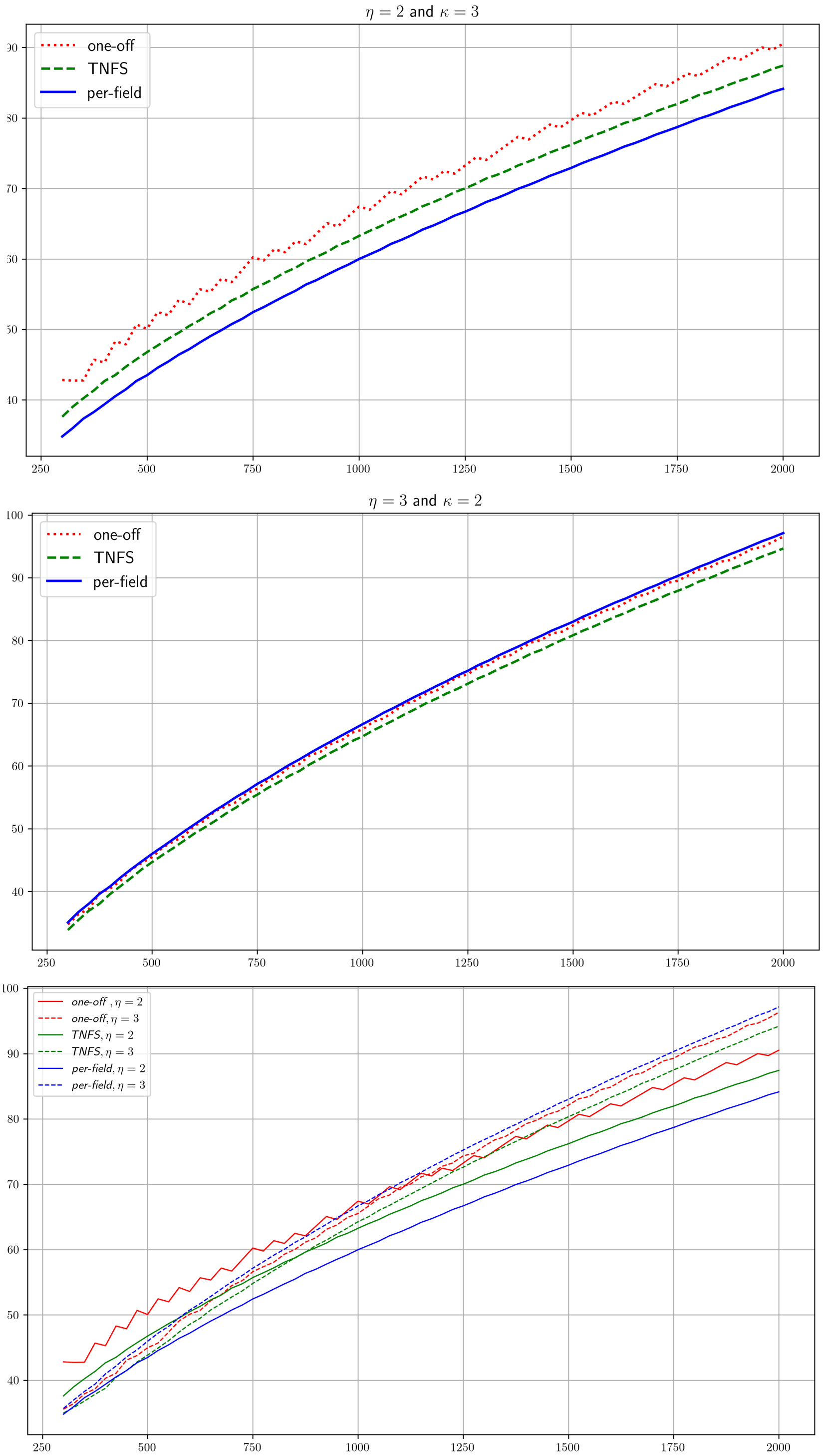


Fig. 10: Cost estimates of TNFS and TNFS Factory ($\log_2$ of the approximate number of operations) for finite fields of degree 6, as a function of the finite field size ($\log_2 Q$, in bits).

Furthermore, the flexibility provided by the potential trade-off between the costs of the *one-off* and the *per-field* steps 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, perhaps even in software like SageMath or Magma.

A drawback of Factory in practical use 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 alleviate this in the manner of the *Factoring Factory* algorithm, as explored in [BL14]. The memory requirements then become equivalent to those for NFS and its variants.

References

[AAP23] Haetham Al Aswad and Cécile Pierrot. Individual discrete logarithm with sublattice reduction. *Designs, Codes and Cryptography*, pages 1–33, 2023. doi:10.1007/s10623-023-01282-w.

[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*, 2015. doi:10.1145/2810103.2813707.

[Bar13] Razvan Barbulescu. *Algorithmes de logarithmes discrets dans les corps finis*. PhD thesis, Université de Lorraine, 2013. URL: https://hal.univ-lorraine.fr/tel-01750438.

[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 Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 62–91. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_3.

[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. doi:10.1007/978-3-662-46800-5_6.

[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. doi:10.1007/978-3-642-55220-5_1.

[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. doi:10.1007/978-3-319-13051-4_3.

[BLP93] J. P. Buhler, A. K. Lenstra, and C. Pomerance. Factoring integers with the number field sieve. In Lenstra and Lenstra, Jr. [LL93], pages 50–94. doi:10.1007/BFb0091539.

[BP14] Razvan Barbulescu and Cécile Pierrot. The Multiple Number Field Sieve for Medium and High Characteristic Finite Fields. *LMS Journal of Computation and Mathematics*, 17:230–246, 2014. URL: https://hal.inria.fr/hal-00952610, doi:10.1112/S1461157014000369.

[CEP83] E. Rodney Canfield, Paul Erdős, and Carl Pomerance. On a problem of Oppenheim concerning "factorisatio numerorum". *Journal of Number Theory*, 17(1):1–28, 1983. doi:10.1016/0022-314X(83)90002-1.

[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 Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Heidelberg, May 2020. doi:10.1007/978-3-030-45721-1_26.

[Cop93] Don Coppersmith. Modifications to the number field sieve. *Journal of Cryptology*, 6(3):169–180, March 1993. doi:10.1007/BF00198464.

[Cop94] Don Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comp.*, 62(205):333–350, 1994. doi:10.1090/S0025-5718-1994-1192970-7.

[DGP21] 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, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 67–96. Springer, Heidelberg, December 2021. doi:10.1007/978-3-030-92062-3_3.

[DM21] Gabrielle De Micheli. *Discrete Logarithm Cryptanalyses : Number Field Sieve and Lattice Tools for Side-Channel Attacks.* Theses, Université de Lorraine, May 2021. URL: https://hal.univ-lorraine.fr/tel-03335360.

[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. doi:10.1007/978-3-319-56620-7_8.

[GGM16] Pierrick Gaudry, Aurore Guillevic, and François Morain. Discrete logarithm record in GF(p^3) of 592 bits (180 decimal digits), August 2016. URL: https://listserv.nodak.edu/cgi-bin/wa.exe?A2=NMBRTHRY;ae418648.1608.

[GGM17] Laurent Grémy, Aurore Guillevic, and François Morain. Breaking DLP in $GF(p^5)$ using 3-dimensional sieving. working paper or preprint, July 2017. URL: https://inria.hal.science/hal-01568373.

[GKZ18] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel. On the discrete logarithm problem in finite fields of fixed characteristic. *Transactions of the American Mathematical Society*, 370(5):3129–3145, 2018. doi:10.1090/tran/7027.

[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. doi:10.1007/978-3-319-69453-5_30.

[Gor93] Daniel M. Gordon. Discrete logarithms in *GF(P)* using the number field sieve. *SIAM J. Discret. Math.*, 6(1):124–138, 1993. doi:10.1137/0406010.

[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 Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 305–326. Springer, Heidelberg, May 2016. doi:10.1007/978-3-662-49896-5_11.

[GS19] Aurore Guillevic and Shashank Singh. On the alpha value of polynomials in the tower number field sieve algorithm. Cryptology ePrint Archive, Report 2019/885, 2019. https://eprint.iacr.org/2019/885.

[GS21] Aurore Guillevic and Shashank Singh. On the alpha value of polynomials in the Tower Number Field Sieve algorithm. *Mathematical Cryptology*, 1(1):1–39, 2021. URL: https://journals.flvc.org/mathcryptology/article/view/125142.

[Gui19] Aurore Guillevic. Faster individual discrete logarithms in finite fields of composite extension degree. *Mathematics of Computation*, 88(317):1273–1301, January 2019. doi:10.1090/mcom/3376.

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

[HAKT14] Kenichiro Hayasaka, Kazumaro Aoki, Tetsutaro Kobayashi, and Tsuyoshi Takagi. An experiment of number field sieve for discrete logarithm problem over GF(p^n). *JSIAM Letters*, 6:53–56, 2014. doi:10.14495/jsiaml.6.53.

[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. doi:10.1007/11818175_19.

[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. doi:10.1007/978-3-319-04873-4_3.

[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. doi:10.1007/978-3-662-53018-4_20.

[KBL14] Thorsten Kleinjung, Joppe W. Bos, and Arjen K. Lenstra. Mersenne factorization factory. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 358–377. Springer, Heidelberg, December 2014. doi:10.1007/978-3-662-45611-8_19.

[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. doi:10.1007/978-3-662-54365-8_16.

[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. URL: https://hal.science/hal-03347994, doi:10.1090/jams/985.

[LGST21] Aude Le Gluher, Pierre-Jean Spaenlehauer, and Emmanuel Thomé. Refined analysis of the asymptotic complexity of the number field sieve. *Mathematical Cryptology*, 1(1):71–88, 2021. URL: https://journals.flvc.org/mathcryptology/article/view/125488.

[LL93] A. K. Lenstra and H. W. Lenstra, Jr., editors. *The development of the number field sieve*, volume 1554 of *Lecture Notes in Math.* Springer–Verlag, 1993. doi:10.1007/BFb0091534.

[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. doi:10.1145/100216.100295.

[LO77] J. C. Lagarias and A. M. Odlyzko. Effective versions of the Chebotarev density theorem. In A. Fröhlich, editor, *Algebraic Number Fields: L functions and Galois properties (Proc. Sympos., Univ. Durham, Durham, 1975)*, pages 409–464. Academic Press, 1977.

[LV01] Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, September 2001. doi:10.1007/s00145-001-0009-4.

[Mat03] Dimitry Matyukhin. On asymptotic complexity of computing discrete logarithms over GF(p). *Discrete Mathematics and Applications*, 13:27–50, 2003. doi:10.1515/156939203321669546.

[Mil20] James S. Milne. Algebraic number theory (v3.08), 2020. URL: https://www.jmilne.org/math/.

[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. doi:10.1007/978-3-662-46800-5_7.

[Pol93] John M. Pollard. The lattice sieve. In Lenstra and Lenstra, Jr. [LL93], pages 43–49. doi:doi.org/10.1007/BFb0091538.

[Rob22] Oisín Robinson. An implementation of the extended tower number field sieve using 4d sieving in a box and a record computation in $\mathbb{F}_{p^4}$. arXiv preprint 2212.04999, 2022. doi:10.48550/arXiv.2212.04999.

[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. doi:10.1007/978-3-662-53887-6_2.

[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. doi:10.1007/978-3-662-49890-3_17.

[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. doi:10.3934/amc.2019028.

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

A Proofs of Theorem 1 and Theorem 2

A.1 The boundary case $\alpha = 2/3$ in Theorem 1

We are here in the regime where $Q^{1/n} = L(1/3, c_p)$ for some c_p.

A.1.1 The boundary case with GJL (case 2a).

The asymptotic analysis in the large characteristic case applies as soon as the parameter $d = 1/\gamma\left(\log(Q)/\log(\log(Q))\right)^{1/3}$ is larger than or equal to $n = 1/c_p\left(\log(Q)/\log(\log(Q))\right)^{1-\alpha}$, which is equivalent to $c_p \geq \gamma$ since $1-\alpha = 1/3$. For this range of finite fields, we get exactly the same asymptotic complexities as in the large characteristic case.

Remark 4. Unlike the Conjugation case that benefits from increasing the sieve dimension t, analysis shows that such a strategy does not pay off with GJL.

A.1.2 The boundary case with Conjugation (case 2b).

The polynomials output by the Conjugation method have degrees $2n$ and n, and coefficient sizes as in Table 6. Let $t \in \mathbb{Z}$ be the sieve dimension. The norms of the sieve elements are $\mathcal{N}_0 = \widetilde{O}\left(A^{(2n)/t}\right) = L_Q\left(2/3, 2c_A/(c_p t)\right)$ and $\mathcal{N}_* = \widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}\right) = L_Q\left(2/3, c_A/(c_p t) + (t-1)c_p/2\right)$. 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:

$$18c_p t X^3 - 24X^2 - 3c_p^2 t(t-1)X + 2c_p(t-1) = 0 \tag{10}$$

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

We want to know how much leeway we have in the choice of p_i. The size of p_i only affects $\mathcal{N}_*$. If we change the asymptotic expression of p_i to $p_i^n \leq Q^{1+o(1)}$ (instead of $p_i^n \approx Q$), then $\mathcal{N}_*$ merely increases to $\widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}p^{o(1)}\right)$. Since, $p^{o(1)}$ is negligible compared to any function in $L_Q(2/3)$, the asymptotic results above are unchanged. (A similar observation applies to the other cases as well.)

A.2 The medium characteristic case $1/3 < \alpha < 2/3$ in Theorem 1

A.2.1 NFS Factory (case 3a).

Let $t = \delta n\left(\log(Q)/\log(\log(Q))\right)^{-1/3+o(1)}$ be the sieve dimension, for a positive constant δ. As Q tends to infinity, t also tends to infinity, so that the constraint that $t \in \mathbb{Z}$ is absorbed by the $o(1)$ in the exponent. The coefficients of the sieve elements are bounded by $A^{1/t}$. Their norms can be expressed as $\mathcal{N}_0 = \widetilde{O}\left(A^{(2n)/t}\right) = L_Q\left(2/3, 2c_A/\delta\right)$ and $\mathcal{N}_* = \widetilde{O}\left(A^{n/t}Q^{(t-1)/(2n)}\right) = L_Q\left(2/3, c_A/\delta + \delta/2\right)$. If we inject these expressions of $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_*}$ in system given by Constraint (5) and Conditions (6), and (7), we obtain

$$c = \left(\frac{1+\sqrt{2}}{3}\right)^{2/3} \approx 0.87, \quad \delta = \frac{2\sqrt{2}}{c} \approx 2.63, \quad c_A = 2c\sqrt{2} \approx 2.45,$$

from which the claimed results follow.

A.2.2 TNFS Factory (case 3b).

We only consider the case where η and κ are coprime. The general case is similar. Let $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q)))^{1/3+o(1)}$ with c_κ a constant. As Q tends to infinity, κ also tends to infinity, so that the constraint that κ is an integer divisor of n can be absorbed by the $o(1)$ in the exponent, provided of course that the input is such that n has such a factor. The sieve is done over elements of the form $a(\iota)X - b(\iota) \in \mathcal{O}_{\mathcal{K}_h}[X]$ with $a(\iota)$ and $b(\iota)$ in $\mathbb{Z}[\iota]$ of degree at most $\eta - 1$. According to the norm bounds of §2.3 (with $\deg_x \phi = 1$ and $\deg_y \phi = \eta - 1$), we have $\mathcal{N}_i(\phi) = \widetilde{O}(\|\phi\|_\infty^{\eta \deg(f_i)} \|f_i\|_\infty^{\eta} \|h\|_\infty^{(\eta-1)\deg(f_i)})$ for all $i \geq 0$. More precisely, since $\|\phi\|_\infty \leq A^{1/(2\eta)}$, we get $\mathcal{N}_0 = \widetilde{O}(A^\kappa) = L_Q(2/3, c_A/c_\kappa)$ and $\mathcal{N}_* = \widetilde{O}(A^{\kappa/2}Q^{1/(2\kappa)}) = L_Q(2/3, c_A/(2c_\kappa) + c_\kappa/2)$. These expressions of $c_{\mathcal{N}_0}$ and $c_{\mathcal{N}_*}$ yield the following optimum:

$$c = \frac{1}{2}\left(\frac{2+2\sqrt{2}}{9}\right)^{\frac{2}{3}} \approx 0.69, \quad c_\kappa = 2\sqrt{c} \approx 1.66, \quad c_A = 2c\sqrt{2} \approx 1.94.$$

A.3 The large characteristic case in Theorem 2 (case 1a)

We consider the polynomials of the first approach of Joux–Pierrot, as in §3.3. Hence $\mathcal{N}_0 = \widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)})$ and $\mathcal{N}_* = \widetilde{O}(A^{\lambda n/t}\log(n)^{\lambda(t-1)})$. Let

$$\lambda = 1/(c_\lambda n)(\log(Q)/\log\log(Q))^{1/3}$$

with c_λ a constant. The norm of the sieve elements are $\mathcal{N}_0 = L_Q(2/3, c_\lambda)$ and $\mathcal{N}_* = L_Q(2/3, c_A/(2c_\lambda))$, since $\log(n)^\lambda$ is negligible compared to $L_Q(\alpha_p - 2/3)$, and $\alpha_p - 2/3 \leq 1/3 < 2/3$. From Condition (7) we get $c_A = 2c + c_\lambda/(3c)$. Substituting c_A in Condition (6), we get $c_\lambda \geq 6c^2/(18c^3-1)$. 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, and get

$$c = \left(\frac{1}{3}\right)^{1/3} \approx 0.69, \quad c_\lambda = \left(\frac{8}{9}\right)^{1/3} \approx 0.96, \quad c_A = 2\left(\frac{8}{9}\right)^{2/3} \approx 1.85.$$

A.4 The medium characteristic case $1/3 < \alpha < 2/3$ in Theorem 2

A.4.1 SNFS Factory (case 2a).

The polynomials are chosen with the second approach of the Joux–Pierrot method of §3.3. Hence $\mathcal{N}_0 = \widetilde{O}(A^{\lambda n/t}\log(n)^{\lambda(t-1)})$ and $\mathcal{N}_* = \widetilde{O}(A^{n/t}Q^{(t-1)/(n\lambda)})$. We set $t = \delta n(\log(Q)/\log\log(Q))^{-1/3+o(1)}$. 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 compared to $L_Q(2/3)$, and $\mathcal{N}_* = L_Q(2/3, c_A/\delta + \delta/\lambda)$. A solution of the system is:

$$c \geq \tilde{c} = \left(\frac{\lambda + 4 + 2\sqrt{2\lambda+4}}{9\lambda}\right)^{1/3}, \tag{11}$$

$$c_A = \frac{6c^2\delta}{3c\delta - \lambda},$$

$$\delta = \frac{\lambda(9c^3+1) + \sqrt{-27\lambda c^3 + \lambda^2(81c^6 - 18c^3 + 1)}}{6c}.$$

When $\lambda \in \{2,3\}$, we set $c = \tilde{c}$. However, when $\lambda \in \{4,5\}$, only the relation collection and linear algebra steps of the *per-field* step can reach complexity $L_Q(1/3, \tilde{c})$. As we show in Appendix B, the individual logarithm step is unfortunately more expensive, and we need to take c somewhat larger than $\tilde{c}$ in order to keep the individual logarithm step negligible. Table 8 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 *per-field* step is $L_Q(1/3, 2c)$.

A.4.2 STNFS Factory (case 2b).

The Special Tower variant targets medium characteristic finite fields of sparse characteristic p, and composite extension degree $n = \kappa\eta$. Let $\kappa = 1/c_\kappa(\log(Q)/\log\log(Q))^{1/3+o(1)}$ for some constant c_κ to be determined. Consider $p_i = P(u)$, where P is a polynomial of degree λ with small coefficients, and $u \approx Q^{1/(\lambda n)}$ an integer. Again we assume that κ and η are coprime. The polynomials are chosen with the second approach of the Joux–Pierrot method as in §3.3. The bound of §2.3 gives $N_0 = \widetilde{O}(A^{\lambda\kappa/2}\log(\kappa)^\lambda) = L_Q(2/3, \lambda c_A/(2c_\kappa))$ and $\mathcal{N}_* = \widetilde{O}(A^{\kappa/2}Q^{1/(\lambda\kappa)}) = L_Q(2/3, c_A/(2c_\kappa) + c_\kappa/\lambda)$. The solution of the system related to Equations (5), (6), and (7) is

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

When $\lambda = 2$, we set $c = \tilde{c}$. However, when $\lambda \geq 3$, the situation is similar to SNFS Factory, and we need to take c larger than $\tilde{c}$ in order to keep the individual logarithm step negligible (see Appendix B). 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 *per-field* step is $L_Q(1/3, 2c)$.

B Complexity of the Individual Logarithm step

The individual logarithm step is the last one in NFS and its variants, and also the last one inside the *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 *per-field* step for all the variants we studied. Hence, the complexities given in Table 3 are indeed the complete asymptotic complexities of the *per-field* step. The individual logarithm step consists of two main steps: the smoothing and the descent step.

B.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 [Gui19, AAP23]. The following lemma recapitulates the complexity of the smoothing step for all Factory variants. This result implies that the smoothing step is negligible compared to the complexity of the *per-field*, for all Factory variants.

Lemma 1. *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 [Gui19], where substituting e and d by 1 is valid for all our Factory variants. □

B.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 1. Thanks to the previous steps, we know the virtual logarithms of the prime ideals in $\mathcal{O}_{\mathcal{K}_0}$ that are both factors of the target and of norm below B. It remains to compute the virtual logarithms of those of norm between B and $\widetilde{B}$. Let $\mathfrak{q}$ be such a prime ideal, of degree one and norm q. 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 the ideal $(a(\iota) - b(\iota)\alpha_0)$ is divisible by $\mathfrak{q}$. 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, 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 :

$$\begin{aligned} \frac{\mathcal{N}_0\left(b(\iota) - a(\iota)\alpha_0\right)}{q} \quad & \text{is } q^\xi\text{-smooth} \\ \text{and } \mathcal{N}_i\left(b(\iota) - a(\iota)\alpha_1\right) \quad & \text{is } q^\xi\text{-smooth.} \end{aligned}$$

This permits to express the virtual logarithm of $\mathfrak{q}$ as a linear combination of virtual logarithms of prime ideals of norm 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 [Gui19], 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.

B.2.1 Complexity of the descent step for NFS Factory and its variants.

For $\mu = (\mu_0, \ldots, \mu_{2\eta-1})$ of infinity 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 §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 $1/3 \le \alpha_q \le 2/3$, 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 *GJL* polynomials, 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 §2.3) times the cost of ECM. Thus this complexity is:

$$\begin{aligned} L_Q\left(\frac{1}{3}, \frac{s}{3\gamma\xi c_q} + \frac{\gamma}{3\xi c_q} + \frac{1}{3\xi\gamma}\right) \quad & \text{if } \alpha_q = \tfrac{1}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{3\xi\gamma}\right) \quad & \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{3\xi\gamma} + \sqrt{\frac{4\xi c_q}{3}}\right) \quad & \text{if } \alpha_q = \tfrac{2}{3}. \end{aligned}$$

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 follows 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 $p_i \approx Q^{1/n} = L_Q(2/3, c_p)$, 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.

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

Table 11: Asymptotic complexities of the individual logarithm step and the *per-field* step in NFS Factory and its variants. This table recaps approximations of c when the complexities are expressed as $L_Q(1/3, c)$.

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		

beginning of §2.3 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:

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

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 $\tilde{\delta} n(\log(Q)/\log\log(Q))^{-1/3+o(1)}$. 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:

$$\begin{aligned}
L_Q\left(\frac{1}{3}, \frac{s}{\xi c_q \tilde{\delta}} + \frac{\tilde{\delta}}{6\xi c_q} + \frac{1}{\xi \tilde{\delta}}\right) \quad & \text{if } \alpha_q = \tfrac{1}{3}, \\
L_Q\left(\frac{1}{3}, \frac{1}{\xi \tilde{\delta}}\right) \quad & \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\
L_Q\left(\frac{1}{3}, \frac{1}{\xi \tilde{\delta}} + \sqrt{\frac{4\xi c_q}{3}}\right) \quad & \text{if } \alpha_q = \tfrac{2}{3}.
\end{aligned}$$

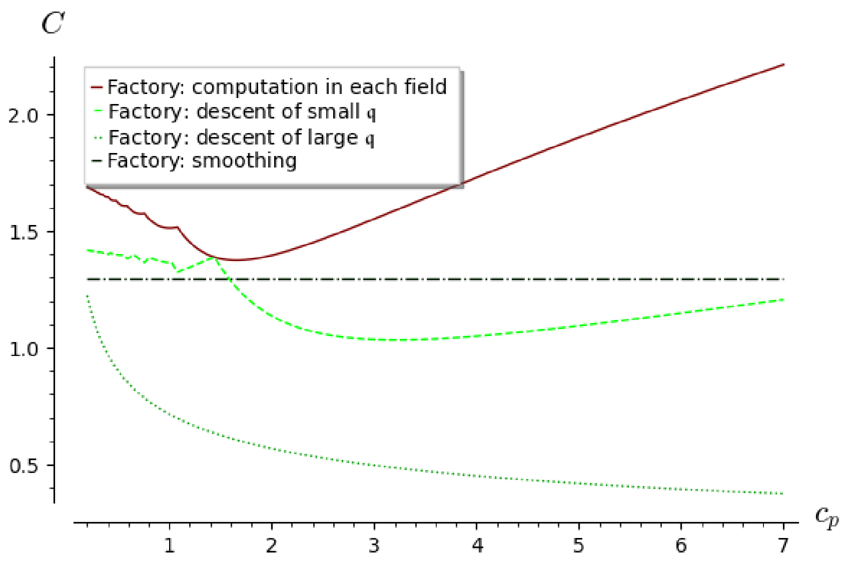


Fig. 12: Asymptotic complexities of some steps inside NFS Factory at the boundary case. Target finite fields have characteristic p such that $p = L_{p^n}(2/3, c_p)$. This graph shows how c varies as a function of c_p when the complexities are expressed as $L_{p^n}(1/3, c)$.

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 *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:

$$\begin{aligned} L_Q\left(\frac{1}{3}, \frac{s}{2\xi c_q c_\kappa} + \frac{c_\kappa}{6\xi c_q} + \frac{1}{2\xi c_\kappa}\right) & \quad \text{if } \alpha_q = \tfrac{1}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{2\xi c_\kappa}\right) & \quad \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{2\xi c_\kappa} + \sqrt{\frac{4\xi c_q}{3}}\right) & \quad \text{if } \alpha_q = \tfrac{2}{3}. \end{aligned}$$

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

$$N = \widetilde{O}((S^2)^{n(\lambda+1)/2} Q^{1/(\lambda n)} q^{n(\lambda+1)/2-1}).$$

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.

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**

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:

$$\begin{aligned} L_Q\left(\frac{1}{3}, \frac{s}{6\xi c_q c_\lambda} + \frac{c_\lambda}{3\xi c_q} + \frac{1}{6\xi c_\lambda}\right) &\quad \text{if } \alpha_q = \tfrac{1}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{6\xi c_\lambda}\right) &\quad \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\ L_Q\left(\frac{1}{3}, \frac{1}{6\xi c_\lambda} + \sqrt{\frac{4\xi c_q}{3}}\right) &\quad \text{if } \alpha_q = \tfrac{2}{3}. \end{aligned}$$

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 $\tilde{\delta} n(\log(Q)/\log\log(Q))^{-1/3+o(1)}$. Therefore, $\eta = 1$, the dimension of $\mathcal{L}_\mathfrak{q}$ is $\tilde{t}$ and its determinant is q. We use the formula for N of §2.3, with the properties of the polynomials output by the Joux–Pierrot method. Writing $S^{\tilde{t}} = L_Q(1/3, s)$, 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:

$$\begin{aligned} L_Q\left(\frac{1}{3}, \frac{s(\lambda+1)}{3\xi c_q \tilde{\delta}} + \frac{\tilde{\delta}}{3\xi c_q \lambda} + \frac{\lambda+1}{3\xi\tilde{\delta}}\right) &\quad \text{if } \alpha_q = \tfrac{1}{3}, \\ L_Q\left(\frac{1}{3}, \frac{\lambda+1}{3\xi\tilde{\delta}}\right) &\quad \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\ L_Q\left(\frac{1}{3}, \frac{\lambda+1}{3\xi\tilde{\delta}} + \sqrt{\frac{4\xi c_q}{3}}\right) &\quad \text{if } \alpha_q = \tfrac{2}{3}. \end{aligned}$$

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 *per-field* step.

Medium characteristic descent step for STNFS Factory. With Joux–Pierrot selection and the usual notations, we get

$$N = \widetilde{O}((S^{2\eta})^{\kappa(\lambda+1)/2} Q^{1/(\lambda\kappa)} q^{\kappa(\lambda+1)/2-1}).$$

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.

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**

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:

$$\begin{aligned} L_Q\left(\frac{1}{3}, \frac{s(\lambda+1)}{6\xi c_q c_\kappa} + \frac{c_\kappa}{3\xi c_q \lambda} + \frac{\lambda+1}{6\xi c_\kappa}\right) \quad & \text{if } \alpha_q = \tfrac{1}{3}, \\ L_Q\left(\frac{1}{3}, \frac{\lambda+1}{6\xi c_\kappa}\right) \quad & \text{if } \tfrac{1}{3} < \alpha_q < \tfrac{2}{3}, \\ L_Q\left(\frac{1}{3}, \frac{\lambda+1}{6\xi c_\kappa} + \sqrt{\frac{4\xi c_q}{3}}\right) \quad & \text{if } \alpha_q = \tfrac{2}{3}. \end{aligned}$$

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.