A note on the behaviour of the NFS in the medium prime case: Smoothness of Norms

Naomi Benger[1], Manuel Charlemagne[2⋆], and Kefei Chen[3]

[1] University of Adelaide, Australia
naomi.benger@adelaide.edu.au
[2] Shanghai Jiao Tong University, China
charlem@sjtu.edu.cn
[3] Hangzhou Normal University, China
chen-kf@cs.sjtu.edu.cn

Abstract. During an ongoing examination of the behaviour, in practice, of the Number Field Sieve (NFS) in the medium prime case we have noticed numerous interesting patterns. In this paper we present findings on run-time observations of one aspect of the sieving stage. The contributions of these observations to the computational mathematics community are twofold: firstly, they bring us a step closer to understanding the true practical effectiveness of the algorithm and secondly, they enabled the development of a test for the effectiveness of the polynomials used in the NFS. The results of this work are of particular interest to cryptographers: the run-time of the NFS determines directly the security level of some discrete logarithm problem based protocols, such as those arising in pairing-based cryptography.

1 Introduction and Motivation

In cryptography the *computational* behaviour of particular algorithms is used to determine the security level of a protocol. More specifically, as the security of cryptographic protocols is based on the hardness of underlying mathematical problems, it follows that the security is reliant on the computational effort required to solve those problems in the given implementation context. It is therefore important that we have a sound knowledge of the behaviour of the algorithms used to solve these problems in practice in order to implement cryptographic protocols with reliable security settings.

[⋆] Founded by the National Natural Science Foundation of China (Grant 61133014) and the State Key Laboratory of Mathematical Engineering and Advanced Computing.

The Discrete Logarithm Problem is the basis of security in numerous cryptographic protocols instantiated in either the additive group of points on an elliptic curve (ECDLP) or a multiplicative group of a finite extension field (DLP). The most efficient known algorithms to solve the ECDLP have been thoroughly examined and the behaviour of these algorithms in practice is well understood, for example see [20, 22, 3, 4, 2]. The DLP in medium prime extension fields has received comparably less attention, so the understanding of the behaviour of the algorithm to solve it is limited. The result of this is that we are not able to give precise implementation guidelines for protocols with security dependent on the hardness of this DLP instance. In particular, in Pairing-Based Cryptography (PBC) the underlying problems that form the basis for security are usually related to variations of both the ECDLP and DLP – the hardness of the first problem is well understood but hardness of the second still remains elusive; this complicates parameter selection.

There are various known algorithms for solving the DLP, the most appropriate depends on the context of the problem. According to the complexity analysis in [14], the Number Field Sieve (NFS) in the medium prime case is the most efficient algorithm to solve the instances of finite field DLP occurring in PBC; these instances are in the "small prime" category [14, Section 3.1].

Little is known about the practical effectiveness of the NFS in the medium prime case which limits our understanding of the security of PBC protocols. In this work we examine one aspect of this NFS version.

The ultimate goal of this continuing project is to increase the general understanding of the behaviour of the NFS. The motivation for this work is threefold: firstly, the behaviour of other NFS versions is known to vary widely [23, 8]. Secondly, implementation of cryptographic protocols on small devices requires accurate security estimates as memory and processing power are limited. Finally, recent advances in solving the DLP in binary fields [11, 12, 15] will promote the use of medium prime fields in PBC, over binary fields; it is necessary to investigate the problem in this increasingly utilised setting.

The analysis of the NFS algorithm in [14] is given for finite fields $\mathbb{F}_{p^n}$ as both p and n tend to infinity which necessitates loss of detail and generalisation to limiting cases. We wish to fine-tune this analysis, model the behaviour of the NFS in the context of PBC, retaining as much detail as possible, performing statistical analysis on experimental results. In this work we outline some of the practical observations made during our analysis. Using these observations we have developed a “pre-NFS” polynomial test on the effectiveness of a particular NFS instance. We also introduce a variation in the NFS polynomial selection. The results of our work will be of interest to implementers of the NFS and pairing-based protocols alike.

The remainder of this paper is organised as follows: In section 2 we give a brief introduction to the NFS. In section 3 we examine one aspect of the theoretical complexity analysis, which solidifies our motivation and introduce the particular aspect of the NFS that this work focuses on. In section 4 we explain our experiment setting and the process of our investigation, presenting the results of this undertaking in section 5. Conclusions and open problems are presented in section 6.

2 NFS in a nutshell

In this section we give a brief introduction to the NFS. The NFS can be considered a sequence of stages for which we assess the complexity separately. The two main parts of the NFS are the sieving stage and the linear algebra stage, these are followed by a descent stage. Suppose we wish to solve a DLP instance in the finite field $\mathbb{F}_{p^n}$.

In the sieving stage we aim to construct a set of linear equations in the logarithms of “small” elements. This is done using two number fields, say $\mathcal{K}_1$ and $\mathcal{K}_2$, both of which contain $\mathbb{F}_{p^n}$ as a subfield. We sieve through the elements of these fields looking for “doubly-smooth” pairs. Two elements form a *pair* if the mappings from their respective number fields to $\mathbb{F}_{p^n}$ maps them to the same element. An element is *smooth* if it can be written as the product of *small* elements, elements with norms a prime number below a given bound; the pair is *doubly-smooth* if both are smooth in their respective number fields.

The system of linear equations is solved in the linear algebra stage, finding the discrete logarithms of the small elements. In the descent stage the specific element of interest is written as a product of small elements and thus its discrete logarithm is found.

In this work we examine the complexity of the sieving stage. For the NFS variation relevant to the PBC context, the two number fields are constructed as $\mathcal{K}_i = \mathbb{Q}[\theta_i]$ for $i \in \{1, 2\}$. The element θ_i is a zero of the irreducible polynomial $f_i(x)$ with integer coefficients such that $n \mid \deg(f_i)$; f_1 and f_2 have a common root modulo p. The elements of these number fields are thus considered as polynomials of degree $t < n$ in θ_i. To compute the norm of an element $a = a(\theta_i)$ in $\mathcal{K}_i$ we consider a as a polynomial in x and take the resultant of $a(x)$ with $f_i(x)$. To find doubly-smooth pairs we look at the factorisation of the norms of the elements.

3 Smoothness probability of norms

The run time of the sieving stage of the NFS is determined by the probability of finding doubly smooth relations. There exist varying methods in the literature to predict smoothness probabilities in different contexts. Denote by $\Psi(x, y)$ the number of integers $\leq x$ with no prime factor exceeding y; the probability of a random integer of size approximately the size of x being *y-smooth* is $\Psi(x, y)/x$. In [13] a comprehensive survey of estimates for $\Psi(x, y)$ is given; in the more recent [7] we are presented with a method for computing Ψ within an arbitrarily tight bound. The results of [17] are extensions of some of the theorems in [13] to the context of algebraic numbr fields – the relevant case for the NFS. The formulæ for calculating the probability is directly obtained from the $\Psi_K(x, y)$ estimate (number of integral ideals with norm $< x$, all of whose prime divisors have norm $< y$ in a number field K) [17, Satz 3]. Computation of this estimate is not feasible; it relies on knowledge of the class number of K and a result of the Dedekind Zeta function, both of which are known to be hard to compute in practice. This estimate thus cannot be used in the analysis of the NFS, a major set back when trying to estimate the computational behaviour of the NFS. Indeed, the variability of the probability of smoothness of norms in given number fields (that is, for a given choice of f_1 and f_2) could be responsible for some of the behaviour noticed by Zajac [23]. The ability to compute this probability would aid in the selection of the polynomial pair (f_1, f_2). The smoothness probability estimate for integers used for the complexity analysis in [14] is from a corollary [10, page 15]; this formula is the most appropriate choice, given that the algorithm is presented for extension degrees k and primes p both tending to infinity. We will show in Section 5 that this estimate can be significantly improved apon for the PBC context.

4 Analysis Context

In this section we outline our experiment parameters and a variation of the NFS polynomial selection process.

Parameter sizes

As the contextual focus is PBC, we used the current field size suggestions in literature (from resources such as [19, 18, 1]) for the security levels 80, 128 and 192 which fixes the types of elliptic curves which can be used and therefore also fixes n. We used the complexity analysis of [14] to compute the corresponding values for t (degree of elements sieved) for each of the 3 instances. That gives us:

	80 bit security	128 bit security	192 bit security
curve	MNT6	BN	KSS18
size of p	160	256	512
n	6	12	18
t	2	3-4	4

We present our results using a working example of the MNT6 case; the methods are directly adaptable to the BN and KSS18 cases.

4.1 Polynomial selection

In order to have the NFS polynomials of simple structure, we find two integers a_1, a_2 of size $\sim \sqrt{p}$ of such that $a_1 \cdot a_2 = p + i$, where i is some small integer. We define the polynomials f_1 and f_2:

$$\begin{aligned} f_1 &= x^n + a_2 \\ f_2 &= a_1 x^n + i \end{aligned}$$

when both are irreducible. It is straightforward to show that these polynomials have a common zero modulo p (in fact have the same set of zeros) and are therefore appropriate for use in the NFS. Using this polynomial selection means that the sieving space did not need to be skewed to balance the sizes of the norms of the elements in each number field (see [14] for details). One important detail supporting the use of binomials in this context is that for efficient pairing implementation the prime is chosen to be $p \equiv 1 \mod n$ (or the prime factors of n) [16], thus it is possible to find irreducible binomials for the NFS polynomial selection in this context. (For the BN case we were able to use irreducible polynomials as used for the tower constructions in [5].)

5 Norm Smoothness Analysis

In this section we present our methodology and the results of our work; the process is divided into three sections:

5.1 We used empirical data to compute the cumulative distribution function of the norms; we make observations about which parameters influence the distribution; we illustrate how the cumulative distribution affects the smoothness probability.

5.2 We show how to compute an improved estimate of the smoothness probability using the cumulative distribution function and the results of [7].

5.3 We present a test for the effectiveness of the f_1 and f_2 polynomials against the expected outcome.

5.1 Cumulative Distribution of Norms

Once the polynomials f_1 and f_2 have been selected, the computation of the norms can be reduced to the evaluation of an integer polynomial. The arguments are taken from a fixed interval $[0, S]$ for a sieving bound S, computed following the instructions in [14], as is the smoothness bound $\mathcal{B}$. We consider the arguments as independent, uniformly distributed, random variables $X_0, \dots, X_t$ from $[0, S]$. The norm is also a random variable, N, which takes values in the range of the function on the random variables $X_0, \dots, X_t$ defined by the determinant of the Sylvester matrix; we denote this function $n(X_0, \dots, X_t)$. We are interested in the cumulative distribution of

$N = n$. The reason for focussing on the cumulative distribution is that the estimates for the probability of smoothness are in fact cumulative probabilities: $\Psi(x, y)/x$ where $\Psi(x, y)$ is the number of integers $\leq x$ with no prime factor exceeding y. We will use the cumulative distribution of N to weight the smoothness probability estimate of the integers to find a more appropriate smoothness probability estimate for the norms. The resultant of polynomials of the form Ax^6+B with $X_0+X_1x+X_2x^2$ (the MNT6 setting) is given by the evaluation of the function

$$n(X_0, X_1, X_2, X_3) = A^2X_0^6-2ABX_0^3X_2^3+9ABX_0^2X_1^2X_2^2-6ABX_0X_1^4X_2+ABX_1^6+B^2X_2^6,$$

where $A = 1$ for f_1. Such a number clearly has much structure and numbers generated using this equation will have a distribution very different from the uniform, the assumption used in the complexity analysis. Trialling various theoretical methods to determine the probability distribution of this N from this function was unfruitful due to the complexity and large number of variables. We therefore generated empirical data to directly determine the cumulative distribution. In the following graph we see the cumulative distribution function as generated for the norms for the MNT6 case. Intuitively, we expect a “clumping” effect in the centre of the probability distribution of the norms due to the fact that we are repeatedly summing 6th degree products of uniform variables from a bounded interval, which will increase the proportion (and therefore probability) of the low to middle range norm values. The result of this clumping effect in the probability distribution translates to a very steep cumulative distribution function over the low and middle range (plentiful) values, which tapers down to very small slope over the larger (rarer) values. This is exactly the result that the data presented. In figure 5.1 we see the cumulative distribution function as generated for the norms for the MNT6 case compared with the uniformly distributed variable. It is clear that this central clumping as discussed above results in the “mid-range” values being more probable in the norm distribution than for the integers: The number of integers in a particular interval increases constantly with the upper bound of the interval whereas the norms have a higher concentration in the mid-range of the same interval, a direct result of the range of the variables X_0, X_1, X_2. Another interesting observation we made, by varying the size of p for more tests, is that the distribution of the norms depends only on n and t. The size of p only dictates the

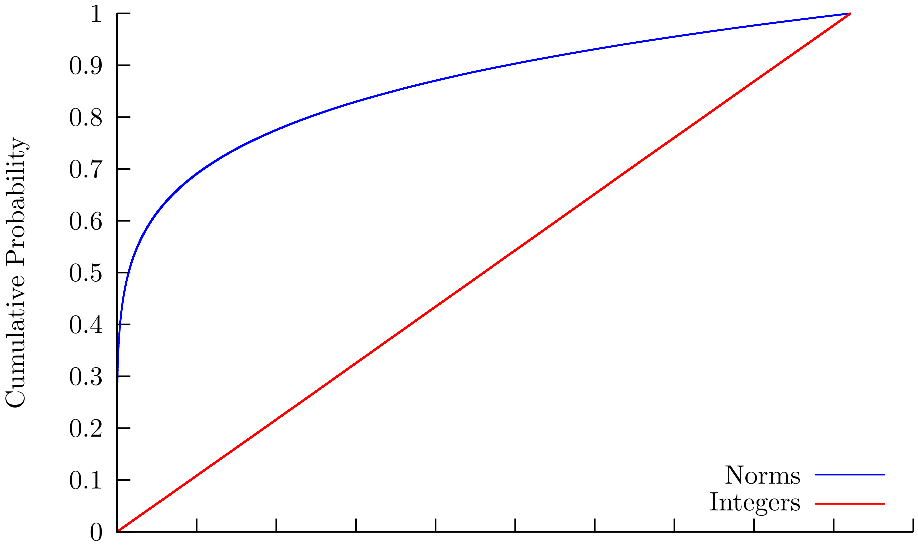


Fig. 5.1. Comparison of the cumulative distribution of norms and uniform integers in the MNT6 case. This particular graph generated using $p = 1125909838976401$ ($s = 349879$).

placement of the distribution not the *shape*. This is not surprising given that the shape of the elements of interest are the output of the above function $n(X_0, \ldots, X_t)$, a degree n function in t variables. For this reason we use a smaller example case than would be used in practice: the MNT6 setting and a 50-bit prime (with $n = 6$ and $t = 2$). This allows us to perform more thorough and varied tests. We recalculated the sieving bound to correspond with the new field size, following the instructions in [14], but preserved the value for t.

Fitting a line to the cumulative distribution The line fitted to the cumulative distribution as presented in figure 5.1 was generated using 2 million norm values (X_0, X_1, X_2 chosen uniformly at random) for a range of primes and polynomials. To obtain the equation of this line we performed curve fitting using R, a language package for statistical computing [21]. Initial experiments indicated that this is not an exponential curve prompted us to use the *Box-Cox method* [9]. This method aids us in identifying a suitable power transformation on the response variable ($y =$ the proportion of norm values $\leq x$) to obtain a linear relationship between the explanatory variable (x)

and the modified response variable; that is, we wish to find a value λ such that $y^\lambda = ax + b$. Often the initial test range for λ is $[-5, 5]$ and the default setting in R is $[-2, 2]$ incrementing in steps of 0.1. The graph in fig 5.1 represents the density of norm values and as the norms are sums of monomials of degree 6 we expect λ to be close to 6 and therefore set our test range to be $[0, 10]$. Instead of using all 2 million points (which would have made the tests very slow) we repeated the test numerous times on random samples of 10000 points. The results for each of the tests gave clear indication that $\lambda = 6$ (see appendix A.1). We then proceed to fit a linear model to the full data set of points of the transformed data (x, y^6) to find that $Y^6 = aX + b$ where $a = 2.171E - 106$ and $b = -1.868E - 5$ with negligible variance. From a statistical perspective the fit is incredibly accurate as there is no noise in the data. Exact output of the linear model fitted in R is given in appendix A.1

How the cumulative distribution affects the smoothness probability To illustrate why the probability of smoothness is affected by the differing probability distributions we use the law of total probability. Let us examine the problem in terms of random variables: Let Z be a random variable which takes integer values from the interval $[0, U]$. Let N be a randomly selected norm (also from the interval $[0, U]$) and let S_Z (resp. S_N) be a binomial random variable such that

$$S_Z = \begin{cases} 1 \text{ if } Z = z \text{ is smooth;} \\ 0 \text{ if } Z = z \text{ is not smooth.} \end{cases} \tag{5.1}$$

We define S_N similarly where smooth in both cases is with respect to some fixed bound $\mathcal{S}$. Now the probability that a random integer z selected from the defined interval uniformly at random is smooth is expressed as

$$P(S_Z = 1|Z = z).$$

Following the law of total probability we partition the interval $[0, U]$ into s equally sized sections, of width $U/s = a$, we can now write this probability as a summation:

$$P(S_Z = 1) = \sum_{i=1}^{s} P(S_Z = 1|z \in [(i-1)a, ia])P(z \in [(i-1)a, ia]).$$

Using the same method we can rewrite $P(S_N = 1)$:

$$P(S_N = 1) = \sum_{i=1}^{s} P(S_N = 1 | n \in [(i-1)a, ia])P(n \in [(i-1)a, ia]).$$

Assuming initially that the probability of a norm being smooth is equal to the probability of an integer of the same size being smooth[4] the left probability value in each of the summations will be identical for every value of i and can be computed using the method of [7] (a free implementation is available at [6] courtesy of the author). From the different distributions of the norms and integers we know that the right hand values are quite different and so the sums will clearly not be equal. The probabilities on the right in the $P(S_Z = 1)$ expression are exactly the s-quantiles (s is the number of partitions); as $P(Z \leq ia) = \frac{ia}{U}$ and so for all i we have:

$$P(z \in [(i-1)a, ia]) = P(Z \leq ia) - P(Z \leq (i-1)a) = \frac{(i-1)a - ia}{U} = \frac{a}{U}.$$

That is, the right probabilities in the $P(S_Z = 1)$ expression are constant and the expression becomes

$$\begin{aligned} P(S_Z = 1) &= \frac{a}{U} \sum_{i=1}^{n} P(S_Z = 1 | z \in [(i-1)a, ia]) \\ &= \frac{a}{U} \sum_{i=1}^{n} \frac{\#\text{smooth} \in [(i-1)a, ia]}{a} \\ &= \varPsi(\mathcal{S}, U)/U. \end{aligned}$$

Examining figure 5.1 we see that the probabilities $P(n \in [(i-1)a, ia])$ will be quite different. Using the law of total probability we not only highlight how the distribution of the norms affects the probability of smoothness, but present a method for computing a better estimate for the smoothness probability for the norms.

5.2 Estimating the smoothness probability of norms

To compute $P(S_N = 1)$ we use the same approach as above, though instead of having the partitions evenly spaced we use the s-quantiles

[4] The validity of this assumption will be discussed further in section 5.2.

of the distribution of the norms; that is, we fix the intervals such that the probability that n is in any interval is fixed at $\frac{1}{s}$. Fixed probability and even spacing are synonymous for the uniform distribution, but not for the distribution of the norms. Using R we easily compute the 100-quantiles of the norms, that is, we find the values $a_0, \ldots, a_{100}$, given in appendix A.2, such that $P(n \in [a_i, a_{i+1}]) = 0.01$ for $i \in [0, 99]$. It remains now, following from our assumption that $P(S_N = 1 | n \in [a_i, a_{i+1}]) = P(S_Z = 1 | z \in [a_i, a_{i+1}])$, to compute

$$0.01 \sum_{i=0}^{99} P(S_N = 1 | n \in [a_i, a_{i+1}])$$

to have the probability of smoothness of the norms. Using the implementation [6] we computed the probabilities in the summation to obtain an estimate of the smoothness probability:

$$P(S_N = 1) = 0.0001699201 = \hat{r} \quad \text{whereas} \quad P(S_Z = 1) = 0.00005777868.$$

The probability of smoothness of the norms over this interval is higher by a factor of almost 3 (2.94). This is no surprise as, intuitively, smaller numbers are more likely to be smooth and the norms have a higher concentration of ‘small’ to ‘medium’ numbers than the integers do. These results are reflected in experimental results obtained by selecting integers at random, following the distribution of the norms, and testing for smoothness; almost 3 times as many smooth integers were obtained using this method compared to selecting integers uniformly at random.

The probability above is an *optimistic-case* probability; we have made the assumption that the probability of a norm being smooth is equal to the probability that an integer of the same size is smooth. That is, for some integers b_0 and b_1 we assumed that

$$P(S_N = 1 | n \in [b_0, b_1]) = P(S_Z = 1 | z \in [b_0, b_1]).$$

This is, however, not necessarily the case. In any given interval, the integers are denser than norm values (approximately $1/n$) and the norms will not necessarily coincide with the smooth integers at the same rate as the actual proportion of smooth integers. As the number of quantiles grows, however, this assumption is better supported by the experimental results than the assumption used in the complexity analysis [14]. In the event that the norms fall on proportionally more smooth integers, the NFS sieving stage will execute faster. In the following table are examples of polynomials found for an example MNT6 prime for which the calculated rate r_1 (from $1000/\hat{r} \approx 5$mil tests).

$p = 1125909838976401$

polynomial	# smooth	$r =$ smooth rate	$\beta =$factor difference from $\hat{r}$
$f = 35374332x^6 + 11$	633	0.0001266	0.7451
$f = 11791444x^6 + 11$	1370	0.0002740	1.6125
$f = 17687166x^6 + 11$	652	0.0001304	0.7674
$f = x^6 + 8264689$	1379	0.0002758	1.6231
$f = 1069214x^6 + 37$	1320	0.0002640	1.5537

The values of β in the fourth column can be compared with the value $1/3 = 0.33\bar{3}$ which is used in the complexity analysis. We see the second, fourth and fifth polynomials produce smooth norms at a rate over 50% faster than the prediction (and so 4.5 times the estimate used in the complexity analysis). We were able to find many examples of polynomials yielding better than estimated performance. The average factor of difference from $\hat{r}$ is $\bar{\beta} = 0.5526999$ shown in Figure 5.2. The long tail of the graph distorts the standard deviation, but examining the quantiles, half of the values are between 0.28 and 0.71 with 1 being near the end of the main peak which is why we use the term *optimistic-case* probability. In figure 5.2 the value given by the asymptotic complexity is shown, it is the mode of the distribution; it is clear that a much larger proportion of the cases perform significantly better than this case. An NFS implementor would always endevour to launch the most efficient algorithm possible, thus the average is a better reflection of the rate than the mode when estimating the run time of the sieving stage and consequantly the security of a PBC protocol. The existing complexity analysis thus gives an underestimation of the probability of finding doubly-smooth pairs.

The goal of an implementer of the NFS is to find a polynomial *pair* such that the rate of smooth relation collection in each number field maximal, thus would aim to find pairs (f_1, f_2) such that $r_1 \cdot r_2 \approx \hat{r}^2$ (r_i = smooth rate of norms computed using f_i, $i \in \{1, 2\}$); this is no straightforward task. We denote $r_1 \cdot r_2 = \delta\hat{r}^2$ and recognise that the possible values for δ will be the product of independently selected values from the distribution of β and will therefore have mean $\bar{\beta}^2$. From our empirical data we computed 0.3320898 very close to the predicted $0.3062898 = 0.5526999^2$ (graph of distribution in appendix B). We can compare this to the estimate used in the complexity analysis which would give a value of $\delta = 1/9 = 0.11\bar{1}$. Even if we consider the median of this distribution, 0.2118865, to minimise the influence of the large, rare cases, it is still approximately twice the probability used in the complexity analysis and reflects more closely the observed behaviour from our experiments.

5.3 Polynomial Selection Test

The mean value of δ gives us a benchmark to aim to obtain, or improve on, when selecting our polynomials for the NFS. For a given instance of the DLP in a finite field $\mathbb{F}_{p^n}$ we perform the following test before launching the NFS sieving stage:

1 Compute the $\hat{r}$, the upper estimate on the smoothness probability of norms as done in section 5.2.
2 Find polynomial pair f_1 and f_2 following the method given in section 4.1 or [14].
3 Generate $1000/\hat{r}$ [8] norm values randomly using both f_1 and f_2 and test for smoothness to approximate the rates of smooth norm occurrence for each number field.
4 Compare the product of the approximate rates of smooth norms to $\hat{r}^2$: if it is “large enough”, proceed with sieving, otherwise return to step 1.

In step 4 by large enough we want to find $\hat{\delta} \geq 0.3320898$ (the mean value).

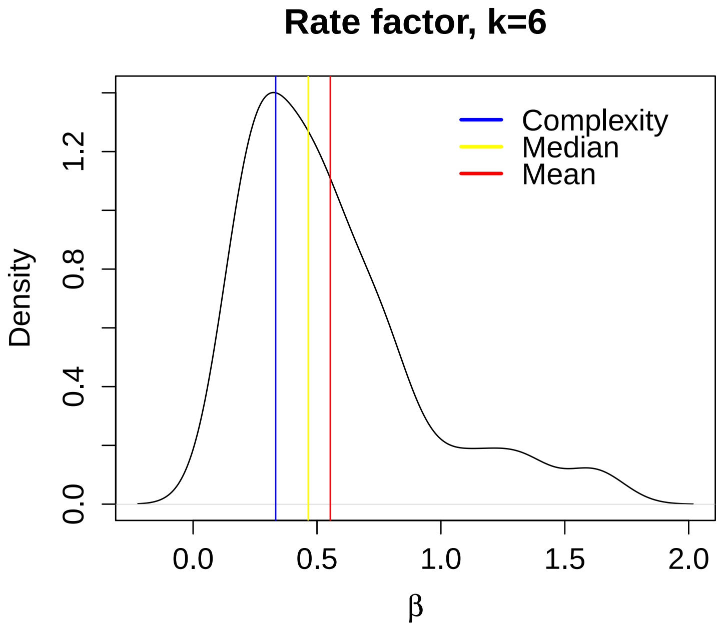


Fig. 5.2. The probability density function of the multiple by which the rate of smooth relation collection differs from the estimated rate.

This differs to how the results in [23] would be used, we have given a benchmark to compare the rate of smooth norm collection against, whereas previous methods would have required the generation of numerous pairs of polynomials to mutually compare the rate of smooth relation collection.[5] Thus, our method results in fewer tests necessary to distinguish if a pair of polynomials is performing at least as well as the average case.

To illustrate, we have generated various f_1 and f_2 pairs for our example case of MNT6 with prime around 50 bits. Our experiments resulted in polynomial pairs for which smooth norms occurred at a much higher rate than others. The following tables show some example cases of polynomial pairs for MNT6 case:

$p = 1125909838976401$

polynomial pair	# smooth	rate smooth	rate double smooth	$\hat{\delta}$
$f_1 = 35374332x^6 + 11$	633	1.266E-4	7.54536E-9	0.2613
$f_2 = x^6 + 31828441$	298	5.96E-5		
$f_1 = 11791444x^6 + 11$	1370	2.740E-4	1.17272E-8	0.4062
$f_2 = x^6 + 95485323$	214	4.28E-5		
$f_1 = 17687166x^6 + 11$	652	1.304E-4	5.42464E-9	0.1879
$f_2 = x^6 + 63656882$	208	4.16E-5		
$f_1 = x^6 + 8264689$	1379	2.758E-4	1.147328E-8	0.3974
$f_2 = 136231362x^6 + 17$	208	4.16E-5		
$f_1 = 1069214x^6 + 37$	1320	2.64E-4	4.9632E-9	0.1719
$f_2 = x^6 + 1053025717$	94	1.88E-5		

Interestingly, these examples highlight that simply taking a small value for i (11 in this case) does not ensure a good polynomial pair, the rate of smooth relation collection for the first three polynomial pairs is vastly different. Though the rate is comparable in the number fields defined using f_2, using the first and third pairs we found

[5] The values for BN and KSS18 cases are currently being computed using the same method.

smooth relations at less than half the rate of the second pair, for which the value $\hat{\delta} = 0.4062$ exceeds the mean value of 0.3320898.

$p = 1126169969103937$

polynomial pair	# smooth	r =est. double smooth rate	$\hat{\delta}$
$f_1 = x^6 + 4747150$	476	1.014832E-8	0.3515
$f_2 = 237230753x^6 + 13$	533		
$f_1 = x^6 + 186474550$	139	6.52744E-9	0.2261
$f_2 = 6039269x^6 + 13$	1174		
$f_1 = x^6 + 516240070$	118	5.1212E-9	0.1774
$f_2 = 2181485x^6 + 13$	1085		
$f_1 = x^6 + 20871667$	202	4.7268E-9	0.1637
$f_2 = 53956877x^6 + 22$	585		
$f_1 = x^6 + 27201934$	251	6.68664E-9	0.2316
$f_2 = 41400364x^6 + 39$	666		

Again in this table we see great variation in the rate of smooth element collection in the first three cases, for which the same value $i = 13$ was used to construct the polynomial pairs. For this prime the value of $\hat{\delta} = 0.3515$ slightly exceeds the mean value 0.3320898.

6 Conclusion and Open Problems

In theoretical complexity of algorithms it is common practice that many constant factors are hidden in the O and L notations. As a result, the theoretical complexity does not give a good reflection of how the algorithm will run in practice. In cryptography, the security of protocols is directly related to the *practicality* of solving hard mathematical problems. Until now the practical behaviour of the number field sieve in the medium prime case has received relatively little attention. Due to its relevance to the security of pairing-based protocols in particular we are examining the run-time behaviour of this algorithm in the given context; the motivation of this work was to deepen the understanding of the NFS in general, in the hope that this would help determining more precise estimates for appropriate parameter sizes for a fixed security level for PBC protocols.

In this work we presented some observations on the behaviour of the NFS in practice. We focussed on the smoothness probability of the norms of number field elements as it determines the practical run-time of the sieving stage. Our observations and analysis have resulted in a pre-sieving test that can be performed on the selected polynomials. This test ensures as efficient a set up and execution of the sieving stage as possible. The new revelation about the probability of smooth norm occurrence is a step towards a more precise run-time estimate of the sieving stage. We also gave a variation of the polynomial selection method given in [14] used to conduct our experiments, which shows promising behaviour.

This work covers the initial progress in the examination of the NFS sieving stage; in order to compute the expected run-time of this stage it remains to investigate the true cost of smoothness test (relative to computing power). Other tasks include a thorough comparison of the polynomial selection methods as given in section 4.1 and [14].

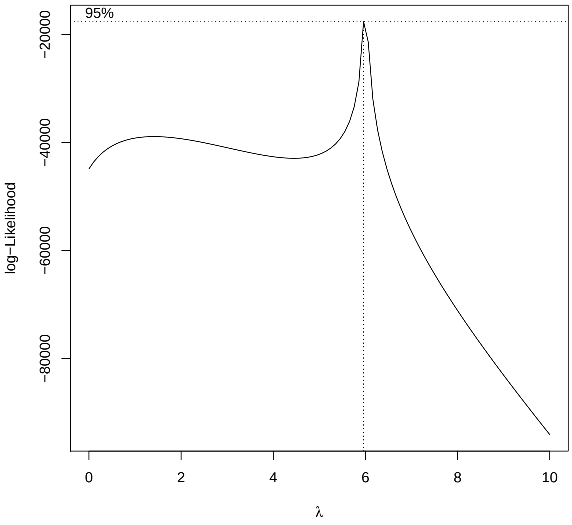
References

1. ECRYPT II Yearly Report on Algorithms and Key Lengths, 2011.
2. S. Bai and R. P. Brent. On the efficiency of pollards rho method for discrete logarithms. *CATS*, pages 125–131, 2008.
3. D. V. Bailey, B. Baldwin, L. Batina, D. J. Bernstein, P. Birkner, J. W. Bos, G. V. Damme, G. Meulenaer, J. Fan, T. Güneysu, F. Gurkaynak, T. Kleinjung, T. Lange, N. Mentens, C. Paar, F. Regazzoni, P. Schwabe, and L. Uhsadel. The Certicom Challenges ECC2-X, 2009.
4. D. V. Bailey, L. Batina, D. J. Bernstein, P. Birkner, J. W. Bos, C. Chen, G. V. Damme, G. Meulenaer, L. J. Dominguez-Perez, J. Fan, T. Güneysu, F. Gurkaynak, T. Kleinjung, T. Lange, N. Mentens, R. Niederhagen, C. Paar, F. Regazzoni, P. Schwabe, L. Uhsadel, A. V. Herrewege, and B. Yang. Breaking ECC2K-130. *Cryptology ePrint Archive*, (541), 2009.
5. N. Benger and M. Scott. Constructing Tower Extensions for the implementation of Pairing-Based Cryptography, 2010.
6. D. J. Bernstein. implementation of ψ estimation method of "arbitrarily tight bounds on the distribution of smooth integers".
7. D. J. Bernstein. Arbitrarily tight bounds on the distribution of smooth integers. *Number theory for the Millennium*, 1:49–66, 2002.
8. D. J. Bernstein. Predicting NFS time, 2008. Slides of a talk given at Cado Workshop on Integer Factorization October 7, 2008. LORIA, Nancy, France.

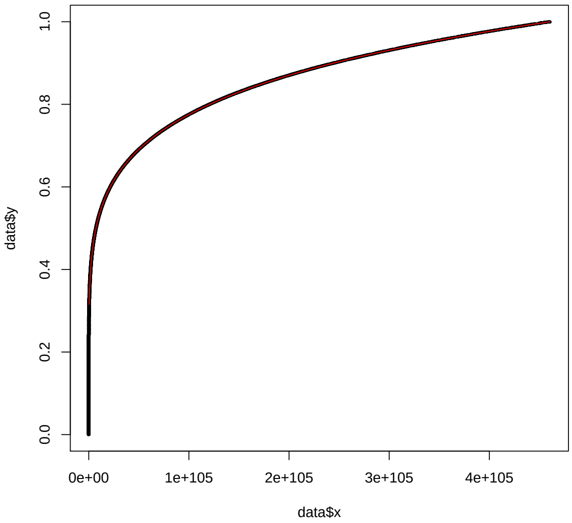
9. G.E.P. Box and D.R. Cox. An analysis of transformations (with discussion). *Journal of the Royal Statistical Society*, 26:211–252, 1964.
10. E. R. Canfield, P. Erdős, and C. Pomerance. On a problem of Oppenheim concerning "Factorisatio Numerorum". *Journal of Number Theory*, 17:1–28, 1983.
11. Faruk Göloğlu, Robert Granger, Gary McGuire, and Jens Zumbrägel. On the function field sieve and the impact of higher splitting probabilities: Application to discrete logarithms in $\mathbb{F}_{2^{1971}}$ and $\mathbb{F}_{2^{3164}}$. Cryptology ePrint Archive, Report 2013/074, 2013. To appear at Crypto 2013.
12. Faruk Göloğlu, Robert Granger, Gary McGuire, and Jens Zumbrägel. Solving a 6120-bit dlp on a desktop computer. Cryptology ePrint Archive, Report 2013/306, 2013.
13. A. Hildebrand and G. Tenenbaum. Integers without large prime factors. *JTNB*, 5:411–484, 1993.
14. A. Joux, R. Lercier, N. Smart, and F. Vercauteren. The Number Field Sieve in the Medium Prime Case. In C. Dwork, editor, *Advances in Cryptology – Crypto 2006*, number 4117 in Lecture Notes in Computer Science, pages 323–341, 2006.
15. Antoine Joux. Faster index calculus for the medium prime case application to 1175-bit and 1425-bit finite fields. In *Advances in Cryptology - EUROCRYPT 2013*, LNCS, pages 177–193. Springer, 2013.
16. N. Koblitz and A. Menezes. Pairing-Based Cryptography at high security levels. *IMA Int. Conf.*, 3796:13 – 36, 2005.
17. U Krause. Abschätzungen für die funktion $\psi_k(x, y)$ in algebraischen zahlkörpern. *Manuscripta Mathematica*, 69:319 – 331, 1990.
18. A. K. Lenstra. Unbelievable Security: Matching AES Security Using Public Key Systems. In C. Boyd, editor, *Advances in Cryptology – Asiacrypt '01*, volume 2248 of *Lecture Notes in Computer Science*, pages 67–86, London, UK, 2001. Springer-Verlag.
19. A. K. Lenstra and E. R. Verheul. Selecting Cryptographic Key Sizes. In H. Imai and Y. Zheng, editors, *Public-Key Cryptography 2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 446–465, London, UK, 2000. Springer-Verlag.
20. J. M. Pollard. Monte Carlo methods for index computation (mod p). *Mathematics of Computation*, 32:918–924, 1978.
21. R Core Team. *R: A Language and Environment for Statistical Computing.* Vienna, Austria, 2010.
22. E. Teske. Speeding up pollards rho method for computing discrete logarithms. In *Algorithmic Number Theory Symposium (ANTS IV)*, number 1423 in LNCS, pages 541–553. Springer-Verlag, 1998.
23. P. Zajac. Basic remarks on the NFS complexity. Cryptology ePrint Archive, Report 2008/064, 2008. `http://eprint.iacr.org/`.

A MNT6 output

Diagram from Box-Cox method clearly indicating that for the MNT6 case that $\lambda = 6$.



Comparison of the experimental data and the fitted line.



A.1 Output of the linear model

Output from R:

```
Residuals:
Min            1Q           Median       3Q          Max
-8.589e-04     -7.149e-05   3.580e-06    1.868e-05   7.928e-04

Coefficients:
               Estimate        Std. Error       t value      Pr(> |t|)
(Intercept)   −1.868e − 05    1.701e − 07      −109.8       < 2e − 16
x             2.171e − 0106   1.331e − 112     1630932.3    < 2e − 16
Residual standard error: 0.0002061 on 1999998 degrees of freedom
Multiple R-squared:   1,   Adjusted R-squared:   1
F-statistic: 2.66e + 12 on 1 and 1999998 DF, p-value: < 2.2e − 16
```

A.2 MNT6 100-quantiles

a_0 =71058 96343 98717 100100 101069 101816 102434 103022 103837 104597

a_{10} =105289 105918 106496 107028 107524 107984 108412 108814 109192 109551

a_{20} =109893 110222 110534 110829 111113 111383 111644 111894 112137 112371

a_{30} =112596 112816 113027 113235 113434 113629 113817 114000 114180 114353

a_{40} =114522 114687 114848 115002 115154 115304 115449 115594 115735 115871

a_{50} =116004 116136 116265 116392 116517 116641 116762 116881 116997 117110

a_{60} =117221 117331 117440 117546 117650 117754 117856 117956 118055 118152

a_{70} =118249 118342 118435 118527 118618 118707 118795 118884 118971 119056

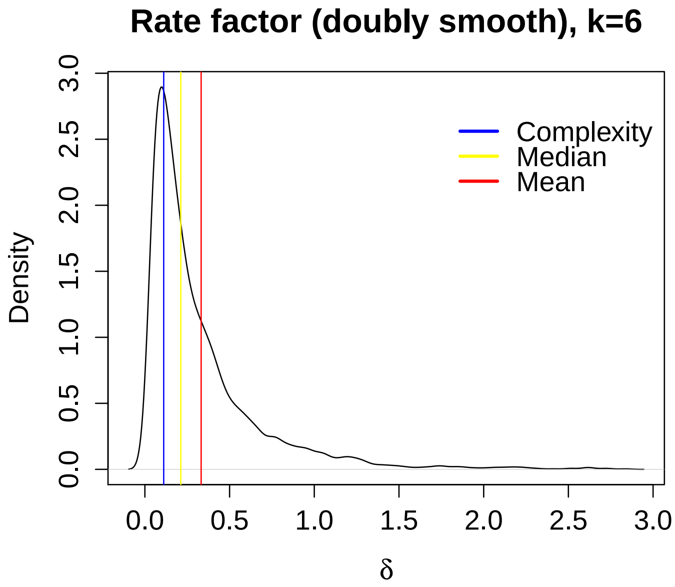
a_{80} =119140 119224 119306 119386 119465 119544 119622 119698 119775 119850

a_{90} =119925 119999 120073 120144 120215 120286 120356 120424 120493 120561

a_{100} =120628

When N is a random variable which takes values from the range of the function $n(X_0, X_1, X_2, X_3) = A^2X_0^6 - 2ABX_0^3X_2^3 + 9ABX_0^2X_1^2X_2^2 - 6ABX_0X_1^4X_2 + ABX_1^6 + B^2X_2^6$ where X_i are uniformly distributed discrete random variables in the interval $[0, 349879]$. The above values are such that for $N = n$, $P(n \in [a_i, a_{i+1}]) = 0.01$ for $i \in [0, 99]$.

B Rate multiple (doubly smooth) distribution



Again, as in figure 5.2 the value given by the asymptotic complexity is shown, it is indeed the the value which occus with the highest probability. We see again that a much larger proportion of the cases perform significantly better than this case, including the mean case, which the empirical data more closely reflects. We reach the same conclusion that the complexity analysis gives a misleading estimate of the average case.